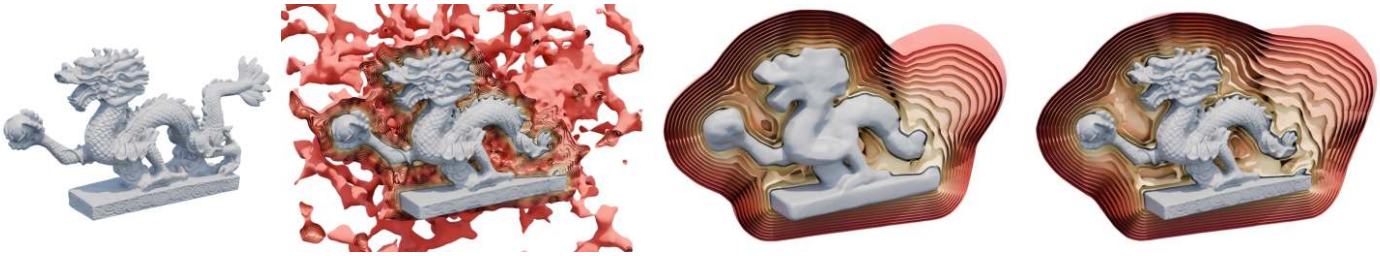
Eurographics Symposium on Geometry Processing 2026
M. Skouras and M. Livesu
(Guest Editors)

COMPUTER GRAPHICS *forum*
Volume 45 (2026), Number 5

Compactly supported detail field for high quality neural implicit surfaces

Guillaume Coiffier[1] and Justine Basselin[2]

[1]Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LJK
[2]Independent researcher



(a) Original 3D Model (b) SirenNet [SMB*20] (c) 1-Lipschitz network [CB24] (d) Ours (Lipschitz net + detail field)

Figure 1: Neural implicit reconstruction of the Dragon model from threedscans.com. We display the full zero level set and slices of regularly spaced isosurfaces between values 0 and 0.2. (a) Original 3D model. (b) A SirenNet [SMB*20] is able to accurately capture the geometry but the implicit function is highly irregular far from its zero level set. (c) A 1-Lipschitz neural network [CB24] provides a smoothly increasing function but fails to capture surface details. (d) Our method combines a 1-Lipschitz neural function with a compactly supported detail field and achieves both a highly detailed zero level set and smooth isosurfaces.

Abstract

Neural implicit surfaces are a powerful tool for encoding a surface as the zero level set of a neural function. Trained using gradient-descent based optimizers, these methods however suffer from a low-frequency bias that prevents them to fit fine details of the surface. Solutions to break this bias exist but often damage the regularity of the long range implicit function, making geometrical queries more difficult. In this work, we combine a 1-Lipschitz neural function with a detail field made of compactly supported radial basis functions. While the neural function handles long-range queries with low surface accuracy, the detail field is fitted to perturb the neural function so that its zero level set precisely goes through sampled points of the surface and locally accounts for small-scale details. As it is compactly supported, regions where the 1-Lipschitz property breaks are finely controlled. We therefore achieve the best of both worlds by greatly increasing the representation power of the implicit function near the zero level set while maintaining robustness far from the zero level set.

CCS Concepts

• ***Computing methodologies*** → *Mesh geometry models; Point-based models;* ***Neural networks;***

1. Introduction

Implicit surfaces offer an efficient and expressive representation of geometrical shapes as the level set of a continuous function, with applications in surface reconstruction [HCJ19], constructive solid geometry [Ric73], rendering [Har95] or numerical simulation [OS88]. Unlike other widespread geometrical representations like meshes or voxel grids, implicit surfaces do not rely on a discretization of space nor a combinatorial structure, which allows for easy topological changes or boolean operations.

Over the years, a variety of techniques has been proposed to represent surfaces using implicit functions. For simple shapes, a closed form expression of the signed distance function (SDF) may be available and easily computable [Qui]. For more complex objects, two main approaches have been developed. The first one consists in combining simple primitives into a tree of blending operations [WGG99]. The second one expresses the target implicit function as a linear combination of basis functions which may be defined either as a collection of finite elements, leading to grid-based

or mesh-based representations [Set96] or as less-structured sets like radial basis functions (RBF) [CBC*01].

More recently, neural implicit surfaces have emerged as a third way of encoding an implicit function by utilizing the parameters of a deep neural network [PFS*19, GYH*20, XTS*22], addressing many of the limitations of classical methods. They have proven to be very efficient in terms of memory footprint [DNJ21], yet they often struggle to reach high levels of geometrical details, even at very high parameter count. The optimization process indeed suffers from a documented *low-frequency bias* that favors large-scale features and makes the capture of small-scale details increasingly difficult. Different techniques, such as positional encoding [TSM*20] or changes in the neural network's architecture [SMB*20] were introduced to tackle this phenomenon, but the reconstruction accuracy was gained at the expense of the regularity of the overall implicit function (see Figure 1). As a result, the final implicit function is more difficult to use in most applications and requires specifically-designed query algorithms to guarantee geometrical robustness [SJ22].

In this work, we introduce a novel method to perturb a trained neural distance function to significantly improve its capture of surface details. A second implicit function, defined as a sum of compactly supported radial basis functions, is added to the output of the neural network, and fitted so that the zero level set of the sum goes precisely through some prescribed points on the target surface. The support of this detail field is defined to only have effect in a narrow band around the zero level set, leaving all the long-range queries to the neural network. By using a neural network whose function is well-behaved far from the zero level set [CB24], we therefore achieve the best of both worlds: a highly accurate representation of the target surface and a very regular distance field far from the surface.

2. Related Work

2.1. Implicit modeling for surface reconstruction

Implicit surfaces have a long and rich history in computer graphics [BB97]. Taking inspiration from scattered data interpolation problems, they have been extensively used to reconstruct surfaces from point clouds. Given points $\{x_i\}_{1\leqslant i\leqslant N}$ in $\mathbb{R}^n$, the idea is to build an interpolating function f that is smooth and whose zero level set includes the points x_i. Early works consider a sum of radial basis functions (RBF) [Buh00] as interpolant f:

$$f(x) = \sum_{i=1}^{N} w_i \, \phi(||x - x_i||).$$

Duchon [Duc77] showed that using $\phi(r) = r^2 \log(r)$ for three-dimensional data would lead to the interpolant minimizing the thin-plate energy. However, solving for the specific weights w_i requires to solve a dense linear problem of size $N \times N$, which lead to performance issues for more than a few thousand points [DLR86, TO99]. Faster methods for fitting a RBF to a large point cloud were later introduced [BCM99, GR87, CBC*01] but they usually require the definition of offset points along the points normal in order to avoid the trivially zero solution. Hermite RBFs [MGV11] instead rely on surface normals to also be fitted correctly. Their performance was later improved by Liu et al. [LWBW16] who introduced an approximate closed-form solution to the problem. More recently, VIPSS [HCJ19] established and solved a quadratic problem which solution is a unit gradient implicit field satisfying pointwise value constraints and minimizing some smoothness energy. The scalability of this method was later improved by using natural neighbors [XJ25]. These methods focus on the reconstruction of the surface and do not however output a distance field.

In parallel, more local approaches were considered. Methods based on moving least squares [Lev04, ABC*01] instead define and deform local patches to represent a surface from a projection operator. Ohtake et al. [OBA*03] consider a blending of patching with weights that partition the unity and improve performance using a multi-level representation. Tobor et al. [TRS04] improve on this idea in terms of performance and robustness to noise. The *implicit moving least squares* (IMLS) surface [Kol08] defines a closed-form function that locally approximates the signed distance field of the input, but suffers from either a very narrow support around input points or excessive smoothing of the shape.

