RBF Your SDF: Radial Basis Function Interpolation of Signed Distance Fields with Implied Tangent Points

YONG CHENG, George Mason University, USA
YOTAM GINGOLD, George Mason University, USA

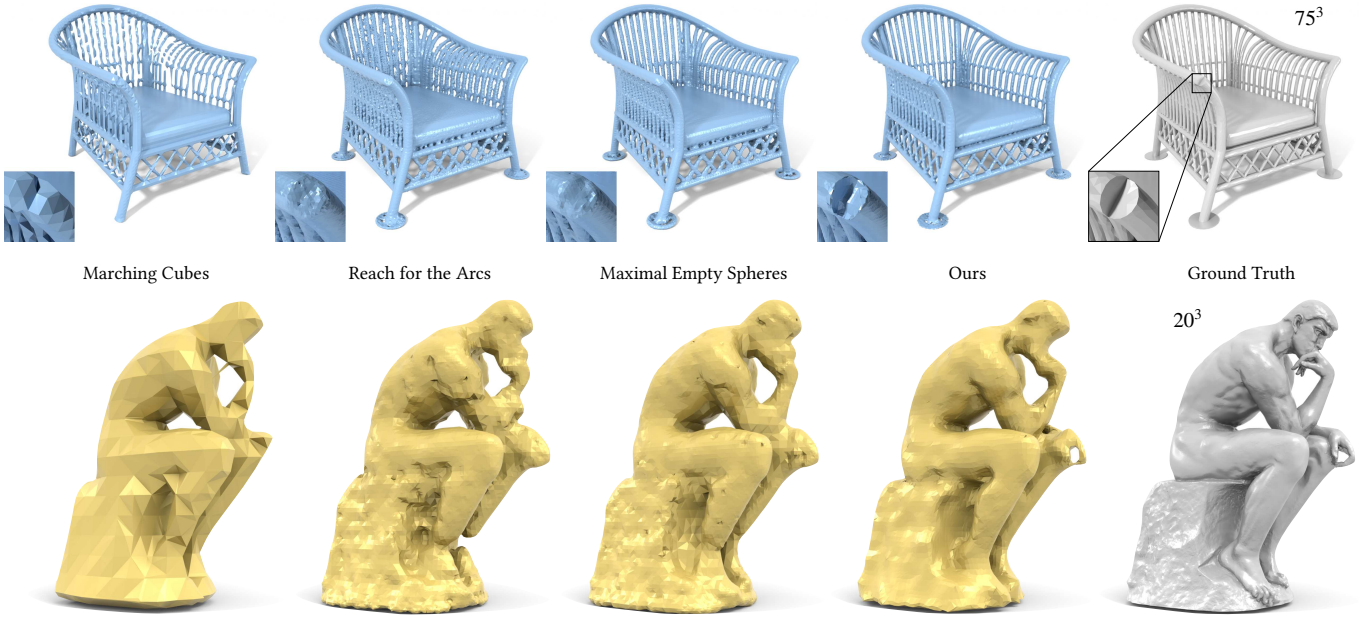


Fig. 1. We marry scattered data interpolation with the tangent sphere observation [Batty 2011; Kobbelt et al. 2001; Sellán et al. 2023] that every Signed Distance Field (SDF) sample implies a point on a sphere tangent to the surface. We propose an iterative procedure to find tangent points consistent with a Radial Basis Function (RBF) interpolating both SDF and tangent point samples. Our algorithm is practical at higher resolutions via a partition-of-unity (PU) formulation and produces lower error at all tested grid sizes.

Signed distance fields (SDFs) are a popular implicit representation of geometry. Converting a discrete set of SDF samples into an explicit surface is a fundamental problem in geometry processing. Traditional reconstruction methods such as marching cubes and dual contouring ignore the geometric information carried by samples far from the surface. Recently, Sellán et al. [2023] and several follow-up works leveraged the tangent-sphere structure of SDFs; every sample implies a point on a sphere tangent to the surface. However, these approaches extract the zero-level set via surface reconstruction, which considers only points and normals on the surface and ignores the remaining samples. We propose an approach that marries the tangent-sphere observation with radial basis function interpolation of all data, the implied surface points and the original data. By detecting spheres with extremely constrained tangent points, a configuration geometrically forced at sharp surface features, we identify and preserve surface corners that surface reconstruction-based methods systematically round. A partition-of-unity decomposition allows our method to scale efficiently to large grid resolutions. Our reconstructions improve both Chamfer and Hausdorff accuracy at every tested resolution.

CCS Concepts: • **Computing methodologies** → **Mesh geometry models**; **Point-based models**; • **Mathematics of computing** → **Interpolation**.

Additional Key Words and Phrases: Implicit surface reconstruction, Signed Distance Fields, Radial Basis Functions, Sharp features

ACM Reference Format:
Yong Cheng and Yotam Gingold. 2026. RBF Your SDF: Radial Basis Function Interpolation of Signed Distance Fields with Implied Tangent Points. In *SIGGRAPH Asia 2026 Conference Papers (SA Conference Papers '26), December 01–04, 2026, Kuala Lumpur, Malaysia.* ACM, New York, NY, USA, 19 pages. https://doi.org/10.1145/3829340.3842207

Authors' Contact Information: Yong Cheng, ycheng27@gmu.edu, George Mason University, USA; Yotam Gingold, ygingold@gmu.edu, George Mason University, USA.

SA Conference Papers '26, Kuala Lumpur, Malaysia

ACM ISBN 979-8-4007-2842-6/2026/12
https://doi.org/10.1145/3829340.3842207

1 Introduction

Reconstructing a surface from a signed distance function (SDF) sampled on a grid or point set is a fundamental problem in geometry processing, with applications in shape modeling, physical simulation, and neural implicit representations. Widely used approaches to zero level-set extraction (e.g., Marching Cubes [Lorensen and

Cline 1987], Marching Tetrahedra [Doi and Koide 1991], and Dual Contouring [Ju et al. 2002]) rely on the local information within cells of a tessellation. Recently, Sellán et al. [2023] observed that every SDF sample $(\mathbf{x}_i, d_i)$ implicitly defines a sphere of radius $|d_i|$ centered at $\mathbf{x}_i$ that is tangent to the surface (and whose interior lies entirely outside or inside the surface).[1] By finding a point on every tangent sphere (and its implied normal), follow-up work [Kohlbrenner and Alexa 2025b; Sellán et al. 2024] achieved state-of-the-art performance by leveraging the well-studied problem of surface reconstruction [Kazhdan and Hoppe 2013] from a set of surface points and normals. However, by relying on surface reconstruction, we claim that these tangent-sphere approaches discard useful information from the original samples and become too sensitive to the quality of their chosen points and normals. Moreover, surface reconstruction based on points and normals struggles with sharp feature points where no one normal describes the local geometry.

At the same time, scattered data interpolation provides a general solution to the problem of reconstructing a continuous function from discrete samples (see §2). Radial basis function (RBF) interpolation, is a classical choice for reconstructing smooth scalar fields from scattered samples. However, although the RBF interpolant agrees with the input SDF at every sample, it does not recover the correct zero level-set. Because no sample lies exactly on the surface, the reconstructed zero level-set is shifted away from the true surface (Figure 2).

Our work marries scattered data interpolation with the tangent sphere observation. We construct an RBF that interpolates each SDF sample $(\mathbf{x}_i, d_i)$ along with a paired zero-valued sample on its tangent sphere $(\mathbf{y}_i, 0)$, where $\|\mathbf{x}_i - \mathbf{y}_i\| = |d_i|$. We propose an iterative procedure to find tangent points consistent with the RBF interpolant. We provide an explicit construction for the region of allowable tangent points on each sample sphere, a formula for its area, and an efficient approximation. We use the cubic kernel, which approximately preserves the Eikonal constraint. To make interpolatory RBF reconstruction practical at high resolutions, we adopt a partition-of-unity (PU) formulation [Franke 1982; Ohtake et al. 2003] that decomposes the domain into overlapping local patches, each solved independently and blended with smooth weights, while remaining strictly interpolatory within each patch. We evaluate our method on a 300-mesh subset of Thingi10K [Zhou and Jacobson 2016] at grid resolutions from 6^3 to 100^3. Our method improves both Chamfer and Hausdorff accuracy at every tested resolution.

2 Related Work

2.1 Surface extraction from SDF samples

Classical, widely-used approaches to explicit surface extraction from sampled implicit functions are Marching Cubes [Lorensen and Cline 1987], Marching Tetrahedra [Doi and Koide 1991; Treece et al. 1999], and the dual variant known as Dual Contouring [Hwang and Sung 2024; Ju et al. 2002]. (See Hwang and Sung [2024] for a recent such work and de Araújo et al. [2015] for a survey that includes alternative approaches.) These methods assume a spatial tessellation and examine the function values in a single cell at a time. They ignore the global geometric information (tangent-sphere property) carried by samples further from the surface. Learning-based variants such as Neural Marching Cubes [Chen and Zhang 2021] and Neural Dual Contouring [Chen et al. 2022] mitigate this by predicting vertex placements from larger local windows and data-driven assumption, but they are more expensive, still only consider local windows, and operate on fixed grids. (They may implicitly make use of the tangent-sphere property via the larger windows.) Our approach first reconstructs a higher quality continuous function (via tangent-spheres and scattered data interpolation) and then relies on Dual Contouring to extract the final mesh.

Sellán et al. [2023] credit Batty [2011] and Kobbelt et al. [2001] with the observation that every SDF sample implies a sphere tangent to (and otherwise disjoint from) the surface. Sellán et al. [2023] exploit this observation in *Reach for the Spheres* (RFTS), an energy-driven mesh flow that improves reconstruction quality but is restricted to the topology of the initial mesh. Their follow-up *Reach for the Arcs* (RFTA) [Sellán et al. 2024] samples tangent points on the intersection-free portion of each sphere and feeds the resulting oriented point cloud to screened Poisson Surface Reconstruction (sPSR) [Kazhdan and Hoppe 2013], removing the topological restriction. The RFTA algorithm optimizes for a set of tangent points that agree with their sPSR. Kohlbrenner and Alexa [2025b] pursue the same overall pipeline deterministically. They use Lie sphere geometry to construct the maximal empty spheres (MES) in the SDF data, take the resulting contact points as oriented surface samples, and likewise feed them to screened Poisson Surface Reconstruction, avoiding the iterative point optimization of RFTA. More recently, the same authors approximate each sample's gradient from a triangulation of the sample locations [Kohlbrenner and Alexa 2026], take $\mathbf{x}_i - d_i\,\mathbf{g}_i$ as a surface point, and discard candidates falling inside another sample's sphere. The gradient estimate is purely local. It degrades near the medial axis, where averaging across the crease yields an invalid direction likely to be filtered away. We instead evaluate our interpolant at the projected location and keep the direction whose projection reaches farthest into the opposite half-space (Sec. 4.2); this selects one of the several contact directions rather than an average of them. Ren et al. [2024]'s work on adaptive refinement formulates point insertion as a Monte Carlo sampling problem but requires SDF queries at arbitrary locations. These works obtain results far superior to classical approaches (e.g., Marching Cubes and Dual Contouring). We are inspired by their approaches. However, instead of relying on surface reconstruction, which relies solely on surface points (and normals), we use both the tangent points (distance = 0) and original points (distance ≠ 0) in a scattered data interpolation algorithm to obtain a higher-quality continuous function.