For a local RBF interpolant, compactly-supported radial basis functions ϕ were also proposed [Wen95]. This makes the fitting process into a sparse linear system, which greatly improves scalability [MYR*05], yet the resulting implicit function is again restricted to a narrow neighborhood of the input. Pan et al. [PS11] proposed to still represent a distance field like this by spreading some RBF centers over the whole domain, but their approach require points outside the surface as well as ground truth distance to fit the surface. In this work, we consider compactly supported RBF as this limited support is actually an advantage in our case: it guarantees that the perturbation field only has effect on a restricted zone around the surface.

2.2. Neural implicit representations

Neural implicit representations [XTS*22] have gained traction in recent years to represent arbitrary fields (color, visual aspect, distance, *etc*.) using a neural network. In contrast to classical techniques, they require no discretization nor sampling of the input shape and are naturally defined over the full space. Early works in this domain encoded the shape as a binary occupancy field [MON*19]. Park et al. [PFS*19] were the first to encode the shape as a *signed distance field*, enabling geometry processing applications like sphere tracing [Har95], offset surfaces [FP06] or smooth blending [Bli82] on neural representations. Gropp et al. [GYH*20] then proposed to regularize the neural field by adding the so-called *eikonal loss*, which penalizes non-unit norm gradients. More recently, methods explored the theory of heat diffusion to define loss functions whose only minimizer is the signed distance field [WWY*25, WHS*], or use specific neural architectures to force the implicit field to underestimate the distance function [CB24]. In this work, we rely on the latter to provide a provably robust neural distance field.

2.3. Low frequency bias in neural implicit representations

As neural implicit representations rely on first order optimizers like stochastic gradient descent or Adam [KB14], they often strug-

gle to capture small scale details, even when the network is over-parametrized. This empirical observation was reported by several previous works [XZX19, BGG*20, RBA*19] which describe it as a *spectral bias* towards low frequencies. During training, large-scale structures are naturally prioritized as a better descent direction for loss minimization, which leaves the final result often stuck in a local minimum where fine details are ignored. While this bias is useful in context where the network needs to generalize without overfitting, it needs to be avoided in neural implicit representations, as their goal is precisely to overfit on the input shape.

An approach to counter the low-frequency bias is to first map the input query of the neural network to a different space, either high frequency trigonometric functions [TSM*20, ZRL21] or splines [WLYT21]. Siren networks [SMB*20] modify their non-linear activation function to $x \mapsto \sin(\omega x)$ to achieve the same effect. These techniques allow for an almost perfect capture of the target shape, but lead to less regular implicit functions, even in the presence of a regularization loss [CB24]. In contrast, our method only perturbs the implicit function near its zero level set, leaving the function mostly unchanged.

Combining neural implicit functions with classical approaches for geometry representations has also been attempted as a way to improve reconstruction accuracy of neural fields. Muller et al. [MESK22] make a neural network interpolate feature vectors defined on a multi-resolution grid, while Takikawa et al. [TLY*21] use a similar mechanism over an octree to represent different levels of detail. These methods again provide great reconstruction accuracy but the regularity of the implicit function far from the level set is again not guaranteed.

Finally, the work of Yifan et al. [YRS22] is perhaps the closest to ours. The authors use a pair of Siren networks with different frequencies to represent a base implicit representation and an implicit displacement field, trained simultaneously to fit the overall shape of the object and the small scale details. While the reconstruction of surfaces is very accurate, this method suffers from the same drawbacks of regular Siren networks in terms of the distance field's regularity.

3. Method

Let Ω be a compact subset of $\mathbb{R}^n$ (with $n = 2$ or 3) and $\partial\Omega$ be its boundary.

The *signed distance function* (SDF) of Ω is defined for all $x \in \mathbb{R}^n$ as the distance from x to the closest point on the boundary of Ω, counted negatively if x is inside Ω and positively otherwise:

$$S_\Omega(x) = \begin{cases} \min_{p\in\partial\Omega} ||x-p|| & \text{if } x \notin \Omega \\ -\min_{p\in\partial\Omega} ||x-p|| & \text{otherwise.} \end{cases}$$

The signed distance function is differentiable everywhere except on the medial axis of the shape. A key property of its gradient is that it has unit norm and points away from the closest point on $\partial\Omega$. Moreover, the sphere of center r and of radius $S_\Omega(r)$ is always tangent to the boundary $\partial\Omega$. This means that having access to the SDF instead of an arbitrary implicit function allows for easier geometrical queries like closest point projection:

$$\pi_{\partial\Omega}(x) = x - S_\Omega(x)\nabla S_\Omega(x) \tag{1}$$

which enables applications such as rendering via sphere tracing [Har95], normal estimation or collision detection [MEM*20].

Our goal is to build a high quality approximation of S_Ω. As it remains a challenge to compute a representation that preserves both the differential properties of the distance field and the geometrical details of its zero level set, we propose to split our approximation into two terms:

$$S_\Omega(x) \simeq f_\theta(x) + d(x)$$

where f_θ will be a neural network parametrized by $\theta \in \mathbb{R}^P$ (Section 3.1) and d will be an explicit detail field computed as a sum of compactly supported RBFs whose centers are sampled on the surface $\partial\Omega$ (Section 3.2). Representing the distance function that way allows to split the requirements into two. On the one hand, the neural field will provide a low-detail approximation of the signed distance field that is defined over all space. On the other hand, the detail field will correct its values near its zero level set and will provide the additional degrees of freedom needed to accurately represent the input surface.

3.1. Fitting the neural implicit function

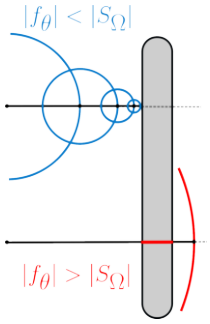
Our neural implicit function f_θ will be represented by a "coordinate neural network", taking as input one point of $\mathbb{R}^n$ and computing the value of the target function at this point. Optimizing such a neural network using methods based on gradient descent remains challenging, as they usually converge to a low detail solution [XZX19]. Computing a more detailed representation while preserving properties of the signed distance function implies creating more gradient discontinuities in the distance field. As such, there seems to be an inevitable tradeoff between reconstruction quality and distance field regularity. While most methods choose to sacrifice the long-distance field for a better reconstruction of the zero level set [SMB*20, TSM*20], we instead focus on the field's regularity and leave the capture of small features to the detail field. To this end, we rely on the method described by Coiffier and Béthune [CB24] which computes a 1-Lipschitz approximation of the signed distance field. Recall that the function f_θ is 1-Lipschitz if:

$$\forall x, y \in \mathbb{R}^n, \quad |f_\theta(x) - f_\theta(y)| \leqslant ||x-y||.$$

For a differentiable function, it is equivalent to require that its gradient never exceeds unit norm:

$$\forall x \in \mathbb{R}^n, \quad ||\nabla f_\theta(x)|| \leqslant 1.$$

Having a 1-Lipschitz neural field means that the implicit function is *conservative* and never overestimates the true distance to the surface $\partial\Omega$. As such, the sphere centered at any point x or radius $|f_\theta(x)|$ never intersects the zero level set. This property is essential in applications like ray marching [Har95, GGPP20] or uniform sampling of the surface [LMSJ25], where intersections between rays and the shape need to be computed.



If the Lipschitz constant is not known or bounded, the ray may indeed "overshoot" the surface, and repairing the error cases requires additional care and computational overhead [SJ22] (see inset Figure, red). On the other hand, a conservative implicit function guarantees that the ray marching procedure will converge to its first intersection with the shape (inset Figure, blue).

To obtain a 1-Lipschitz neural function, we first define some Lipschitz-constrained neural network [AHD*23] and optimize its parameters in order to minimize the *hinge-Kantorovitch-Rubinstein* (hKR) loss function [SMG*21]:

$$\min_{\theta} \; -y(x) f_\theta(x) + \lambda \max(0,\, m - y(x) f_\theta(x))$$

where m and λ are fixed parameters, and y is an inside/outside predicate for Ω: $y(x) = -1$ inside and $+1$ outside of Ω.

Although Ω could also be represented as a triangle soup or an oriented point cloud, we consider triangle meshes as an input in all our practical experiments. This allows y to be computed efficiently using the generalized winding number [BDS*18].

3.2. Computing the detail field

Once the neural network f_θ is trained, we obtain a low frequency 1-Lipschitz approximation of the signed distance field where all small scale details are unfortunately lost. Minimizing the hKR loss with a margin parameter m indeed introduces an imprecision onto the zero level set of order m [CB24], which hinders the ability of the method to capture fine features. Here, we propose to recover those features by perturbing the neural function so that its zero level set precisely goes through prescribed points. Let $\mathcal{X} = \{x_i\}_{1 \leqslant i \leqslant N}$ be N points sampled uniformly on the surface $\partial\Omega$. We define the detail field d as a sum of compactly supported radial basis functions centered at the x_i:

$$d(x) = \sum_{i=1}^{N} w_i \, . \, \phi\left(\frac{||x - x_i||}{\sigma}\right) \tag{2}$$

where:

$$\phi(r) = \begin{cases} (1-|r|)^2 & \text{if } |r| \leqslant 1 \\ 0 & \text{otherwise} \end{cases}.$$

The N weights $w_i \in \mathbb{R}$ are uniquely determined by the N constraints stating that d should compensate the value of f_θ at each x_i in order to force the zero level set to stay on $\partial\Omega$:

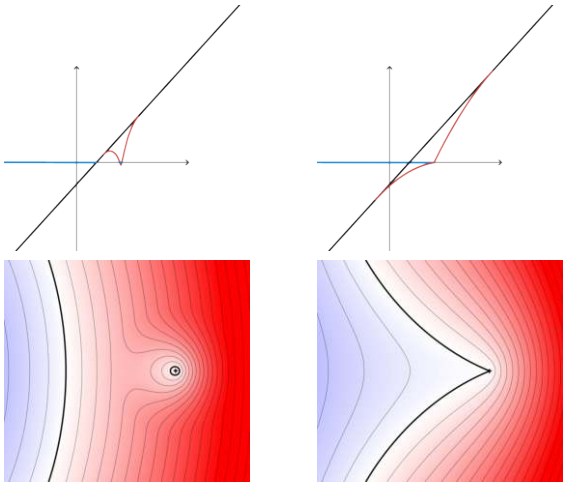
$$\forall i \in \{1, ..., N\}, \quad d(x_i) = -f_\theta(x_i).$$

This leads to a linear system:

$$Aw = -\begin{pmatrix} f_\theta(x_1) & \dots & f_\theta(x_N) \end{pmatrix}^T \tag{3}$$

where:

$$A_{ij} = \phi\left(\frac{||x_i - x_j||}{\sigma}\right).$$



(a) (b)

Figure 2: Modification of an implicit function using a single radial basis function, shown as a 1D example (top) and a 2D level set (bottom). (a) If the support of the basis function becomes too small, the zero level set may undergo topological changes, potentially introducing disconnected components, looking like "bubbles". (b) For a large enough support size, the level set is moved without altering its topology.

Due to the use of a compactly supported function ϕ, the matrix A is sparse and its fill ratio depends on the support size parameter σ. Moreover, A was shown to be symmetric positive definite for this specific function ϕ for any value of the x_i [Wen95]. Hence the system can be efficiently solved using a sparse conjugate gradient solver.

Once the weights w_i are computed, the detail field is queried at a position $x \in \mathbb{R}^n$ using a space partitioning data structure such as a kd-tree over $\mathcal{X}$. This structure allows us to efficiently find all RBF centers x_i that are at distance at most σ of x and thus restrain the computation of the sum (Eq. (2)) to only basis functions that are contributing.

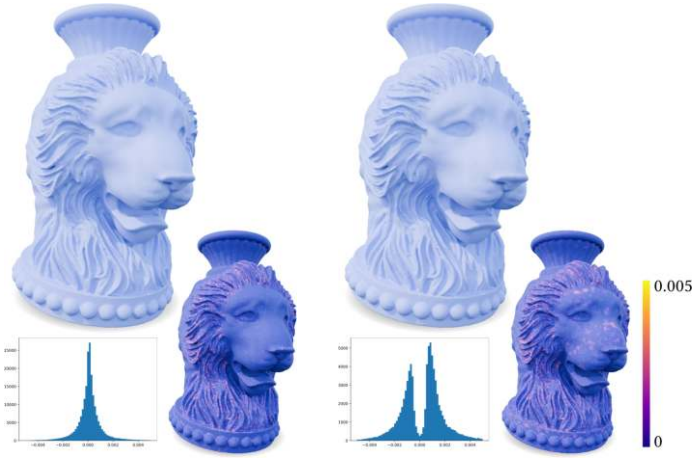
3.3. Choice of support size σ

In the formulation of the detail field (Eq. (2)), the parameter σ directly controls the support size of each basis function. Its value depends on the sampling $\{x_i\}$ but is hard to estimate in practice. On the one hand, it should be made as small as possible in order to preserve the sparsity of the linear system in Eq. (3) and overall performance of the method. On the other hand, a σ that is too small can lead to a detail field that is not expressive enough or can lead to spurious components in the zero level set (Figure 2), which is undesirable.

We determine the optimal value of σ experimentally. First, notice that the zero level set of f_θ should be completely included in the support of d, as it would otherwise be impossible for the detail field to have an influence there. Therefore, the support size σ naturally depends on the error made by the neural function. To account for

(a) $\sigma = 0.1L$ (b) $\sigma = 0.4L$ (c) $\sigma = L$ (d) $\sigma = 2L$

Figure 3: Variation of the RBF support size σ on the *Gargoyle* model. Reconstructed zero level sets are displayed in blue while distance error to the ground truth and extent of the detail field's support are displayed on the bottom left and bottom right respectively. (a) The detail field is unable to correct the level set since its support does not include the whole surface. (b) σ is large enough to include the full level set but not large enough to perturb it sufficiently, yielding a noisy output. (c) σ is large enough to provide a good quality approximation. (d) Making σ larger does not improve the quality of the reconstruction anymore but increases computation time. In practice, we set $\sigma = 1.1L$.



(a) Original reconstruction with full detail field (200k basis functions). (b) Pruned reconstruction (81k basis functions).

Figure 4: (a) Surface reconstruction on the *Lion vase* dataset with a full detail field computed with 200k centers sampled uniformly over the surface. (b) Same reconstruction after pruning basis functions with weights smaller than 5×10^{-4} in absolute value. Histograms show the distribution of weights w_i associated with each basis function. Heat map views show the distance to the ground truth mesh. On this example, we are able to remove almost 60% of the points with little to no reconstruction error.

this dependency, we start by computing the maximal distance L between a sample point in $\mathcal{X}$ and the zero level set of f_θ:

$$L = \max_{x \in \mathcal{X}} ||x - \pi_{f_\theta}(x)||$$

where π is the orthogonal projection to the zero level set of f_θ, computed via iterations of Eq. (1). We then set $\sigma = 1.1L$ to have a small margin. While this value does not guarantee inclusion of the zero level set of f_θ in the support of d, we determined experimentally (see Figure 3) that this value was sufficient for applications as long as the sampling $\mathcal{X}$ is dense enough.

3.4. RBF pruning

The points of $\mathcal{X}$ are sampled uniformly over $\partial\Omega$, which leads to a uniform coverage. However, the input dataset may present large areas of low details where the surface is already perfectly captured and where a detail field is unnecessary. This is illustrated in Figure 4 on the *Lion vase* dataset, where the top and bottom part are flat and most of the details happen in the lion's mane. With 200k base functions, the detail field is able to reconstruct the input mesh with great accuracy. Yet the distribution of weights w_i computed form the linear system of Eq. (3) indicate that a large amount of basis functions have a weight close to zero, which means that their contribution to the overall detail field is limited.