2.2 Power-diagram-based SDF contouring

Power diagrams have long been used for surface reconstruction. The power crust of Amenta et al. [2001] approximates the medial axis transform of an object by *polar balls* derived from the Voronoi diagram of a dense sample of points *on* its surface, labels the cells of the balls' power diagram as inside or outside, and outputs the power-diagram faces separating the two as the reconstructed surface. Two recent works [Kohlbrenner and Alexa 2025a; Wang et al.

[1] They credit Batty [2011] and Kobbelt et al. [2001] for this observation.

2025] apply the same machinery in the SDF setting, where the balls come directly from the samples. They build on the tangent-sphere observation by constructing power diagrams of weighted SDF samples, where each sample is weighted by its squared signed distance. Kohlbrenner and Alexa [2025a] extract the surface as either a primal marching-tetrahedra contour on the regular triangulation or its dual power contour. They additionally propose a refinement scheme (RCR) that incrementally inserts new sites at locations chosen from the dual power vertices; this scheme requires the ability to query the SDF at arbitrary new locations and is therefore inapplicable when only a fixed set of samples is available—the setting we and RFTS/RFTA address. On fixed samples alone, the authors report that RFTA still outperforms their primal and dual variants. Wang et al. [2025] likewise construct a power diagram, augmented with surface projection points $\mathbf{p}_i - \phi(\mathbf{p}_i)\nabla\phi(\mathbf{p}_i)$ derived from per-sample gradients; their method thus assumes both SDF values *and* gradients as input, in contrast to the SDF-only setting of RFTS, RFTA, and our method.

2.3 Scattered data interpolation and radial basis functions

Scattered data interpolation seeks a function that passes through prescribed values, and possibly derivatives, at arbitrary point locations. Radial basis functions (RBFs) are a classical and well-studied tool for this task [Anjyo et al. 2014; Duchon 1977; Wendland 2004]. Carr et al. [2001] and Turk and O'Brien [2002] pioneered the use of RBF interpolation to recover surfaces from oriented point clouds (e.g., surface points) by fitting an implicit field to point positions augmented with off-surface constraints. To handle inputs without normals, Huang et al. [2019] (VIPSS) and the follow-up *NN-VIPSS* [Xia and Ju 2025] fit a Hermite RBF [Macêdo et al. 2011] while jointly optimizing the unknown sample normals. We apply RBF interpolation to the input SDF samples, which in general do not lie on the surface, together with tangent-sphere points. Because the values are not all zero, we do not need normal information. Similar to NN-VIPSS, we also employ a partition of unity scheme for acceleration. Drake et al. [2022] proposed curl-free RBF kernels in a partition-of-unity formulation to reconstruct surfaces from oriented point clouds at scale. We experimented with this approach, but obtained lower-quality surfaces.

3 Background

Problem setup. Let $\Omega \subset \mathbb{R}^3$ be an unknown closed surface that we wish to reconstruct, and let $D : \mathbb{R}^3 \to \mathbb{R}$ denote its signed distance function, with the convention that D is negative in the interior and positive on the exterior, so that $\Omega = \{\mathbf{x} \in \mathbb{R}^3 : D(\mathbf{x}) = 0\}$. We are given n samples $\{(\mathbf{x}_i, d_i)\}_{i=1}^n$ with $d_i = D(\mathbf{x}_i)$, either on a regular grid or scattered. Our goal is to recover a continuous function $\tilde{D} : \mathbb{R}^3 \to \mathbb{R}$ that *interpolates* the samples, $\tilde{D}(\mathbf{x}_i) = d_i$ for all i, and whose zero level set $\{\tilde{D} = 0\}$ closely approximates Ω.

3.1 Tangent-sphere structure of SDF samples

Each sample carries more information than its scalar value alone. As observed by prior work Batty [2011]; Kobbelt et al. [2001]; Sellán et al. [2023], the unsigned value $|d_i|$ defines a sphere $S_i := \{\mathbf{x} \in \mathbb{R}^3 : \|\mathbf{x} - \mathbf{x}_i\| = |d_i|\}$ whose enclosed ball lies entirely on one side of Ω (interior if $d_i < 0$, exterior otherwise), and which is tangent to Ω at the closest surface point

$$\mathbf{a}_i \;=\; \mathbf{x}_i - d_i\,\nabla D(\mathbf{x}_i) \;\in\; S_i, \qquad D(\mathbf{a}_i) = 0. \tag{1}$$

Each input sample thus implies an additional zero-valued constraint located on S_i. The position $\mathbf{a}_i$, however, is not directly available, since it depends on the unknown gradient $\nabla D(\mathbf{x}_i)$; recovering an estimate of $\mathbf{a}_i$ is the core algorithmic step described in Section 4.2. Every tangent sphere surface has at least one point not contained in the interior of other spheres; we call the set of such points the *exposed region.*[2]

3.2 Radial basis function interpolation

For data $\{(\mathbf{x}_i, d_i)\}_{i=1}^n \subset \mathbb{R}^3 \times \mathbb{R}$, a radial basis function (RBF) interpolant takes the form [Wendland 2004]

$$f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i\, \phi(\|\mathbf{x} - \mathbf{x}_i\|) + p(\mathbf{x}), \tag{2}$$

where $\phi : \mathbb{R}_{\geq 0} \to \mathbb{R}$ is a fixed radial kernel and p a low-order polynomial. We use the *cubic polyharmonic* kernel $\phi(r) = r^3$ together with a degree-1 polynomial $p(\mathbf{x}) = \mathbf{c}^\top\mathbf{x}+c_3$. The coefficients $\boldsymbol{\alpha} \in \mathbb{R}^n$ and $(\mathbf{c}, c_3) \in \mathbb{R}^4$ are determined by the n interpolation conditions $f(\mathbf{x}_i) = d_i$ together with the four moment conditions $\sum_i \alpha_i\mathbf{x}_i = \mathbf{0}$ and $\sum_i \alpha_i = 0$, yielding the symmetric saddle-point system

$$\begin{pmatrix} A & P \\ P^\top & 0 \end{pmatrix} \begin{pmatrix} \boldsymbol{\alpha} \\ \mathbf{c}_{0:3} \end{pmatrix} = \begin{pmatrix} \mathbf{d} \\ \mathbf{0} \end{pmatrix}, \qquad A_{ij} = \|\mathbf{x}_i - \mathbf{x}_j\|^3, \tag{3}$$

with $P \in \mathbb{R}^{n\times 4}$ the polynomial design matrix whose i-th row is $(\mathbf{x}_i^\top, 1)$, and $\mathbf{c}_{0:3} = (c_0, c_1, c_2, c_3)^\top$. The cubic kernel is conditionally positive definite of order 2 ([Wendland 2004], Corollary 8.18), so a linear polynomial augment is the minimum required for unique solvability.

We selected the cubic (3D triharmonic) kernel $\phi(r) = r^3$, after empirically comparing it against r, $r^2 \log r$, the Gaussian kernel, and Wendland's compactly supported C^2 kernel. The cubic kernel best preserves the unit-norm gradient property of a true SDF between data points and in the far field (Appendix Figure 18). This can be explained in 1D, where evaluating the cubic kernel at x "to the right of" the span of data x_i lets us expand the cubic kernel without absolute value, i.e.,

$$\sum_i \alpha_i\,(x - x_i)^3 = x^3 \sum_i \alpha_i - 3x^2 \sum_i \alpha_i x_i + 3x \sum_i \alpha_i x_i^2 - \sum_i \alpha_i x_i^3.$$

The zero-moment conditions eliminate the cubic and quadratic terms $x^3 \sum \alpha_i - 3x^2 \sum \alpha_i x_i$, leaving behind a linear function.

We do not regularize the system. Replacing the kernel block A in (3) with $A+\lambda I$ (Tikhonov) does help on the occasional ill-conditioned input in Drake et al. [2022], but on most inputs it smooths out the details exact interpolation aims to preserve. We instead eliminate the near-coincident constraints that motivate it [Wendland 2004] upstream by de-duplication (§4.2).

[2] Sellán et al. [2024] call this the *feasibility arc.* We avoid this term to prevent confusion with the arcs which bound the exposed region.

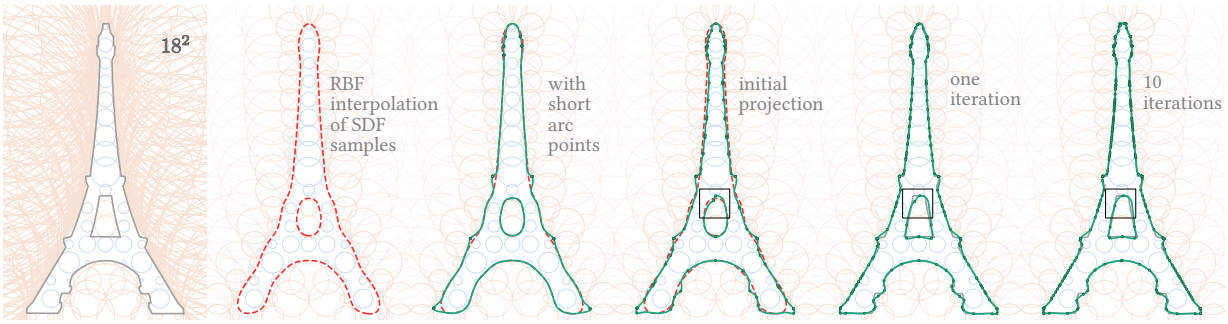


Fig. 2. Progressive 2D reconstruction of the EIFFEL outline from 18^2 SDF samples. From left to right: (i) ground truth; (ii) the zero level set of the interpolant of the SDF samples alone, which floats off the true surface and rounds off every sharp feature; (iii) after adding short-arc tangent points (Sec. 4.1), which restore the corners; (iv) after one outer iteration, with samples whose spheres have no intersecting neighbor deferred from projection; (v) the next iteration projects every remaining sample, but those failing the feasibility test are rejected; (vi) after 10 iterations, additional projections become feasible and the level set has converged to the true surface. Insets in the last 3 panels show how iterations smooth the level set near tangencies. Sample circles are faded by radius for clarity in panels (ii)–(vi). Tangent projections are drawn as light-green dots. For comparison, panels (iii) and (iv) overlay the level set (green) on (ii) (red dashed); (v) and (vi) overlay it on the ground truth (gray).

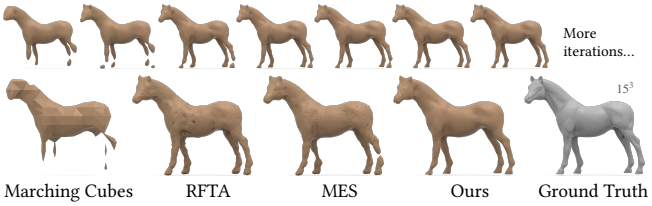


Fig. 3. Reconstruction of the HORSE mesh at 15^3 sampling. *Top row:* Our zero level set across iterations, starting from the interpolant of the SDF samples alone on the left and refined by successive iterative-projection steps to the right. *Bottom row:* Output versus ground truth. Our output mesh is closer to the ground truth geometrically and topologically compared to alternatives.