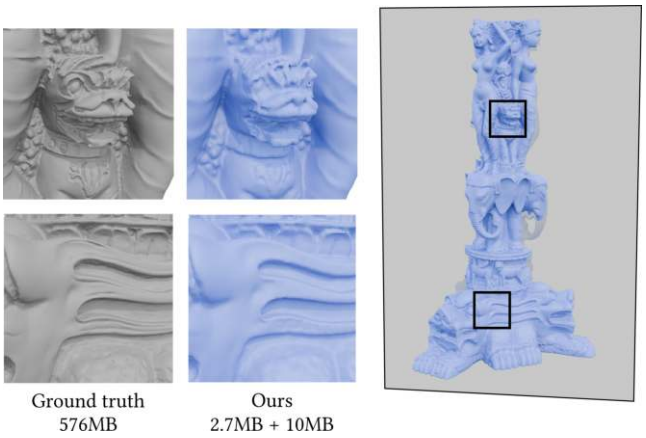
We therefore develop a pruning strategy to only keep meaningful basis functions by removing every center with a weight smaller than a threshold in absolute value. After this first filtering, we also remove isolated base functions, that is points x_i that are at distance at least σ from their nearest neighbors. Then, the RBF system of Eq. 3 is solved again using only this subset of points. This heuristic allows us to better estimate the number of samples needed to accurately fit the target surface.

4. Experiments

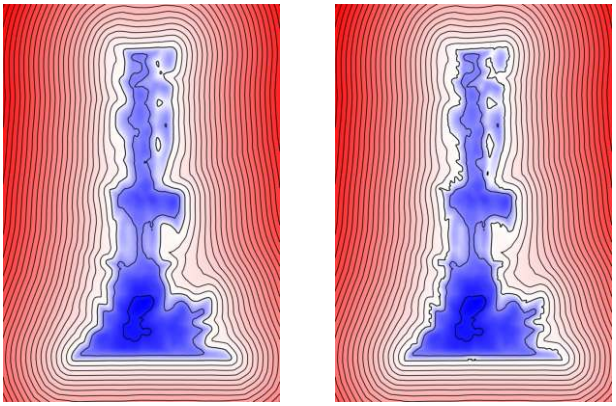
4.1. Experimental setup

All our experiments were conducted on a Ubuntu 24.04 using a Nvidia RTX PRO Blackwell 2000 GPU. The method was implemented in Python 3.12 using pytorch [PGM*19] for neural network training and scipy [VGO*20] for its kd-tree implementation. An implementation of our method is available at https://github.com/GCoiffier/neural-sdf-detail-field.

The geometrical inputs of our algorithm are scaled to fit into $[-1, 1]^n$. 500k points are then sampled in the domain, with 100k uniformly and 400k sampled near the surface by first sampling a



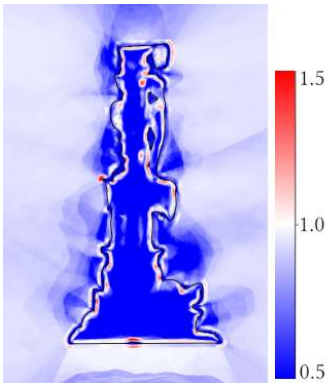
(a) Reconstruction using marching cubes.



(b) Level sets *without* detail field (neural function only)

(c) Level sets *with* detail field (neural function + RBF field)

(d) Support of the detail field



(e) Gradient norm

Figure 5: Reconstruction of the *Thai statue* dataset using 500k RBF centers. The input mesh is correctly modeled while being compressed by a ratio of 50. As the detail field has compact support around the zero level set (d), the isovalues of the implicit function are minimally perturbed (c). Since the neural function was 1-Lipschitz, areas where the gradient of the full function exceeds unit norm are localized around the zero level set (e).

surface point and perturbing it by a normal distribution of standard variation 0.02. For the neural field, we use the same 1-Lipschitz neural architecture as [CB24] with either 10 layers of size 128 (166k parameters, 677KB) or 10 layers of size 256 (660k parameters, 2.7MB) for more complex models. We train these networks with the *Muon* optimizer [Muo] with a learning rate of 10^{-4} as it has experimentally been showed to converge faster to more accurate results in the context of neural fields [MHS*25]. We set $m = 0.01$ and $\lambda = 10$ at the start of the training. We train for 300 epochs and multiply λ by 10 every 100 epoch. This training procedure takes up to an hour.

For the detail field, we sample between 50k and 500k points uniformly over the surface to form the set of RBF centers $\mathcal{X}$. We adjust this number of samples depending on the complexity of the model, while 100k is the default value of our algorithm. Building and solving the linear system of Eq. (3) for 100k centers takes up to 5 minutes, while higher center counts can take up to 15 minutes. The detail field can then be completely described using the positions of RBF centers, the vector of weights w_i and the value of σ. For 100k points, this information represents 2.2MB of data.

4.2. Results

We run our algorithm on a variety of surface meshes, including both high-level features and fine details. Meshes were taken from the Thingi10k dataset [ZJ16], the Stanford 3D Scanning repository [Sta], `threedscans.com` as well as from user *YahooJAPAN* from thingiverse [Yah].

Results shown in the figures were rendered by extracting a mesh from the zero level set of our final implicit function. For this, we use the standard marching cube algorithm [LC87] with resolution 400^3. A gallery of reconstructed level sets from a variety of models is available in the supplemental.

On Figure 1 and 5, we demonstrate that our method is able to capture the input geometry with great accuracy, while only minimally modifying the zero level set. For a pure RBF implicit function, this compact support would have been a drawback as long-range queries to the implicit field would yield a constant value. Here, these queries are handled by the 1-Lipschitz neural network alone, so that we inherit its robustness properties. This is shown in Figure 5 where we display the norm of the implicit function's gradient over a cut. We can see that, as expected, the gradient can only exceed unit norm in the compact support of the detail field, making the function 1-Lipschitz outside of the support. Inside the support, the total implicit function $f_\theta + d$ remains Lipschitz as a sum of Lipschitz functions, though its Lipschitz constant can only be loosely bounded by $L+1$ where L is the maximum number of overlapping RBFs. This structure of the total implicit field hints at the modification needed for ray intersections to be computed on our model: first consider standard ray marching up until the σ-level set of f_θ is reached, then consider either smaller steps by querying $\frac{f_\theta + d}{L+1}$ or robust query algorithms like range analysis [SJ22].