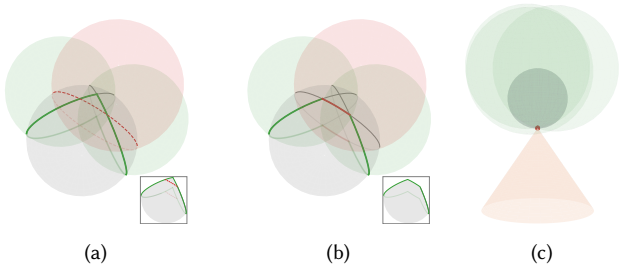


Fig. 4. Per-sphere incremental update of the exposed region boundary on a sample sphere (gray). (a) and (b) show the same update in 2D (top) and 3D (bottom). (a) Existing caps and their currently exposed arcs (green); the incoming cap is shown as a dashed red circle. The exposed region is shown inset. (b) After the new cap is committed, existing green arcs are clipped where they enter the new cap and the new cap's surviving arcs (red) are added to the boundary. The exposed region (inset) shrinks accordingly. (c) At a sharp corner, neighboring sample spheres bracket the arcs from many directions and the exposed region collapses to the corner point (red).

4 Algorithm

Given SDF samples $\{(\mathbf{x}_i, d_i)\}_{i=1}^n$, our algorithm (Figures 2 and 3) recovers a smooth (except at sharp features) and accurate interpolatory $\tilde{D}$, followed by mesh extraction. The goal of our algorithm is to find a paired tangent point $\mathbf{y}_i$ for each sample $(\mathbf{x}_i, d_i)$. The core of our algorithm is straightforward:

while not converged **do**
  $\mathbf{y}_i \leftarrow$ Estimate tangent points for samples $\mathbf{x}_i$
  $\tilde{D} \leftarrow \mathrm{RBF}(\{(\mathbf{x}_i, d_i)\} \cup \{\mathbf{y}_i, 0\})$
end while
return IsosurfaceExtract$(\tilde{D}, 0)$

See Appendix A for in-depth pseudocode and a description of our partition of unity scheme.

The first iteration extracts tangent points for spheres whose exposed regions have collapsed to (nearly) a point (Section 4.1). We then alternate between computing an interpolatory partition-of-unity RBF $\tilde{D}$ over the samples $(\mathbf{x}_i, d_i)$ and current tangent points $\mathbf{y}_i$, and updating each sample sphere's tangent direction by an S^2 search using the current $\tilde{D}$ as a guide (Section 4.2). Both steps of the algorithm are parallelizable, the first step since tangent points can be estimated independently, and the second step because of partition of unity. Finally, because $\tilde{D}$ is defined everywhere, one can sample it on a new grid at any resolution and apply any standard contouring algorithm to extract the mesh. We extract the surface mesh from the zero level set of $\tilde{D}$ using Dual Contouring [Ju et al. 2002] at a per-axis extraction resolution of $\min\big(\max\big(\lceil 4(\sqrt[3]{n}-1)+1\rceil,\ 64\big),\ 512\big)$. The bounds keep the output usable for very sparse inputs and cap the extraction cost for very dense ones.

4.1 Geometrically Determined Tangent Points

In two cases, the tangent point is completely determined by geometry alone. (A) At a sharp surface feature, a sample sphere can be squeezed by many neighbors until its exposed region collapses to (nearly) a single point (Figure 4c). This happens most reliably at corners (where the sphere is bracketed from all directions) but also along sharp edges when the sampling along the edge is dense enough to bracket a smaller sphere. (B) When two spheres are *inner tangent*, the smaller one is strictly inside the larger one except at the

tangent point while $\|\mathbf{x}_i - \mathbf{x}_j\| = |d_i - d_j|$ (Figure 5c). Both of these situations can be detected by computing the size of exposed regions. We emit such completely determined tangent points as zero-valued interpolation constraints, bypassing the iterative projection step (Section 4.2) required for general sample spheres.

Small exposed regions. To compute the size of an exposed region, we first compute the set of arcs on S_i that bound it. Given such arcs, we can precisely compute the area of the exposed region via Gauss-Bonnet,[3] though we observe that perimeter collapse is sufficient to find tiny regions, requires no connectivity of arcs, and is perhaps even preferable since slivers may have small area but leave substantial ambiguity as to the location of the tangent point. Exposed regions whose total arc length falls below ϵ_{degen} are treated as collapsed to a point.

Given a sample sphere S_i with center $\mathbf{c}_i$ and radius r_i, each intersecting neighbor S_j (let $D_{ij} = \|\mathbf{c}_j - \mathbf{c}_i\|$) carves a spherical cap C_j on S_i, bounded by a circle that lies in the cutting plane $\mathbf{n}_{ij} \cdot \mathbf{x} = d_{ij}$ with

$$\mathbf{n}_{ij} = \frac{\mathbf{c}_j - \mathbf{c}_i}{D_{ij}}, \qquad h_{ij} = \frac{D_{ij}^2 + r_i^2 - r_j^2}{2\,D_{ij}}, \qquad d_{ij} = \mathbf{n}_{ij} \cdot \mathbf{c}_i + h_{ij}. \quad (4)$$

The circle bounding C_j has radius $\rho_j = \sqrt{r_i^2 - h_{ij}^2}$.

We process neighbors of S_i one at a time, maintaining the running list of boundary arcs $\{a_\ell\}_\ell$, each stored as an angular interval $[t_\ell^{\text{start}}, t_\ell^{\text{end}})$ on the circle of its host cap $C_{c(\ell)}$. When a new cap C_{new} is added, the update is bidirectional (Figure 4):

(1) *Clip existing arcs by* C_{new}. For every arc a_ℓ, intersect its host circle with C_{new}'s cutting plane, then drop the angular portion of a_ℓ that lies inside C_{new}.
(2) *Compute* C_{new}*'s own arcs against the preceding caps.* Starting from the full circle $[0, 2\pi]$ on C_{new}, intersect with the plane of each previously processed cap. The surviving intervals are C_{new}'s contribution to the boundary.

Pseudocode for this process appears in Appendix A.

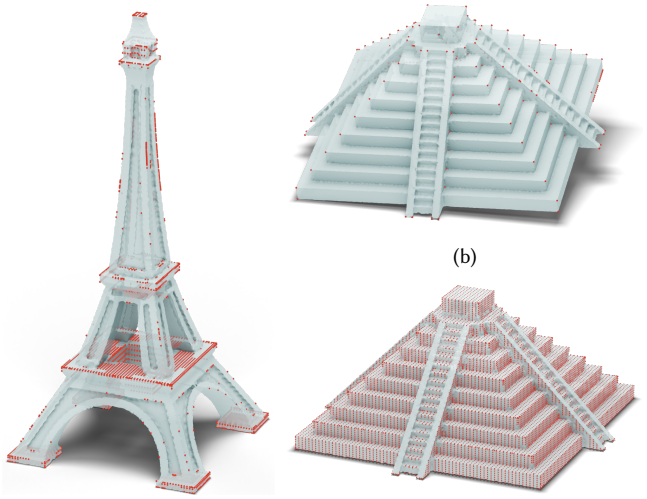
Tangent points at near degeneracies. The interval intersection in step (2) tolerates a small "negative" arc. A preceding cap's plane may cut the C_{new} circle just outside its arc interval. If this would create a short negative interval ($< \epsilon_{\text{tan}} = 10^{-4}$ rad), we emit a *zero-length* arc at the midpoint of the negative arc instead of discarding it.

If, after processing all neighbors, the total arc length on S_i falls below a small threshold ϵ_{degen},

$$\sum_\ell \rho_{c(\ell)} \left(t_\ell^{\text{end}} - t_\ell^{\text{start}}\right) \;<\; \epsilon_{\text{degen}}, \quad (5)$$

the exposed region has shrunk to (essentially) a point (Figures 4c and 5). We emit each arc's midpoint directly as a tangent-point candidate.

[3]Gauss-Bonnet states $\int_M K\,dA + \int_{\partial M} \kappa_g\,ds = 2\pi\chi(M)$. Because M is a subset of a sphere with radius r, $\int_M K\,dA = \frac{A}{r^2}$, where A is the unknown area of M, and $\chi(M) = C - L$, where C are the number of exposed region connected components and L the number of holes in them. Rearranging terms, the formula is $A = r^2(2\pi(C-L) - \int_{\partial M} \kappa_g\,ds)$. The integrated geodesic curvature along a boundary loop of piecewise circular arcs on the sphere is $\sum \frac{h_i}{r}\phi_i + \sum \theta_{i\to j}$, where h_i is the distance of circular arc i's plane from the sphere center, ϕ_i the arc's radians (i.e., 2π for a complete circular arc), and $\theta_{i\to j}$ is the exterior turning angle (signed angle between tangents on the sphere, positive when turning towards the interior) where arc i meets arc j.



(a) (b) (c)

Fig. 5. Small exposed regions' tangent points (red dots) at $n = 100^3$. A reconstructed (a) Eiffel and (b) Pyramid surface. On an axis-aligned Pyramid (c), collinear spheres project to a common point on flat surfaces as well.

A sphere may have several disjoint small exposed regions. We keep the arc midpoint with the smallest $|\tilde{D}_0|$, where $\tilde{D}_0$ is the interpolant fit to the SDF samples alone, and only if $|\tilde{D}_0| < \epsilon_{\text{val}} = 0.1$. If any candidate is kept, it becomes a committed tangent point and its sphere skips the iterative projection step (Sec. 4.2).

We use $\epsilon_{\text{degen}} = 10^{-5}$ for meshes normalized to a unit bounding box. Experimentally, the algorithm is insensitive to this choice across $\epsilon_{\text{degen}} \in [10^{-8}, 10^{-4}]$ (Appendix D.4).

Computational Complexity. For a sample sphere with k intersecting neighbors, clipping each cap circle against the remaining $k-1$ caps costs $O(k)$, so the per-sphere work is $O(k^2)$. Aggregating across n samples gives $O(n\,k^2)$. Naively, k scales linearly with n as the sampling density grows. Refining the sampling inserts new spheres between any two that already overlap, and those new spheres must overlap both, so each sphere's intersection count grows *linearly* with the total sample count n. Supplementary Figure 16(ii) shows both the median and the maximum neighbor count scaling linearly with n. In the worst case, $k = n$ and the complexity is $O(n^3)$.

We avoid this scaling, as only an $O(1)$ (non-pathological) subset of neighbors actually contribute arcs to its exposed region, reducing the total cost to $O(n)$ plus the $O(n \log n)$ (non-pathological) cost of finding appropriate neighbors via a *regular triangulation* (RT). Two spheres share exposed region boundary arcs only if their samples are neighbors in the RT of all samples [Edelsbrunner 1995; Kohlbrenner and Alexa 2025a, 2026]. A sample with a small exposed region, however, is numerically fragile and may be omitted from the RT. In that case, its appropriate neighbor set is the vertices of the RT simplex containing it.

4.2 Iterative Projection

Direction refinement. Tangent spheres without small exposed regions determine their tangent points iteratively. A natural approach to locate where each sample's tangent sphere touches the surface

is to extract a mesh from the current SDF and perform closest-point queries against it [Sellán et al. 2024]. This, however, ties the projection accuracy to mesh resolution. At low resolution, many projections collapse onto the same vertex, while at high resolution the cost is dominated by isosurface extraction. We instead refine the tangent direction directly on S^2 in a mesh-free manner.