4.3. Partial detail fields

As our detail field is non-zero only around balls centered at points of $\mathcal{X}$, it is possible to control where the details should be applied

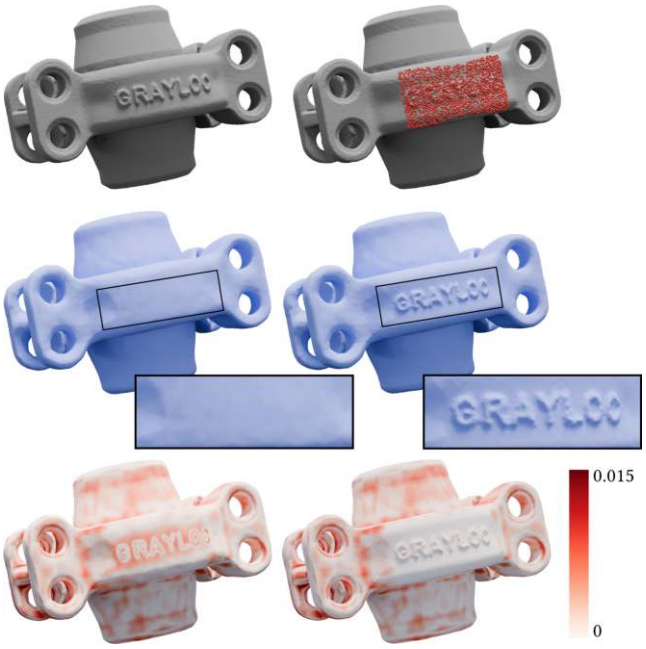


Figure 6: Partial RBF detail field on the *grayloc* model (top left). 3k RBF centers are placed on a small patch of the surface (red dots, top right). Extracting the level set of the neural function only leads to a complete loss of the written details (middle left). On the other hand, taking into account the detail field (middle right) allows for an accurate reconstruction on the area of interest while leaving the rest of the model unchanged. Distance error with relation to the ground truth geometry are shown on the bottom row.

by controlling the sampling of $\mathcal{X}$. This property allows for a partial detail field that correct only a small area of the input geometry. We illustrate this on Figure 6, where the input model is only fitted with a small patch of 3k RBF centers around the area of interest. Computing the detail field allows the detail to be captured while the rest of the model remains unchanged.

4.4. Comparison with pure neural implicit representations

We run comparisons between our method and various neural implicit representations on Figure 7, on an input mesh that exhibits both a complex topology and fine surface details.

Both Siren networks [SMB*20] and *Random Fourier Features* [TSM*20] (RFF) are neural implicit methods developed to minimize the low-frequency bias. Here, they were trained using the setup from Gropp et al. [GYH*20] for a total of 150 epochs. In particular, in addition of losses that penalize non-zero values on the surface $\partial\Omega$ or non alignment with surface normals, these methods optimize the *eikonal* loss:

$$\mathcal{L}_{eik}(f) = \mathbb{E}_{x\in\mathbb{R}^n}\ (||\nabla f(x)|| - 1)^2.$$

On Figure 7a and 7b, we can see that these methods accurately capture the zero level set of the shape as well as some of its details. However, even with an eikonal regularization, their isovalues remain irregular and their gradient often exceed unit norm.

On the other hand, the *hotspot* method [WWY*25] was recently proposed as a neural implicit field that converges to the true signed distance field. In Figure 7c, we observe that the resulting iso surfaces are indeed more regular. However, the method struggles to capture fine details, and the gradient may still exceed unit norm. Note that this result is based on our re-implementation of the method following details from the original article.

In contrast, our method separates the problems of accurately describing the zero level set and having a well-behaved implicit function far from it into its two components. As such, it has a comparable surface reconstruction accuracy as Siren networks and RFF, but does not suffer from irregular iso surfaces.

4.5. Implicit vs explicit displacement field

In another experiment, we compare the results of our method with the *implicit displacement field* (IDF) method from Yifan et al. [YRS22]. Unlike our approach, which displaces the zero level set of f_θ explicitly by adding another field d, this implicit displacement field only acts on the normal direction of the base implicit function f_θ through the formula:

$$S_\Omega(x) \simeq f(x + n(x)d(x)) \tag{4}$$

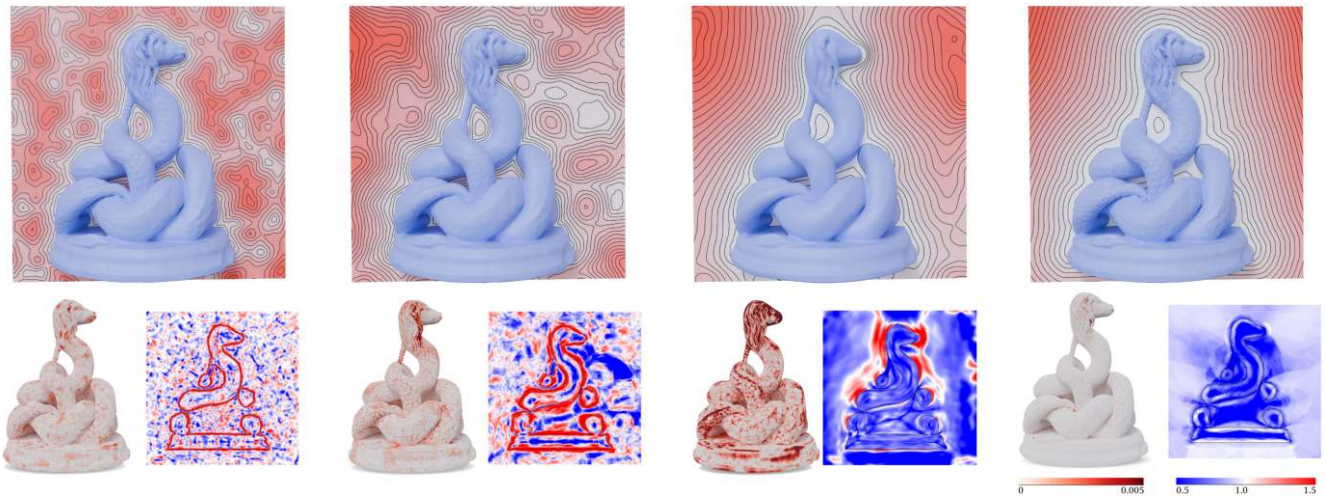
where $n(x) = \frac{\nabla f(x)}{||\nabla f(x)||}$. This approach mimics displacements maps as commonly used in graphics [Coo84].

On Figure 8, we run comparisons with a re-implementation of IDF on the *Armadillo* dataset. We observe that both methods are able to reconstruct the zero level set with perfect visual accuracy, although our method exhibits lower distance error to the ground truth mesh overall. Our RBF based method indeed guarantees that the zero level set passes through some prescribed points onto the surface, while this is only loosely encouraged by a loss term in previous works. Moreover, as both the neural implicit function and the detail field in IDF are Siren networks, no guarantee of compact support size or gradient regularity can be proven. On Figure 9, we can see that both our method and IDF allow a satisfactory reconstruction of the input geometry as the zero level set, but only our method provides a regular level set for the isovalue 0.05.

We also compare with our own RBF detail field fitted to be an implicit displacement instead of an explicit one. Instead of solving for weights such that $d(x_i) = -f_\theta(x_i)$ for all $x_i \in \mathcal{X}$, we project each x_i to their closest point y_i onto the zero level set of f_θ and solve $d(x_i) = ||x_i - y_i||$ for all x_i. The resulting implicit function is then evaluated using Eq. (4). We observe on Figure 8c that the result of this implicit detail field is very close but slightly worse than the result for the explicit field computed from the same centers $\mathcal{X}$. Moreover, an implicit field also requires to compute the gradient of f_θ at the query point for evaluation, which makes queries more computationally intensive.

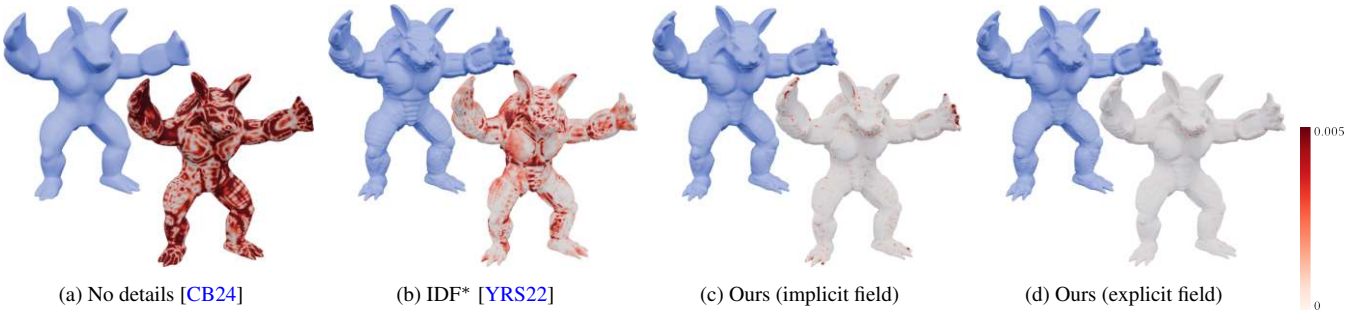
5. Discussion and conclusion

We introduced a detail field based on compactly supported radial basis functions. Combined with a 1-Lipschitz neural distance field, we obtained an implicit representation that is a good approximation of a signed distance field far from the surface while preserving



(a) Siren [SMB*20] (868s) (b) RFF [TSM*20] (718s) (c) Hotspot [WWY*25] (2644s) (d) Ours (1081s + 807s)

Figure 7: Comparisons of various neural implicit representations with our method on the *glykon* model. The top image show the reconstructed zero level set as well as contours of the implicit function along a cutting plane. On the bottom, we display an error map on the level set with relation to ground truth (left) and the norm of the implicit function's gradient along the same cutting plane. (a,b) Both SirenNet and RFF are able to accurately capture the zero level set but exhibit irregular isosurfaces and gradients. (c) *Hotspot* minimizes a loss whose only minimizer is the signed distance field: it provides smoother iso surfaces but cannot capture fine details. (d) Our method uses 300k RBFs and achieves a tradeoff between detail preservation and regularity of the implicit function. Total training and fitting times are provided in captions.



(a) No details [CB24] (b) IDF* [YRS22] (c) Ours (implicit field) (d) Ours (explicit field)

Figure 8: Comparison of displacement fields methods on the *armadillo* model. The reconstructed zero level set (blue) and an error map to the ground truth mesh are shown. (a) The neural function from Coiffier and Béthune [CB24] is unable to capture details but serves as a base for our RBF detail field. (b) Yifan et al. use a Siren network as an implicit displacement field for another base Siren network and achieves high accuracy reconstruction. (c,d) Our RBF field can be used both explicitly and implicitly for high quality surface reconstruction.

details of the zero level set. This approach solves the main drawbacks of both neural-based and RBF-based methods. On the one hand, the detail field helps the neural function to capture fine details and breaks the low-frequency bias observed during training. As its expressivity scales with the number of centers provided, the final representation is also competitively small compared to input meshes with the same level of details. On the other hand, a compactly supported RBF field alone would either be defined only on a narrow band near the surface or require a dense sampling of the whole space to represent a distance field. Having the field only correct another approximate function alleviates this issue, and the compactness of the support provides control on the volume over which the perturbation occurs.

Yet, this tradeoff did not completely solve the issues of both representations. On Figure 10, we show two examples where the result of our method is not satisfactory. On the *Peafowl* dataset (Figure 10a), the detail field is not expressive enough to account for the high curvature of the input mesh, leading to a noisy output. On the *wicker chair* dataset (Figure 10b), the neural function struggles to capture the fine topology of the model. In this uncertain area, the variations of the implicit function remain small, which prevents the detail field from accurately correct the zero level set. Moreover, querying the detail field requires spatial search algorithms that are not trivially parallelized. This adds to the overall computation time, for methods which are already computationally very heavy.

Future works on the topic of implicit detail fields could include improvements to the sampling of RBF centers to better account for detail distribution, as well as theoretical results on the number

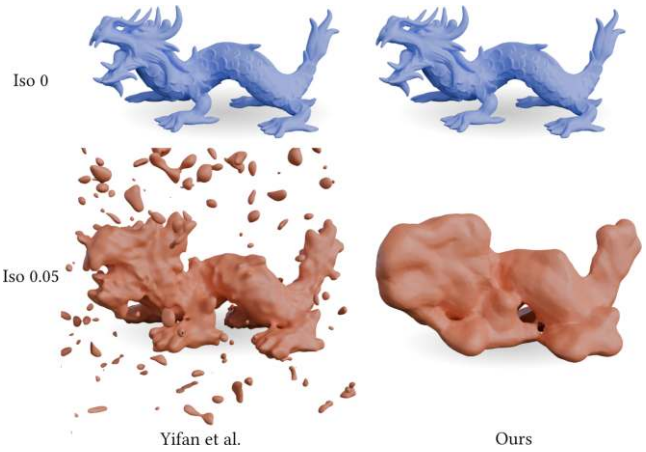
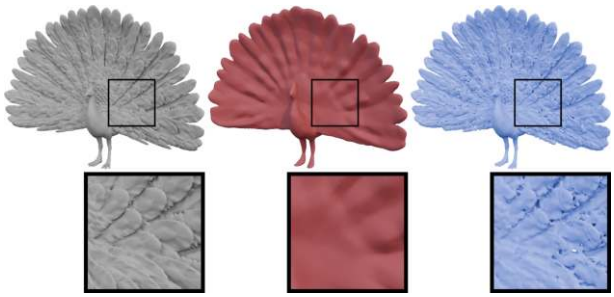


Figure 9: Comparison of the 0 and the 0.05 level set of both our method and the method from Yifan et al. [YRS22] on the *xyzrgb_dragon* dataset. Both methods accurately reconstruct the zero level set, but Yifan et al. rely on a SirenNet that lead to an irregular distance field further away from the surface, even at convergence.



(a) Peafowl (from *YahooJAPAN*), 200k centers (677KB + 4MB)

(b) Wicker Chair (from Thingi10k [ZJ16]), 175k centers (677KB + 3.5MB)

Figure 10: Challenging datasets for our method. Column represent the input mesh (gray), the implicit surface defined by the neural function only (red) and our implicit surface that includes the RBF detail field (blue). On the Peafowl dataset, the detail field is not expressive enough to fully capture the details. On the *Wicker chair* dataset, the topology of the output is almost captured but spurious components remain on the zero level set.

of RBF centers required for a given problem. An algorithm to locally determine a variable support size for each RBF could also be designed, provided guarantees on the positive-definiteness of the RBF linear system could be proven in this more general case. Finally, performance issues could be addressed by exploring efficient closed-form solution for the RBF field instead of solving the system in the spirit of Liu et al. [LWBW16].

Acknowledgements

We would like to thank Georges-Pierre Bonneau, Stefanie Hahmann and Chandradeep Pokhariya for their comments, proofreading and discussion regarding this article.

References

[ABC*01] ALEXA M., BEHR J., COHEN-OR D., FLEISHMAN S., LEVIN D., SILVA C.: Point set surfaces. In *Proceedings Visualization, 2001. VIS '01.* (San Diego, CA, USA, 2001), IEEE, pp. 21–537. doi:10.1109/VISUAL.2001.964489. 2

[AHD*23] ARAUJO A., HAVENS A., DELATTRE B., ALLAUZEN A., HU B.: A Unified Algebraic Perspective on Lipschitz Neural Networks, Oct. 2023. arXiv:2303.03169. 4

[BB97] BLOOMENTHAL J., BAJAJ C.: *Introduction to Implicit Surfaces*. Morgan Kaufmann, Aug. 1997. 2

[BCM99] BEATSON R., CHERRIE J., MOUAT C.: Fast fitting of radial basis functions: Methods based on preconditioned GMRES iteration. *Advances in Computational Mathematics 11*, 2 (Nov. 1999), 253–270. doi:10.1023/A:1018932227617. 2

[BDS*18] BARILL G., DICKSON N. G., SCHMIDT R., LEVIN D. I. W., JACOBSON A.: Fast winding numbers for soups and clouds. *ACM Transactions on Graphics 37*, 4 (Aug. 2018), 1–12. doi:10.1145/3197517.3201337. 4