Specifically, for each sample $\mathbf{x}_i$ with $d_i = \tilde{D}(\mathbf{x}_i)$, we seek the unit direction $\mathbf{g}_i$ for which the projected point $\mathbf{y}_i = \mathbf{x}_i - d_i\,\mathbf{g}_i$ is tangent to the surface, i.e., $\mathbf{g}_i$ aligns with the outward unit normal $\hat{\mathbf{n}}(\mathbf{y}) := \nabla\tilde{D}(\mathbf{y})/\|\nabla\tilde{D}(\mathbf{y})\|$ at $\mathbf{y}_i$. In other words, among all points $\mathbf{y}$ on the sphere $\|\mathbf{y} - \mathbf{x}\| = |d|$, we seek the one that reaches farthest past the surface into the region of opposite sign, so we minimize

$$f(\mathbf{g}) \;=\; \tilde{D}(\mathbf{x} - d\,\mathbf{g}) \,/\, d \tag{6}$$

over $\mathbf{g} \in S^2$. Dividing by the signed distance d both orients the objective for samples of either sign and cancels the chain rule, so the Euclidean gradient is exactly $\nabla f = -\nabla\tilde{D}(\mathbf{y})$. At a stationary direction $\mathbf{g}^*$, the tangential part of $\nabla\tilde{D}(\mathbf{y})$ vanishes, i.e., $\mathbf{g}^* \parallel \nabla\tilde{D}(\mathbf{y}^*)$ and $\mathbf{g}^*$ agrees with $\hat{\mathbf{n}}$ as required.

We minimize f with BFGS [Nocedal and Wright 2006], normalizing after each step to project onto the sphere. We terminate once the tangential gradient falls below 10^{-4} ($\approx 0.006°$) or after five evaluations, since the next outer iteration re-solves the samples against an updated $\tilde{D}$ anyway. Tangential steps are capped at 0.2 initially and at 1 thereafter, rotating $\mathbf{g}$ by at most 11° and 45°.

Initialization. Small-exposed-region samples handled by the short-arc step (Section 4.1) take no part in the iterative refinement at any iteration. For the remaining samples, every iteration warm-starts from the previous iteration's converged $\mathbf{g}$. The very first $\mathbf{g}^{(0)}$ comes from a Fibonacci-lattice search: we sample 64 near-uniform directions $\mathbf{u}_j$ on S^2 and pick the one minimizing $\mathrm{sgn}(d_i)\,\tilde{D}(\mathbf{x}_i - d_i\,\mathbf{u}_j)$.

As a heuristic, we also defer the initialization for any sample whose sphere intersects no neighboring sphere, so that the high-confidence tangent points are committed first. Once these constraints are folded into the interpolant, the next outer iteration's $\tilde{D}$ is more accurate, and the Fibonacci-lattice search lands on a better warm-start for these deferred samples. We mark their directions as empty in the current iteration and re-initialize them by the same lattice search against the updated $\tilde{D}$ in the next iteration.

Filtering invalid projections. Projected points that fall outside the exposed region of their sphere are culled; the alternatives of accepting them as-is or projecting them onto the exposed region boundary underperform empirically. The RT neighbors found for the short-arc phase are sufficient to filter infeasible projections. Because each gradient update can shift the projected point, a sphere whose projection was feasible in the previous iteration may become infeasible after a new $\mathbf{g}^{(k+1)}$ is taken. Rather than accept the regression, we reject any update that turns a previously feasible projection infeasible and retain the prior $\mathbf{g}$ instead, ensuring the set of feasible projections grows monotonically across iterations.

Clamping. Sharp edges, like corners, also yield small exposed regions, but theirs are *skinny* rather than point-like: too elongated for the short-arc test (Sec. 4.1) to collapse them, yet narrow enough that the feasibility test above rejects projections that miss them only slightly. Losing these projections is costly, since the dense band of tangent points along an edge is what lets the interpolant reproduce it sharply. We therefore clamp such a projection to the nearest point on its exposed region boundary after 7 outer iterations, rather than culling it, whenever both its distance to that region and the region's total arc length fall below $\epsilon_{\text{clamp}} = 0.1$. Restricting the clamp to these near-degenerate cases avoids the degradation observed when boundary projection is applied to every infeasible projection.

De-duplication. Clusters of near-coincident projections may occur, either emitted by the short-arc step or produced by independent samples' tangent points converging to the same surface point. This increases per-patch sample counts and decreases conditioning of the RBF system [Wendland 2004], without contributing new geometric information. We de-duplicate points with a fixed radius of 5×10^{-4}.

Partition-of-unity weighting. See Appendix B for details of our partition of unity scheme.

5 Experiments

In every table, the best score in each column and metric is in bold and the second best is underlined.

5.1 Accuracy

Dataset. We evaluated our method on a subset of the Thingi10K dataset [Zhou and Jacobson 2016], which provides 10,000 3D-printing models curated from Thingiverse. To ensure well-defined signed distance functions, we restricted the evaluation set to meshes that are manifold, closed, solid, have no self-intersections, and contain between 500 and 200,000 vertices. The lower bound excludes degenerate inputs; the upper bound caps the memory footprint of dense SDF sampling. From the 5,249 models meeting these criteria, we deterministically selected the first 300 by ascending file ID. We did not reorient the meshes to align sharp features with the sampling grid, although most of the selected models happen to be modeled axis-aligned. All models are normalized such that the bounding box is centered at the origin with the longest edge equal to one. We then sampled its signed distance function on a uniform $\sqrt[3]{n} \times \sqrt[3]{n} \times \sqrt[3]{n}$ grid within an inflated bounding box (axis-aligned padding of 0.1). We used libigl [Jacobson et al. 2018] to compute unsigned distances and generalized winding numbers [Jacobson et al. 2013] for signs. We swept $\sqrt[3]{n} \in \{6, 10, 20, 30, 40, 50, 60, 80, 100\}$ to characterize each method across resolutions. A gallery of outputs can be seen in Figure 11 and in the supplemental materials.

Baselines and failure modes. We compared our approach to MES (via the authors' CGAL *Maximal Empty Spheres* module), RFTA (via the authors' implementation), and Marching Cubes (via scikit-image [van der Walt et al. 2014]). For RFTA and MES, we evaluated three screening weights $\sigma = 1, 10, 100$. At very low grid resolutions, the samples may be too sparse for MES to certify any valid empty spheres, and no contact points are produced. The failure count per grid size is: 6^3: 60, 10^3: 11, 20^3: 1. RFTA always produced a mesh at $\sigma = 10$, but Poisson Surface Reconstruction occasionally produces stray isosurface components well beyond the sampled region. We remove components with a majority of vertices outside the input

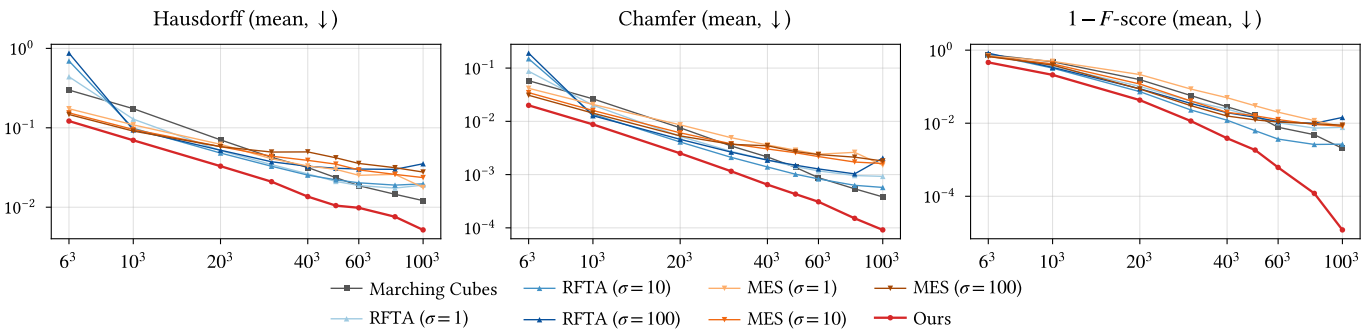


Fig. 6. Accuracy versus number of samples n on 300 Thingi10K meshes (mean over successful cases). Lower is better in all three panels. RFTA and MES are shown for three screening weights $\sigma \in \{1, 10, 100\}$. Our method dominates every baseline variant at every resolution, and the gap widens with n: at n=100^3 the next-best method (MC) has 4.2× our mean Chamfer, 2.3× our mean Hausdorff, and more than 100× our $1-F$ values.

samples' bounding box, falling back to the largest connected component when no component satisfies this criterion. Marching Cubes fails when SDF samples are all positive or all negative. The failure count per grid size is: 6^3: 48, 10^3: 7, 20^3: 1. Our approach succeeded for all inputs.

Metrics. We report the Hausdorff, Chamfer, and F-score (threshold 1% of the unit cube diagonal) on successful cases. Figure 6 and Appendix Table 5 show that our method achieves strictly lower Hausdorff and Chamfer error and strictly higher F-score than every baseline variant across the entire resolution sweep; the gap widens with number of samples n. At n=100^3 the best RFTA or MES mean Chamfer (RFTA-σ=10, 5.74×10^{-4}) is 6.3× ours (9.15×10^{-5}), the best Hausdorff (MES-σ=1) is 3.4× ours (5.18×10^{-3}), and our mean F-score is 0.99999, with every baseline's error rate $1-F$ more than 100× ours. At n=10^3 the best baseline's mean Chamfer (RFTA-σ=100, 1.27×10^{-2}) is 1.5× ours (8.76×10^{-3}) and F-score is 0.79 versus 0.68; at n=6^3 the best baseline's Chamfer is again 1.5× ours (3.09×10^{-2} vs. 2.00×10^{-2}, MES-σ=100) and F-score is 0.54 versus 0.33. Our pipeline also returns a mesh on all 300 inputs at every resolution tested, while MC and MES respectively fail on 48 and 60 inputs at $n = 6^3$.

Ablations. See the supplemental materials for ablation experiments evaluating partition of unity, short-arc seeding, the choice of RBF kernel, and the tangent-point collapse threshold.

Sharp features. To assess sharp-feature reconstruction, we additionally evaluated the *Edge Chamfer Distance* [Chen and Zhang 2021] on the 50 meshes with the most sharp features (using Chen and Zhang [2021]'s sharpness criteria). Our method had the lowest Edge Chamfer Distance at four of five tested resolutions (Table 1).

We attribute our accuracy to several factors. Estimated tangent points are hard constraints rather than smoothed as in the sPSR algorithm used by MES and RFTA. Unlike MES and RFTA, our RBF interpolation does not explicitly consider normals, instead relying on off-surface samples. Indeed, we experimented with Hermite RBF interpolation [Macêdo et al. 2011], but obtained worse performance since gradients are not well-defined at sharp features. In MES and RFTA, inaccurate tangent points imply inaccurate normals.

Table 1. Mean Hausdorff, Chamfer, and edge Chamfer distance [Chen and Zhang 2021] over the 50-mesh sharp-feature Thingi10K subset (Sec. 5.1).

Method	Hausdorff (×10^{-3})	Chamfer (×10^{-4})	Edge Chamfer (×10^{-4})
$n = 30^3$			
RFTA	31.8	31.7	**14.6**
MES	49.1	59.8	26.1
MC	56.0	49.5	116
Ours w/o clamping	30.6	22.6	18.2
Ours	**30.2**	**22.0**	16.4
$n = 60^3$			
RFTA	23.0	17.6	13.0
MES	34.7	38.6	12.6
MC	26.4	13.7	90.7
Ours w/o clamping	**17.8**	7.26	11.7
Ours	17.9	**6.97**	**9.42**
$n = 100^3$			
RFTA	21.9	14.6	22.6
MES	44.1	44.5	16.1
MC	20.1	7.20	34.6
Ours w/o clamping	5.98	2.71	5.51
Ours	**5.92**	**2.52**	**3.83**
$n = 150^3$			
RFTA	31.5	39.8	33.4
MES	39.4	41.5	78.0
MC	15.4	3.23	22.3
Ours w/o clamping	**3.97**	1.07	2.93
Ours	4.05	**0.968**	**2.13**
$n = 200^3$			
RFTA	19.8	13.5	26.6
MES	56.3	64.9	202
MC	13.1	1.89	13.2
Ours w/o clamping	3.36	0.601	2.16
Ours	**3.29**	**0.534**	**1.79**

Is our improved performance due to better tangent points or RBF interpolation? To understand our algorithm's improved performance, we evaluated different combinations of tangent points (ours, RFTA, MES, and ground truth) and surface reconstruction methods (sPSR and RBF) on the first 50 models of our Thingi10K subset. We used an sPSR screening value of 10 and filtered spurious surfaces from RFTA and MES output. Results can be seen in Figure 7; the underlying numbers are in Appendix Tables 7 and 8.