[BGG*20] BASRI R., GALUN M., GEIFMAN A., JACOBS D., KASTEN Y., KRITCHMAN S.: Frequency Bias in Neural Networks for Input of Non-Uniform Density. In *Proceedings of the 37th International Conference on Machine Learning* (Nov. 2020), PMLR, pp. 685–694. 3

[Bli82] BLINN J. F.: A Generalization of Algebraic Surface Drawing. *ACM Transactions on Graphics 1*, 3 (July 1982), 235–256. doi:10.1145/357306.357310. 2

[Buh00] BUHMANN M. D.: Radial basis functions. *Acta Numerica 9* (Jan. 2000), 1–38. doi:10.1017/S0962492900000015. 2

[CB24] COIFFIER G., BÉTHUNE L.: 1-Lipschitz Neural Distance Fields. *Computer Graphics Forum 43*, 5 (2024), e15128. doi:10.1111/cgf.15128. 1, 2, 3, 4, 6, 8

[CBC*01] CARR J. C., BEATSON R. K., CHERRIE J. B., MITCHELL T. J., FRIGHT W. R., MCCALLUM B. C., EVANS T. R.: Reconstruction and representation of 3D objects with radial basis functions. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, Aug. 2001), SIGGRAPH '01, Association for Computing Machinery, pp. 67–76. doi:10.1145/383259.383266. 2

[Coo84] COOK R. L.: Shade trees. *SIGGRAPH Comput. Graph. 18*, 3 (Jan. 1984), 223–231. doi:10.1145/964965.808602. 7

[DLR86] DYN N., LEVIN D., RIPPA S.: Numerical Procedures for Surface Fitting of Scattered Data by Radial Functions. *SIAM Journal on Scientific and Statistical Computing 7*, 2 (Apr. 1986), 639–659. doi:10.1137/0907043. 2

[DNJ21] DAVIES T., NOWROUZEZAHRAI D., JACOBSON A.: On the Effectiveness of Weight-Encoded Neural Implicit 3D Shapes, Jan. 2021. arXiv:2009.09808, doi:10.48550/arXiv.2009.09808. 2

[Duc77] DUCHON J.: Splines minimizing rotation-invariant semi-norms in Sobolev spaces. In *Constructive Theory of Functions of Several Variables* (Berlin, Heidelberg, 1977), Schempp W., Zeller K., (Eds.), Springer, pp. 85–100. doi:10.1007/BFb0086566. 2

[FP06] FRISKEN S. F., PERRY R. N.: Designing with distance fields. In *ACM SIGGRAPH 2006 Courses* (New York, NY, USA, July 2006), SIGGRAPH '06, Association for Computing Machinery, pp. 60–66. doi:10.1145/1185657.1185675. 2

[GGPP20] GALIN E., GUÉRIN E., PARIS A., PEYTAVIE A.: Segment Tracing Using Local Lipschitz Bounds. *Computer Graphics Forum 39*, 2 (2020), 545–554. doi:10.1111/cgf.13951. 3

[GR87] GREENGARD L., ROKHLIN V.: A fast algorithm for particle simulations. *Journal of Computational Physics 73*, 2 (Dec. 1987), 325–348. doi:10.1016/0021-9991(87)90140-9. 2

[GYH*20] GROPP A., YARIV L., HAIM N., ATZMON M., LIPMAN Y.: Implicit Geometric Regularization for Learning Shapes, July 2020. arXiv:2002.10099. 2, 7

[Har95] HART J.: Sphere Tracing: A Geometric Method for the Antialiased Ray Tracing of Implicit Surfaces. *The Visual Computer 12* (June 1995). doi:10.1007/s003710050084. 1, 2, 3

[HCJ19] HUANG Z., CARR N., JU T.: Variational implicit point set surfaces. *ACM Transactions on Graphics 38*, 4 (Aug. 2019), 1–13. doi:10.1145/3306346.3322994. 1, 2

[KB14] KINGMA D. P., BA J.: Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014). 2

[Kol08] KOLLURI R.: Provably good moving least squares. *ACM Trans. Algorithms 4*, 2 (May 2008), 18:1–18:25. doi:10.1145/1361192.1361195. 2

[LC87] LORENSEN W. E., CLINE H. E.: Marching cubes: A high resolution 3D surface construction algorithm. *ACM SIGGRAPH Computer Graphics 21*, 4 (Aug. 1987), 163–169. doi:10.1145/37402.37422. 6

[Lev04] LEVIN D.: Mesh-Independent Surface Interpolation. In *Geometric Modeling for Scientific Visualization* (Berlin, Heidelberg, 2004), Brunnett G., Hamann B., Müller H., Linsen L., (Eds.), Springer, pp. 37–49. doi:10.1007/978-3-662-07443-5_3. 2

[LMSJ25] LING S., MADAN A., SHARP N., JACOBSON A.: Uniform Sampling of Surfaces by Casting Rays, June 2025. arXiv:2506.05268, doi:10.48550/arXiv.2506.05268. 3

[LWBW16] LIU S., WANG C. C., BRUNNETT G., WANG J.: A closed-form formulation of HRBF-based surface reconstruction by approximate solution. *Computer-Aided Design 78* (Sept. 2016), 147–157. doi:10.1016/j.cad.2016.05.001. 2, 9

[MEM*20] MACKLIN M., ERLEBEN K., MÜLLER M., CHENTANEZ N., JESCHKE S., CORSE Z.: Local Optimization for Robust Signed Distance Field Collision. *Proceedings of the ACM on Computer Graphics and Interactive Techniques 3*, 1 (Apr. 2020), 1–17. doi:10.1145/3384538. 3

[MESK22] MÜLLER T., EVANS A., SCHIED C., KELLER A.: Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics 41*, 4 (July 2022), 1–15. doi:10.1145/3528223.3530127. 3

[MGV11] MACÊDO I., GOIS J. P., VELHO L.: Hermite Radial Basis Functions Implicits. *Computer Graphics Forum 30*, 1 (2011), 27–42. doi:10.1111/j.1467-8659.2010.01785.x. 2

[MHS*25] MCGINNIS J., HÖLZL F. A., SHIT S., BIEDER F., FRIEDRICH P., MÜHLAU M., MENZE B., RUECKERT D., WIESTLER B.: Optimizing Rank for High-Fidelity Implicit Neural Representations, Dec. 2025. arXiv:2512.14366, doi:10.48550/arXiv.2512.14366. 6

[MON*19] MESCHEDER L., OECHSLE M., NIEMEYER M., NOWOZIN S., GEIGER A.: Occupancy Networks: Learning 3D Reconstruction in Function Space, Apr. 2019. arXiv:1812.03828. 2

[Muo] Muon: An optimizer for hidden layers in neural networks | Keller Jordan blog. https://kellerjordan.github.io/posts/muon/. 6

[MYR*05] MORSE B. S., YOO T. S., RHEINGANS P., CHEN D. T., SUBRAMANIAN K. R.: Interpolating implicit surfaces from scattered surface data using compactly supported radial basis functions. In *ACM SIGGRAPH 2005 Courses on - SIGGRAPH '05* (Los Angeles, California, 2005), ACM Press, p. 78. doi:10.1145/1198555.1198645. 2

[OBA*03] OHTAKE Y., BELYAEV A., ALEXA M., TURK G., SEIDEL H.-P.: Multi-level partition of unity implicits. *ACM Trans. Graph. 22*, 3 (July 2003), 463–470. doi:10.1145/882262.882293. 2

[OS88] OSHER S., SETHIAN J. A.: Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations. *Journal of Computational Physics 79*, 1 (Nov. 1988), 12–49. doi:10.1016/0021-9991(88)90002-2. 1

[PFS*19] PARK J. J., FLORENCE P., STRAUB J., NEWCOMBE R., LOVEGROVE S.: DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation, Jan. 2019. arXiv:1901.05103, doi:10.48550/arXiv.1901.05103. 2

[PGM*19] PASZKE A., GROSS S., MASSA F., LERER A., BRADBURY J., CHANAN G., KILLEEN T., LIN Z., GIMELSHEIN N., ANTIGA L., ET AL.: Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems 32* (2019). 5

[PS11] PAN R., SKALA V.: A two-level approach to implicit surface modeling with compactly supported radial basis functions. *Engineering with Computers 27*, 3 (July 2011), 299–307. doi:10.1007/s00366-010-0199-1. 2

[Qui] QUILEZ I.: 3d sdf functions. https://iquilezles.org/articles/distfunctions/. Accessed: 2026-04-06. 1

[RBA*19] RAHAMAN N., BARATIN A., ARPIT D., DRAXLER F., LIN M., HAMPRECHT F., BENGIO Y., COURVILLE A.: On the Spectral Bias of Neural Networks. In *Proceedings of the 36th International Conference on Machine Learning* (May 2019), PMLR, pp. 5301–5310. 3

[Ric73] RICCI A.: A constructive geometry for computer graphics. *The Computer Journal 16*, 2 (Jan. 1973), 157–160. doi:10.1093/comjnl/16.2.157. 1

[Set96] SETHIAN J. A.: A fast marching level set method for monotonically advancing fronts. *Proceedings of the National Academy of Sciences of the United States of America 93*, 4 (Feb. 1996), 1591–1595. doi:10.1073/pnas.93.4.1591. 2

[SJ22] SHARP N., JACOBSON A.: Spelunking the deep: Guaranteed queries on general neural implicit surfaces via range analysis. *ACM Transactions on Graphics 41*, 4 (July 2022), 1–16. doi:10.1145/3528223.3530155. 2, 4, 6

[SMB*20] SITZMANN V., MARTEL J., BERGMAN A., LINDELL D., WETZSTEIN G.: Implicit Neural Representations with Periodic Activation Functions. In *Advances in Neural Information Processing Systems* (2020), vol. 33, Curran Associates, Inc., pp. 7462–7473. 1, 2, 3, 7, 8

[SMG*21] SERRURIER M., MAMALET F., GONZALEZ-SANZ A., BOISSIN T., LOUBES J.-M., DEL BARRIO E.: Achieving robustness in classification using optimal transport with hinge regularization. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (Nashville, TN, USA, June 2021), IEEE, pp. 505–514. doi:10.1109/CVPR46437.2021.00057. 4

[Sta] Stanford 3d scanning repository. http://graphics.stanford.edu/data/3Dscanrep/. Accessed: 2026-04-13. 6

[TLY*21] TAKIKAWA T., LITALIEN J., YIN K., KREIS K., LOOP C., NOWROUZEZAHRAI D., JACOBSON A., MCGUIRE M., FIDLER S.: Neural Geometric Level of Detail: Real-time Rendering with Implicit 3D Shapes. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (Nashville, TN, USA, June 2021), IEEE, pp. 11353–11362. doi:10.1109/CVPR46437.2021.01120. 3

[TO99] TURK G., O'BRIEN J. F.: Shape Transformation Using Variational Implicit Functions. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques - SIGGRAPH '99* (1999), pp. 335–342. arXiv:2303.02937, doi:10.1145/311535.311580. 2

[TRS04] TOBOR I., REUTER P., SCHLICK C.: Multi-scale reconstruction of implicit surfaces with attributes from large unorganized point sets. In *Proceedings Shape Modeling Applications, 2004.* (June 2004), pp. 19–30. doi:10.1109/SMI.2004.1314490. 2

[TSM*20] TANCIK M., SRINIVASAN P., MILDENHALL B., FRIDOVICH-KEIL S., RAGHAVAN N., SINGHAL U., RAMAMOORTHI R., BARRON J., NG R.: Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains. In *Advances in Neural Information Processing Systems* (2020), vol. 33, Curran Associates, Inc., pp. 7537–7547. 2, 3, 7, 8

[VGO*20] VIRTANEN P., GOMMERS R., OLIPHANT T. E., HABERLAND M., REDDY T., COURNAPEAU D., BUROVSKI E., PETERSON P., WECKESSER W., BRIGHT J., VAN DER WALT S. J., BRETT M., WILSON J., MILLMAN K. J., MAYOROV N., NELSON A. R. J., JONES E., KERN R., LARSON E., CAREY C. J., POLAT İ., FENG Y., MOORE E. W., VANDERPLAS J., LAXALDE D., PERKTOLD J., CIMRMAN R., HENRIKSEN I., QUINTERO E. A., HARRIS C. R., ARCHIBALD A. M., RIBEIRO A. H., PEDREGOSA F., VAN MULBREGT P., SCIPY 1.0 CONTRIBUTORS: SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods 17* (2020), 261–272. doi:10.1038/s41592-019-0686-2. 5

[Wen95] WENDLAND H.: Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree. *Advances in computational Mathematics 4*, 1 (1995), 389–396. 2, 4

[WGG99] WYVILL B., GUY A., GALIN E.: Extending the CSG Tree. Warping, Blending and Boolean Operations in an Implicit Surface Modeling System. *Computer Graphics Forum 18*, 2 (1999), 149–158. doi:10.1111/1467-8659.00365. 1

[WHS*] WEIDEMAIER S., HARTWIG F., SASSEN J., CONTI S., BEN-CHEN M., RUMPF M.: SDFs from Unoriented Point Clouds using Neural Variational Heat Distances. *Computer Graphics Forum n/a*, n/a, e70296. doi:10.1111/cgf.70296. 2

[WLYT21] WANG P.-S., LIU Y., YANG Y.-Q., TONG X.: Spline Positional Encoding for Learning 3D Implicit Signed Distance Fields, Oct. 2021. arXiv:2106.01553, doi:10.48550/arXiv.2106.01553. 3

[WWY*25] WANG Z., WANG C., YOSHINO T., TAO S., FU Z., LI T.-M.: HotSpot: Signed Distance Function Optimization with an Asymptotically Sufficient Condition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2025), pp. 1276–1286. 2, 7, 8

[XJ25] XIA J., JU T.: Variational Surface Reconstruction Using Natural Neighbors. *ACM Transactions on Graphics 44*, 4 (Aug. 2025), 1–19. doi:10.1145/3731191. 2

[XTS*22] XIE Y., TAKIKAWA T., SAITO S., LITANY O., YAN S., KHAN N., TOMBARI F., TOMPKIN J., SITZMANN V., SRIDHAR S.: Neural Fields in Visual Computing and Beyond. *Computer Graphics Forum 41*, 2 (2022), 641–676. doi:10.1111/cgf.14505. 2

[XZX19] XU Z.-Q. J., ZHANG Y., XIAO Y.: Training Behavior of Deep Neural Network in Frequency Domain. In *Neural Information Processing* (Cham, 2019), Gedeon T., Wong K. W., Lee M., (Eds.), Springer International Publishing, pp. 264–274. doi:10.1007/978-3-030-36708-4_22. 3

[Yah] Yahoojapan 3d models. https://www.thingiverse.com/YahooJAPAN/designs. Accessed: 2026-04-13. 6

[YRS22] YIFAN W., RAHMANN L., SORKINE-HORNUNG O.: Geometry-Consistent Neural Shape Representation with Implicit Displacement Fields, Feb. 2022. arXiv:2106.05187, doi:10.48550/arXiv.2106.05187. 3, 7, 8, 9

[ZJ16] ZHOU Q., JACOBSON A.: Thingi10k: A dataset of 10,000 3d-printing models. *arXiv preprint arXiv:1605.04797* (2016). 6, 9

[ZRL21] ZHENG J., RAMASINGHE S., LUCEY S.: Rethinking Positional Encoding, Oct. 2021. arXiv:2107.02561, doi:10.48550/arXiv.2107.02561. 3