For three sources of tangent points (ours, MES, ground truth), RBF reconstruction substantially (Chamfer) or somewhat (Hausdorff) improves error versus sPSR. For RFTA, the effect is reversed; RBF harms error a little (Chamfer) or a lot (Hausdorff). The Chamfer

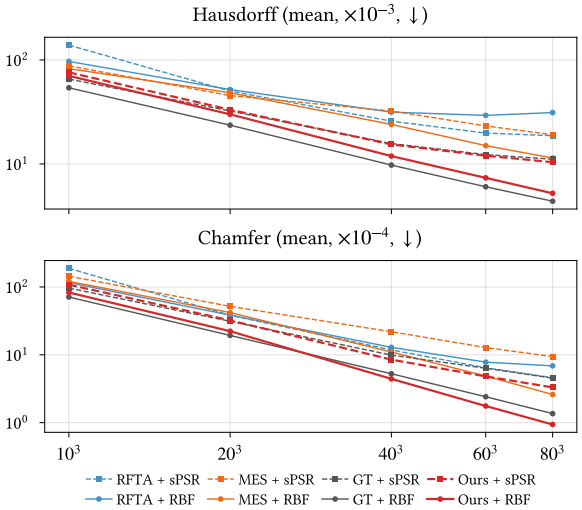


Fig. 7. Every combination of tangent-point source (color) and reconstruction method (solid with circles: RBF; dashed with squares: sPSR) on the first 50 models of our Thingi10K subset. GT are the ground truth closest surface points for each sample.

Table 2. Robustness to irregular sampling. Mean Hausdorff / Chamfer distance (both $\times 10^{-3}$, ↓) over 10 shapes with 50^3 samples placed on a regular grid versus at scattered positions.

Method	Grid	Scattered
RFTA	18.5 / 0.850	21.0 / 0.883
MES	16.8 / 1.08	19.2 / 0.837
MC	26.7 / 1.36	–
Ours	**10.3** / **0.349**	**10.2** / **0.327**

distance is similar for RBF reconstruction of our tangent points versus ground truth tangent points. This suggests that, for most methods, RBF reconstruction is superior to sPSR.

For either reconstruction method, our tangent points outperform MES, RFTA, and often ground truth.[4] Thus our method's improved performance can be attributed both to our improved tangent points and the RBF reconstruction.

Scattered data. We obtain similar Chamfer, Hausdorff, and F1 scores with n points distributed randomly rather than arranged in a grid (Table 2).

Truncated values. When samples farther than an absolute threshold b are deleted, our approach degrades far more gracefully than RFTA and MES (Table 3). On the Rossignol model at $n = 50^3$, our error is essentially unchanged down to $b = 0.05$ (6.80 versus 6.85 Hausdorff, $\times 10^{-3}$), and it still recovers a reasonable surface at $b = 0.002$, whereas at $b = 0.1$ the Chamfer error of RFTA and MES has already grown 189× and 173×. This may be because the cubic kernel extrapolates well (Section 3.2). Marching Cubes clamps out-of-range samples rather than deleting them, so it retains the sign of the field everywhere and is barely affected by truncation.

[4]Kohlbrenner and Alexa [2026] also observed suboptimal ground truth performance.

Table 3. Robustness to truncation. Hausdorff / Chamfer distance (both $\times 10^{-3}$, ↓) on a single model (Rossignol) at $n = 50^3$, with the distance field truncated to $[-b, b]$. MC clamps to $\pm b$, while the others discard out-of-range samples.

Method	No truncation	$b = 0.1$	$b = 0.05$	$b = 0.005$	$b = 0.002$
RFTA	17.0 / 0.602	491 / 114	406 / 69.5	138 / 16.7	267 / 30.4
MES	13.5 / 0.618	510 / 107	208 / 38.4	365 / 57.4	2090 / 322
MC	23.0 / 1.35	23.0 / 1.35	23.0 / 1.35	**23.0** / 2.17	**20.1** / 2.79
Ours	**6.85** / **0.280**	**6.91** / **0.289**	**6.80** / **0.311**	24.9 / **0.932**	53.6 / **2.67**

Table 4. Robustness to noisy distance values. Mean Hausdorff / Chamfer distance (both $\times 10^{-3}$, ↓) over 10 shapes at $n = 50^3$.

Method	$\sigma = 0$	$\sigma = 0.001$	$\sigma = 0.005$	$\sigma = 0.01$
RFTA	18.5 / 0.850	28.1 / 1.29	44.5 / 4.74	84.1 / 11.5
MES	16.8 / 1.08	14.8 / 1.11	35.2 / 4.03	38.7 / 8.23
MC	26.7 / 1.36	26.6 / 1.52	24.7 / **2.77**	**32.2** / **4.38**
Ours	**10.3** / **0.349**	**13.2** / **1.08**	**21.1** / 3.12	36.8 / 5.16

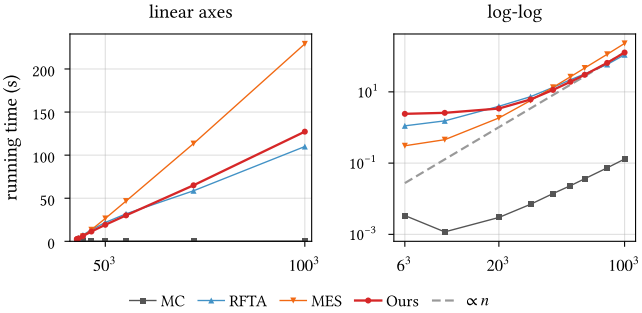


Fig. 8. Average runtime (seconds) versus samples n over 30 Thingi10K inputs, on linear (left) and logarithmic (right) axes.

It therefore outperforms our approach in Hausdorff distance once $b \leq 0.005$, although our Chamfer distance stays the lowest at every bound.

Noise. We perturbed the distance values with zero-mean Gaussian noise of standard deviation σ (Table 4). Our method is the most accurate of the four at $\sigma = 0$ (no noise) and $\sigma = 0.001$ in both metrics, and outperforms RFTA and MES at every noise level tested (up to $\sigma = 0.01$). However, its advantage over Marching Cubes diminishes as σ grows. Marching Cubes attains the lowest Chamfer distance above $\sigma = 0.005$ and the lowest Hausdorff distance above $\sigma = 0.01$. This is the cost of our exact interpolation; noise anywhere in the field enters our reconstruction, while Marching Cubes reads only cells that straddle the zero crossing.

High sample counts. To understand whether our approach outperforms marching cubes at high densities, we ran our method on up to 300^3 samples on the first 50 meshes in our dataset. For $\sqrt[3]{n} = 40$, 60, 80, 100, 150, 200, 300, our approach's Chamfer distance is 0.29, 0.31, 0.29, 0.23, 0.31, 0.34, 0.36 × that of MC, respectively. Thus our approach produces superior output to MC even at high densities, whereas MC outperforms RFTA and MES above $n = 60^3$ (Figure 6).

5.2 Running Time

We measured wall-clock time on an Apple MacBook Pro (Mac17,7) with an M5 Max processor (18 CPU/40 GPU cores) and 128 GB of unified memory. RFTA's GPU back-end was enabled. Figure 8 reports the mean over 30 Thingi10K meshes against the sample count n. Our approach grows almost linearly in n. Our approach is faster than MES above $\sqrt[3]{n} \geq 40$; it is 1.8× faster at $n{=}100^3$. Our approach is slightly slower than RFTA, by 1.2× at $n{=}100^3$. Marching Cubes is shown as a runtime lower bound.

Complexity. Empirically, regressing the log of our running time (Fig. 8) against $\log n$ over $\sqrt[3]{n} \geq 30$ gives a slope of ~0.84, so the cost is essentially linear in the input sample count over the tested range. The slope falls slightly below one because a fixed start-up cost of a few seconds is amortized as n grows; it rises to ~0.88 for $\sqrt[3]{n} \geq 40$). Combining the per-stage analyses in Sec. 4.1, Sec. 4.2, and Appendix B, we analyze the expected cost for well-distributed samples and the worst case.

Average case. Constructing the regular triangulation of n samples takes expected $O(n \log n)$ time. Its expected vertex degree is constant, so each representative neighbor set has $O(1)$ size and the short-arc stage costs $O(n)$ in total. The iterative projection of Sec. 4.2 runs a small constant number of outer iterations. Each iteration does three things: an inner optimization at every sample, a feasibility check on every resulting projection, and one rebuild of the interpolant with the freshly committed tangent points. The inner optimization takes a bounded number of steps. Each step costs $O(\log n)$, since a BVH query locates the patches covering the query point, while bounded patch overlap keeps the blend itself $O(1)$. The feasibility checks run against constant-size RT neighbor sets. The rebuild costs $O(n \log n)$ and dominates, so the whole stage is $O(n \log n)$ and folds into the PU–RBF term above. The total runtime is:

$$\underbrace{O(n \log n)}_{\text{RT construction}} + \underbrace{O(n)}_{\text{short-arc}} + \underbrace{O(n \log n)}_{\substack{\text{PU–RBF \&} \\ \text{iterative projection}}} = O(n \log n)$$

Worst case. A three-dimensional regular triangulation guarantees no constant bound on vertex degree: it may contain $O(n^2)$ simplices and take $O(n^2)$ time to construct, and a single sample may acquire $O(n)$ representative neighbors, in which case the short-arc stage degenerates to the unrestricted $O(n\,k^2) = O(n^3)$ bound above. Reaching this bound, however, requires nearly degenerate sample distributions, e.g., points on two skew lines (with equal weights) [Erickson 2003].

Convergence. The iterative projection converges within only a handful of steps on every input we have tested; we always run a constant 10 iterations. Figure 9 shows the per-step error on two representative inputs. The Chamfer distance drops sharply over the first few steps and then quickly plateaus, indicating that the majority of feasible projections are recovered early and further refinement yields diminishing returns. This justifies our fixed step budget. The Hausdorff distance follows the same overall trend; the small upticks visible on Armadillo reflect its sensitivity to a single worst point, while the Chamfer distance, which averages over all points, continues to decrease monotonically.

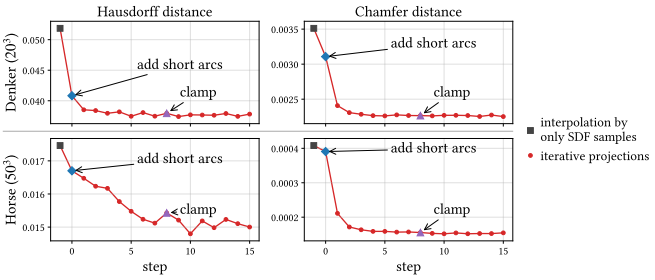


Fig. 9. Convergence of iterative projection on two representative inputs.

Fig. 10. Given samples of an unsigned distance field of an open surface, our method extracts a thin closed shell around it by contouring a small positive isovalue (0.005 in this example).

6 Conclusion & Limitations

We presented a radial basis function (RBF) interpolation of signed distance field samples that pairs every input sample $(\mathbf{x}_i, d_i)$ with a point $(\mathbf{y}_i, 0)$ on its implied tangent sphere. The reconstructed zero level set passes exactly through the recovered tangent points. Avoiding a Hermite term keeps the formulation well-defined at sharp features, where surface normals can be multi-valued. We proposed an S^2 gradient-based optimization to refine each tangent point and an analytic boundary construction to handle samples whose exposed regions collapse. Our approach scales efficiently with partition of unity.

The interpolant supports evaluation at arbitrary locations and could drive adaptive sample-insertion schemes such as RCR [Kohlbrenner and Alexa 2025a]. Figure 10 shows an example where our approach is able to extract an open surface by contouring a small, positive isovalue on UDF. This is possible because $\tilde{D}$ approximates a distance field rather than an indicator function as in sPSR, so a positive isovalue is the offset surface at that distance. We leave a general exploration of unsigned distance fields as future work.

Several limitations remain. $\tilde{D}$ is only an approximate SDF and is not guaranteed to satisfy the Eikonal equation $\|\nabla\tilde{D}\| = 1$, so the aforementioned adaptive-insertion schemes would need care; we leave this direction to future work. Furthermore, our kernel is isotropic and so has no preferred direction along a crease. We reproduce a sharp edge not because the interpolant represents the crease itself, but because the tangent points recovered along the edge are dense enough that the residual undulation between them is imperceptible. Recovering sharp features from sparser tangent points may call for an anisotropic formulation [De Marchi et al. 2020], which we also leave to future work. Our extraction resolution is set by a heuristic tied to the input sample count. We have no criterion certifying that it is fine enough. A thin structure that $\tilde{D}$ represents may still be missed if the extraction grid does not resolve

it. The grid can always be refined further, but choosing a sufficient resolution is currently left to the user.

Acknowledgments

We are grateful for the anonymous reviewers' constructive suggestions which greatly improved the paper. This work was supported by the United States National Science Foundation (IIS-2402893).

References

Nina Amenta, Sunghee Choi, and Ravi Krishna Kolluri. 2001. The Power Crust. In *Proceedings of the Sixth ACM Symposium on Solid Modeling and Applications*. 249–266. doi:10.1145/376957.376986

Ken Anjyo, John P. Lewis, and Frédéric Pighin. 2014. Scattered Data Interpolation for Computer Graphics. In *ACM SIGGRAPH 2014 Courses*. 1–69.

Christopher Batty. 2011. Sharp Signed Distance Field Reconstruction. https://web.archive.org/web/20111112202136/http://www.cs.columbia.edu/~batty/misc/levelset_meshing/level_set_reconstruction.html

Jonathan C. Carr, Richard K. Beatson, Jon B. Cherrie, Tim J. Mitchell, W. Richard Fright, Bruce C. McCallum, and Tim R. Evans. 2001. Reconstruction and Representation of 3D Objects with Radial Basis Functions. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*. ACM, 67–76.

Zhiqin Chen, Andrea Tagliasacchi, Thomas Funkhouser, and Hao Zhang. 2022. Neural Dual Contouring. *ACM Transactions on Graphics* 41, 4 (2022), 1–13.

Zhiqin Chen and Hao Zhang. 2021. Neural Marching Cubes. *ACM Transactions on Graphics* 40, 6 (2021), 1–15.

B. R. de Araújo, Daniel S. Lopes, Pauline Jepp, Joaquim A. Jorge, and Brian Wyvill. 2015. A Survey on Implicit Surface Polygonization. *ACM Comput. Surv.* 47, 4 (May 2015), 60:1–60:39. doi:10.1145/2732197

S. De Marchi, F. Marchetti, and E. Perracchione. 2020. Jumping with variably scaled discontinuous kernels (VSDKs). *BIT Numerical Mathematics* 60, 2 (June 2020), 441–463. doi:10.1007/s10543-019-00786-z

Akio Doi and Akio Koide. 1991. An efficient method of triangulating equi-valued surfaces by using tetrahedral cells. *IEICE Transactions on Information and Systems* 74, 1 (1991), 214–224.

Kathryn P. Drake, Edward J. Fuselier, and Grady B. Wright. 2022. Implicit Surface Reconstruction with a Curl-Free Radial Basis Function Partition of Unity Method. *SIAM Journal on Scientific Computing* 44, 5 (2022). doi:10.1137/22M1474485

Jean Duchon. 1977. Splines Minimizing Rotation-Invariant Semi-Norms in Sobolev Spaces. In *Constructive Theory of Functions of Several Variables*. Springer Berlin Heidelberg, 85–100. doi:10.1007/BFb0086566

Herbert Edelsbrunner. 1995. The Union of Balls and Its Dual Shape. *Discrete & Computational Geometry* 13, 3–4 (1995), 415–440. doi:10.1007/BF02574053

Jeff Erickson. 2003. Nice Point Sets Can Have Nasty Delaunay Triangulations. *Discrete & Computational Geometry* 30, 1 (2003), 109–132. doi:10.1007/s00454-003-2927-4

Richard Franke. 1982. Smooth interpolation of scattered data by local thin plate splines. *Computers & Mathematics with Applications* 8, 4 (Jan. 1982), 273–281. doi:10.1016/0898-1221(82)90009-8

Zhiyang Huang, Nathan Carr, and Tao Ju. 2019. Variational Implicit Point Set Surfaces. *ACM Transactions on Graphics* 38, 4 (2019), 1–13.

Jisung Hwang and Minhyuk Sung. 2024. Occupancy-Based Dual Contouring. In *SIGGRAPH Asia 2024 Conference Papers*.

Alec Jacobson, Ladislav Kavan, and Olga Sorkine-Hornung. 2013. Robust Inside-Outside Segmentation using Generalized Winding Numbers. *ACM Transactions on Graphics* (2013).

Alec Jacobson, Daniele Panozzo, et al. 2018. libigl: A simple C++ geometry processing library. https://libigl.github.io/.

Tao Ju, Frank Losasso, Scott Schaefer, and Joe Warren. 2002. Dual Contouring of Hermite Data. In *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*. 339–346.

Michael Kazhdan and Hugues Hoppe. 2013. Screened Poisson Surface Reconstruction. *ACM Transactions on Graphics* 32, 3 (2013), 29:1–29:13.

Leif P. Kobbelt, Mario Botsch, Ulrich Schwanecke, and Hans-Peter Seidel. 2001. Feature sensitive surface extraction from volume data. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*. ACM, 57–66. doi:10.1145/383259.383265

Maximilian Kohlbrenner and Marc Alexa. 2025a. Isosurface Extraction for Signed Distance Functions using Power Diagrams. *Computer Graphics Forum* 44, 2 (2025). Proc. Eurographics 2025.

Maximilian Kohlbrenner and Marc Alexa. 2025b. A Polyhedral Construction of Empty Spheres in Discrete Distance Fields. In *SIGGRAPH 2025 Conference Papers*. 10 pages. doi:10.1145/3721238.3730748

Maximilian Kohlbrenner and Marc Alexa. 2026. Contouring Signed Distance Fields by Approximating Gradients. *Computer Graphics Forum* (2026), e70373. doi:10.1111/cgf.70373

William E. Lorensen and Harvey E. Cline. 1987. Marching Cubes: A High Resolution 3D Surface Construction Algorithm. In *Computer Graphics (Proc. SIGGRAPH)*, Vol. 21. 163–169.

I. Macêdo, J. P. Gois, and L. Velho. 2011. Hermite Radial Basis Functions Implicits. *Computer Graphics Forum* 30, 1 (2011), 27–42. doi:10.1111/j.1467-8659.2010.01785.x

Jorge Nocedal and Stephen J. Wright. 2006. *Numerical Optimization*. Springer. doi:10.1007/978-0-387-40065-5

Yutaka Ohtake, Alexander Belyaev, Marc Alexa, Greg Turk, and Hans-Peter Seidel. 2003. Multi-level Partition of Unity Implicits. *ACM Transactions on Graphics* 22, 3 (2003), 463–470.

Daxuan Ren, Hezi Shi, Jianmin Zheng, and Jianfei Cai. 2024. McGrids: Monte Carlo-Driven Adaptive Grids for Iso-Surface Extraction.

Silvia Sellán, Christopher Batty, and Oded Stein. 2023. Reach For the Spheres: Tangency-Aware Surface Reconstruction of SDFs. In *SIGGRAPH Asia 2023 Conference Papers*. Article 73, 11 pages.

Silvia Sellán, Yingying Ren, Christopher Batty, and Oded Stein. 2024. Reach for the Arcs: Reconstructing Surfaces from SDFs via Tangent Points. In *ACM SIGGRAPH 2024 Conference Papers*. 14 pages. doi:10.1145/3641519.3657419

Graham M Treece, Richard W Prager, and Andrew H Gee. 1999. Regularised marching tetrahedra: improved iso-surface extraction. *Computers & Graphics* 23, 4 (1999), 583–598.

Greg Turk and James F. O'Brien. 2002. Modelling with Implicit Surfaces That Interpolate. *ACM Transactions on Graphics* 21, 4 (2002), 855–873.

Stéfan van der Walt, Johannes L. Schönberger, Juan Nunez-Iglesias, François Boulogne, Joshua D. Warner, Neil Yager, Emmanuelle Gouillart, Tony Yu, and the scikit-image contributors. 2014. scikit-image: image processing in Python. *PeerJ* 2 (6 2014), e453. doi:10.7717/peerj.453

Pengfei Wang, Ziyang Zhang, Wensong Wang, Shuangmin Chen, Lin Lu, Shiqing Xin, Changhe Tu, and Wenping Wang. 2025. Power Diagram Enhanced Adaptive Isosurface Extraction from Signed Distance Fields. arXiv:2506.09579 [cs.CG]

Holger Wendland. 2004. *Scattered Data Approximation*. Cambridge Monographs on Applied and Computational Mathematics, Vol. 17. Cambridge University Press.

Jianjun Xia and Tao Ju. 2025. Variational Surface Reconstruction Using Natural Neighbors. *ACM Transactions on Graphics* 44, 4 (2025). doi:10.1145/3731191

Qingnan Zhou and Alec Jacobson. 2016. Thingi10K: A Dataset of 10,000 3D-Printing Models. *arXiv preprint arXiv:1605.04797* (2016).

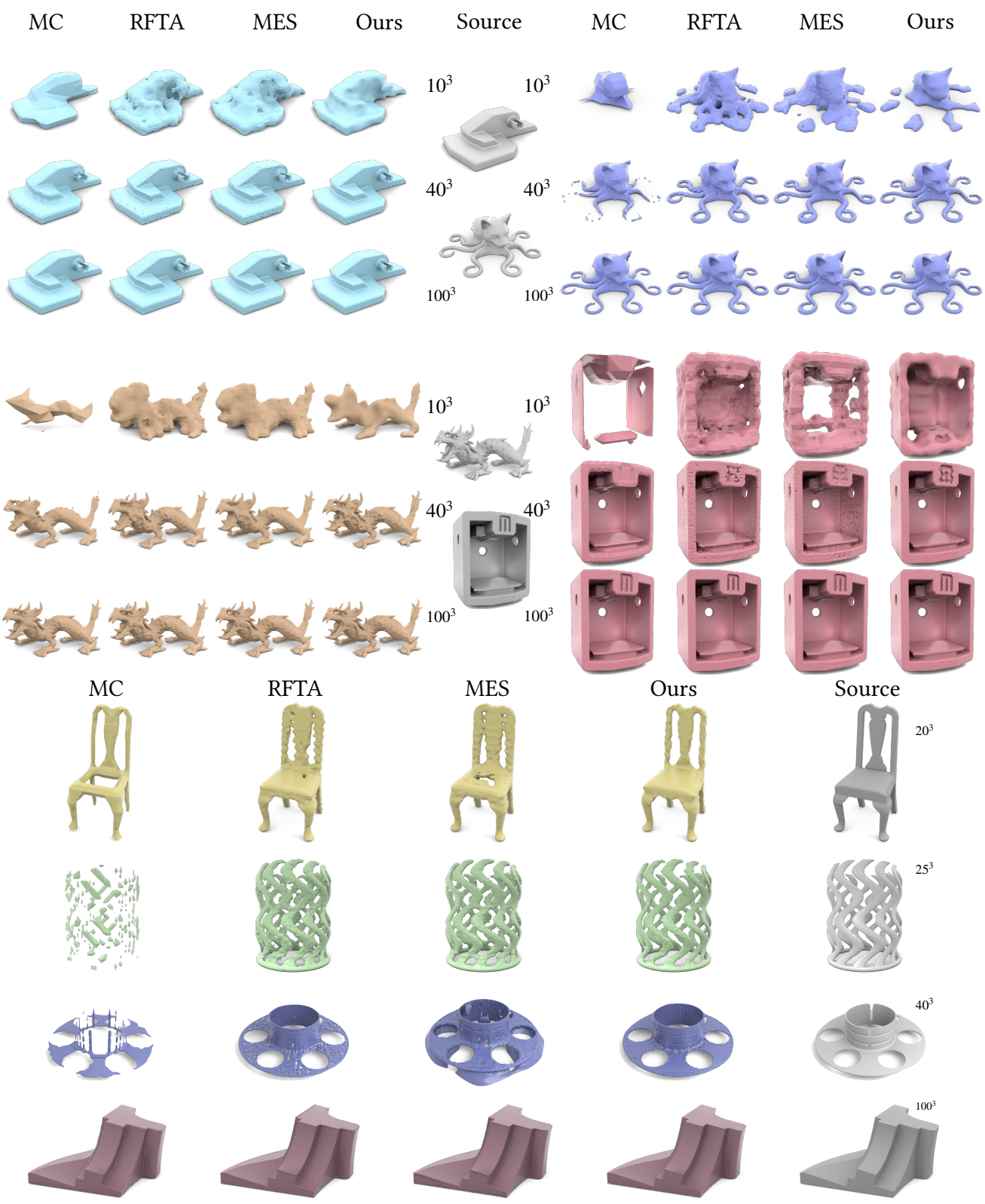


Fig. 11. Reconstructions produced by our method, MES, RFTA, and Marching Cubes at varying sampling grid densities.

A Algorithm Pseudocode

This section spells out the pipeline of the main paper's Section 4. Algorithm 1 computes the boundary arcs of one sample sphere's exposed region; Algorithm 2 wraps it into the short-arc tangent-point construction (Sec. 4.1); and Algorithm 3 combines that construction with the iterative projection loop (Sec. 4.2) to produce the final mesh.

Algorithm 1 processes the neighbors of a sample sphere S_i one at a time, and the update for each newly added cap C_{new} is bidirectional. We write $\mathcal{K}_i$ for the caps carved on S_i and $\mathcal{B}_i$ for the surviving boundary arcs, reserving C for the constraint set of Algorithm 3.

Algorithm 1 BoundaryArcs: exposed-region boundary on a sample sphere S_i.

Input: Caps $\mathcal{K}_i$ carved on S_i by its intersecting neighbors (Sec. 4.1).
Output: Boundary arcs $\mathcal{B}_i = \{a_\ell\}_\ell$ of S_i's exposed region.

$\mathcal{B}_i \leftarrow \emptyset$; $\mathcal{K}_{\text{prev}} \leftarrow \emptyset$ ▷ Caps processed so far
for each cap $C_{\text{new}} \in \mathcal{K}_i$ **do**
  for each arc $a_\ell \in \mathcal{B}_i$ **do** ▷ Step 1: clip existing arcs
    drop the portion of a_ℓ inside C_{new}
  end for
  $\mathcal{B}_{\text{new}} \leftarrow$ full circle $[0, 2\pi]$ of C_{new} ▷ Step 2: new cap's arcs
  for each cap $C \in \mathcal{K}_{\text{prev}}$ **do**
    drop the portion of $\mathcal{B}_{\text{new}}$ inside C
  end for
  $\mathcal{B}_i \leftarrow \mathcal{B}_i \cup \mathcal{B}_{\text{new}}$; $\mathcal{K}_{\text{prev}} \leftarrow \mathcal{K}_{\text{prev}} \cup \{C_{\text{new}}\}$
end for
return surviving arcs $\mathcal{B}_i$

ShortArcs (Algorithm 2) declares a sample sphere degenerate when the total length of its exposed-region boundary falls below ϵ_{degen}, and emits the midpoint of each surviving arc as a candidate tangent point. A sphere with a non-degenerate exposed region returns $\mathcal{A}_i = \emptyset$ and is left to the iterative projection loop.

Algorithm 2 ShortArcs: candidate tangent points from collapsed exposed regions.

Input: SDF samples $\{(\mathbf{x}_i, d_i)\}_{i=1}^n$; collapse threshold ϵ_{degen} (default 10^{-5}).
Output: Per-sphere candidate tangent points $\{\mathcal{A}_i\}_{i=1}^n$.

for all $i = 1, \ldots, n$ **in parallel do**
  $\mathcal{A}_i \leftarrow \emptyset$
  $\mathcal{K}_i \leftarrow$ caps carved on S_i by its intersecting neighbors
  if $\mathcal{K}_i \neq \emptyset$ **then** ▷ A fully exposed S_i cannot be degenerate
    $\mathcal{B}_i \leftarrow$ BoundaryArcs$(\mathcal{K}_i)$ ▷ Algorithm 1
    $L_i \leftarrow \sum_{a_\ell \in \mathcal{B}_i} \rho_{c(\ell)} \left(t_\ell^{\text{end}} - t_\ell^{\text{start}}\right)$ ▷ Exposed boundary length
    if $L_i < \epsilon_{\text{degen}}$ **then** ▷ Exposed region collapsed to a point
      $\mathcal{A}_i \leftarrow \{$ Midpoint$(a_\ell) : a_\ell \in \mathcal{B}_i \}$
    end if
  end if
end for
return $\{\mathcal{A}_i\}_{i=1}^n$

In Algorithm 3, OptimizeGradient performs the S^2 gradient-based refinement of the tangent direction $\mathbf{g}_i$ at sample $\mathbf{x}_i$ under the current $\tilde{D}$. Feasible tests whether the implied tangent point $\mathbf{y}_i$ lies in S_i's exposed region using the cached neighbor list with a BVH fallback (Sec. 4.2). At most one short-arc candidate per sphere is committed, the one whose initial RBF value $|\tilde{D}_0|$ is smallest; the spheres that keep a candidate form the set $\mathcal{T}$ and are excluded from the projection loop.

Algorithm 3 RBF-SDF interpolation with implied tangent points.

Input: SDF samples $\{(\mathbf{x}_i, d_i)\}_{i=1}^n$; outer iterations T.
Output: Triangle mesh of the zero level set of an interpolant $\tilde{D}$ that fits the input SDF samples and the recovered tangent points.

$\tilde{D}_0 \leftarrow \text{RBF}\left(\{(\mathbf{x}_i, d_i)\}_{i=1}^n\right)$ ▷ Initial interpolant from SDF samples only
$\{\mathcal{A}_i\}_{i=1}^n \leftarrow$ ShortArcs$\left(\{(\mathbf{x}_i, d_i)\}_{i=1}^n\right)$ ▷ Algorithm 2
$\mathbf{a}_i^\star \leftarrow \arg\min_{\mathbf{a} \in \mathcal{A}_i} |\tilde{D}_0(\mathbf{a})|$ ▷ Best candidate on each S_i with $\mathcal{A}_i \neq \emptyset$
$\mathcal{T} \leftarrow \left\{ i : \mathcal{A}_i \neq \emptyset,\ |\tilde{D}_0(\mathbf{a}_i^\star)| < \epsilon_{\text{val}} \right\}$ ▷ Spheres with an accepted short-arc point
$\mathcal{Y} \leftarrow \left\{ \mathbf{a}_i^\star : i \in \mathcal{T} \right\}$ ▷ At most one tangent point per sphere (Sec. 4.1)
$C \leftarrow \{(\mathbf{x}_i, d_i)\}_{i=1}^n \cup \{(\mathbf{y}, 0) : \mathbf{y} \in \mathcal{Y}\}$
$\tilde{D} \leftarrow \text{RBF}(C)$ ▷ Refit with short-arc tangent points
for $t = 1, \ldots, T$ **do**
  if $t = 1$ **then**
    $\mathcal{I}_t \leftarrow \{\, i \notin \mathcal{T} : S_i$ intersects some neighbor sphere $\}$ ▷ Defer no-neighbor samples on the first iteration
  else
    $\mathcal{I}_t \leftarrow \{\, i :$ sample i has no committed tangent point $\}$
  end if
  for all $i \in \mathcal{I}_t$ **in parallel do**
    $\mathbf{g}_i \leftarrow$ OptimizeGradient$\left(\mathbf{x}_i, d_i, \tilde{D}\right)$ ▷ S^2 minimization of f, Sec. 4.2
    $\mathbf{y}_i \leftarrow \mathbf{x}_i - d_i\, \mathbf{g}_i$ ▷ Implied tangent point on S_i
    if Feasible$(\mathbf{y}_i)$ **then** ▷ $\mathbf{y}_i$ lies in S_i's exposed region
      $C \leftarrow C \cup \{(\mathbf{y}_i, 0)\}$
    end if
  end for
  $\tilde{D} \leftarrow \text{RBF}(C)$ ▷ Refit interpolant with newly committed tangent points
end for
return IsosurfaceExtract$(\tilde{D}, 0)$ ▷ Dual Contouring

B RBF Acceleration

A direct implementation of the interpolatory RBF solve is $O(n^3)$ due to dense matrix inversion, which becomes prohibitive for high-resolution SDF grids. We reduce this to $O(n \log n)$ with a partition of unity decomposition [Franke 1982; Ohtake et al. 2003]. We perform the decomposition anew for each iteration's RBF solve. We cover the bounding domain with overlapping patches $\{U_j\}_{j=1}^J$. We solve independent local RBF interpolations f_j from the samples in each U_j. The local solutions are blended into a global function via a

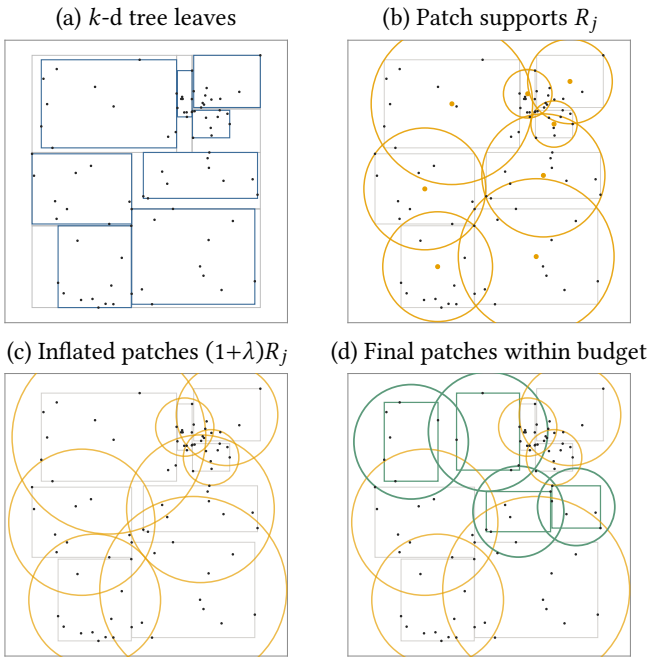


Fig. 12. Partition-of-unity patch construction. (a) A k-d tree partitions the samples into leaves. Grey lines show the ambient median-split partition. Blue rectangles are each leaf's tight bounding box. (b) Each leaf induces a Wendland support of radius R_j equal to half its bounding-box diagonal. (c) Supports are inflated by $(1+\lambda)$ so neighboring patches share a non-trivial common region. (d) Patches whose post-inflation sample count exceeds the budget are recursively subdivided (green).

normalized partition of unity,

$$\tilde{D}(\mathbf{x}) \;=\; \sum_{j=1}^{J} w_j(\mathbf{x})\, f_j(\mathbf{x}), \qquad w_j(\mathbf{x}) \;:=\; \frac{\hat{w}_j(\mathbf{x})}{\sum_k \hat{w}_k(\mathbf{x})}, \tag{7}$$

where each $\hat{w}_j$ is a Wendland C^2 compactly supported function [Wendland 2004],

$$\hat{w}_j(\mathbf{x}) \;=\; \left(1 - \frac{\|\mathbf{x}-\boldsymbol{\xi}_j\|}{R_j}\right)_+^4 \left(4\,\frac{\|\mathbf{x}-\boldsymbol{\xi}_j\|}{R_j} + 1\right), \tag{8}$$

centered at the patch center $\boldsymbol{\xi}_j$ with support radius R_j, where $(\,\cdot\,)_+ = \max(\,\cdot\,, 0)$.

Patch construction. We partition the input samples with a k-d tree (Fig. 12a). At each level we split the current node's bounding box along its longest axis at the median sample's coordinate, and recurse until every leaf holds at most 200 samples. Each leaf becomes a patch U_j, with center $\boldsymbol{\xi}_j$ at the leaf's bounding-box center and support radius R_j equal to half the leaf's bounding-box diagonal (Fig. 12b). The Wendland weight $\hat{w}_j$ is therefore supported on a ball of radius R_j around $\boldsymbol{\xi}_j$ that covers the entire leaf, so every sample in the leaf is included in U_j. We then inflate each patch's support radius by a factor of $1 + \lambda$ to introduce overlap between neighboring patches (Fig. 12c). Inflation can pull in enough additional samples to exceed the per-patch solve budget, particularly after iterative projection as projected tangent points cluster more densely on the surface than the original SDF samples. Whenever the post-inflation sample count of a patch exceeds the budget (set to 675), we recursively subdivide its leaf before inflation and rebuild the patch (Fig. 12d). We skip this subdivision when the leaf already contains only two samples, to avoid degenerate splits.

We empirically set $\lambda = 0.2$. Larger λ increases neighboring-patch overlap, so adjacent local solves share more samples and agree more closely on their common region, at the cost of more patches (and more local solves).

Blended interpolant. On each patch U_j we solve the local RBF system (3) on the subset of samples in U_j to obtain a local interpolant f_j, and blend them via the partition of unity (Eq. 7). To evaluate at a query point $\mathbf{x}$, we identify the patches U_j whose supports contain $\mathbf{x}$, evaluate $f_j(\mathbf{x})$ on each, and combine them with the normalized weights $w_j(\mathbf{x})$.

In rare cases a query point falls in a small gap between neighboring patches—outside $\bigcup_j \mathrm{supp}(\hat{w}_j)$. We then fall back to the patch of the nearest constraint point and evaluate its local RBF f_j directly. The cubic kernel fit to SDF data produces a smooth interpolant that is approximately Eikonal ($\|\nabla f_j\| \approx 1$) away from the medial axis (Figure 18), so short-range extrapolation across an inter-patch gap remains gradient-consistent with the blended interpolant on either side and introduces no visible discontinuity in $\tilde{D}$.

A more consequential degeneracy arises when the points within a patch are coplanar or collinear. (Singular values less than 0.01× the largest are considered degenerate.) The local system is then rank-deficient, and the interpolant is unconstrained outside the sample subspace. To address this, we add additional samples to the patch. Along the smallest singular value dimension, we add the closest sample to the patch center in both the positive and negative direction that are at least one patch radius away from the subspace. The additional samples enter the local interpolation system only; the patch's center, extent, and weight function are unchanged. We repeat this until the system is non-degenerate.

By construction, $w_j(\mathbf{x}_i) > 0$ implies $\mathbf{x}_i$ lies in neighborhood U_j, so every active patch at $\mathbf{x}_i$ interpolates it ($f_j(\mathbf{x}_i) = d_i$). Each local RBF f_j inherits value interpolation, C^2 smoothness, and linear reproduction from the global RBF. The partition of unity preserves all three when blending them into $\tilde{D}$.

Complexity. The decomposition is built with a k-d tree at $O(n \log n)$ cost, followed by bounded re-splitting of patches that exceed the per-patch budget m after inflation, which adds at most another $O(n \log n)$ cost. Given this constant budget, each per-patch solve costs $O(m^3) = O(1)$, and bounded patch overlap ensures $Jm = O(n)$, where J is the patch count, so the cumulative solve cost is $O(n)$. The total cost is therefore dominated by the $O(n \log n)$ partitioning. We empirically verify these assumptions in Fig. 13. Per-patch sample counts remain concentrated below the budget across our benchmark, and the total patch count J grows linearly with n. The end-to-end speedup over the global RBF solve is reported in Figure 15.

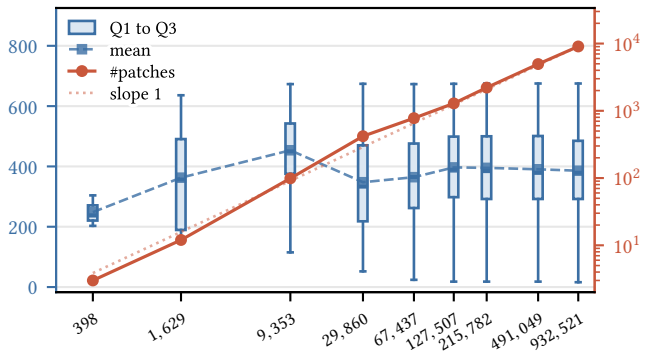


Fig. 13. Statistics of partition-of-unity patches across the CHAIR test mesh. The boxes plot the distribution of per-patch sample counts after subdivision, all below the budget of 675; the solid orange line plots total patch count J vs. constraint point count n. The dashed orange line plots a reference slope of $O(n)$. The x-axis (interpolated points) and orange y-axis on the right are on a log scale, while the box plot's y-axis on the left is linear.

C Per-Method Accuracy

This section accompanies Figure 6 of the main paper with the underlying numerical means (Table 5) and the per-mesh distributions (Figure 14).

Successful-cases denominator. Each (method, $\sqrt[3]{n}$) tuple is evaluated on a 300-mesh subset of Thingi10K. Every reported metric is averaged over the cases on which the method produced a non-empty mesh. Marching Cubes returns no mesh when all SDF samples share the same sign. This occured 48, 7, and 1 times when $\sqrt[3]{n} \in \{6, 10, 20\}$, respectively. The MES baselines also fail at the same low resolutions (60, 11, and 1 times, respectively, plus a handful at $\sqrt[3]{n} = 100$). RFTA and our method both complete on all 300 inputs at every tested resolution. The boxplots in Figure 14 share the same denominator as the table. Whiskers omit outliers, so the long upper tails of the MES variants at high resolution (visible as Chamfer/Hausdorff outliers an order of magnitude above their medians) are reflected in the means but not in the whisker extents.

D Ablations

D.1 Partition of unity

We measured the wall-clock performance of our algorithm with and without the partition-of-unity acceleration (Figure 15). The non-accelerated variant solves a single global RBF system over all n samples, which runs in $O(n^3)$ time, and quickly becomes impractical (already 2169 s at $K = 30$). The right panel of Fig. 15 confirms that the PU and global reconstructions look very similar at both $K = 10$ and $K = 30$.

D.2 Short arc and clamping

Figure 17 evaluates our algorithm with and without the short-arc seeding (Sec. 4.1) and clamping (Sec. 4.2) on three inputs (EIFFEL, LOEWE, HORSE). All four variants of our algorithm, including the simplest version without short-arc seeding or clamping, outperform RFTA [Sellán et al. 2024] for most models and resolutions. When focusing on the relative Chamfer distance of our variants versus the simplest version, we see that both switches improve performance at higher sampling rates. At low resolutions, clamping is neutral and short-arc seeding harms the EIFFEL example. We have not tested this extensively, but believe it is due to EIFFEL's sharp features. Enabling both is best for $n \geq 25^3$ at every resolution on EIFFEL and LOEWE, and at all but two on HORSE, reaching 0.73× on EIFFEL at n=100^3. On the smoother LOEWE and HORSE examples, the variants have a smaller effect; the two together save about 8% at the finest sampling.

Figure 16i illustrates a subtler degeneracy. When a chain of sample spheres shares a corner tangent point and their centers are nearly collinear, symmetry places a short arc not only at the true corner but also at its mirror image across the line of centers. In the interior of the sampling, surrounding spheres clip the mirror arc away; near a boundary of the sampled region no sphere is available to clip it, and the mirror arc would otherwise emit a phantom tangent-point candidate far from the true surface. Disconnected short-arc regions can be constructed deliberately as well, which is why we screen candidates by their initial RBF value rather than accepting all of them.

D.3 RBF kernel

We compared the cubic (triharmonic in 3D) kernel $\phi(r) = r^3$ used in our method against four alternatives: $\phi(r) = r$ (biharmonic in 3D), $\phi(r) = r^2 \log r$ (biharmonic in 3D), the Gaussian kernel, and Wendland's compactly supported C^2 kernel. Figure 18 shows heatmaps of the reconstructed implicit field on a representative input under each kernel. The cubic kernel produces the most stable extrapolation, with a far-field gradient close to the unit-norm property $\|\nabla D\| = 1$ of a true SDF.

D.4 Tangent-Point Collapse Threshold

Section 4.1 of the main paper introduces a length threshold ϵ_{degen} that decides when the boundary of a sphere's exposed region has shrunk enough to collapse to a single tangent point. We use $\epsilon_{\text{degen}} = 10^{-8}$ on meshes normalized to a unit bounding box. To verify that this choice is reasonable and to characterize the algorithm's sensitivity to it, we swept $\epsilon_{\text{degen}} \in \{10^{-4}, 10^{-5}, 10^{-6}, 10^{-7}, 10^{-8}\}$ on the EIFFEL mesh at two sampling densities (Tab. 6).

Our choice of $\epsilon_{\text{degen}} = 10^{-8}$ is the strictest value in our sweep. It produces tangent-point candidates closest to the underlying surface (smallest mean surface distance and highest fraction of candidates within 10^{-5} on the test mesh). The data in Tab. 6 serves as a sensitivity analysis on a single mesh. It shows that the algorithm is insensitive to ϵ_{degen} in the tested range. Chamfer and F1 are similar across this range. Hausdorff exhibits the minor variability expected of a worst-case distance on essentially similar input.

E Tangent points versus reconstruction

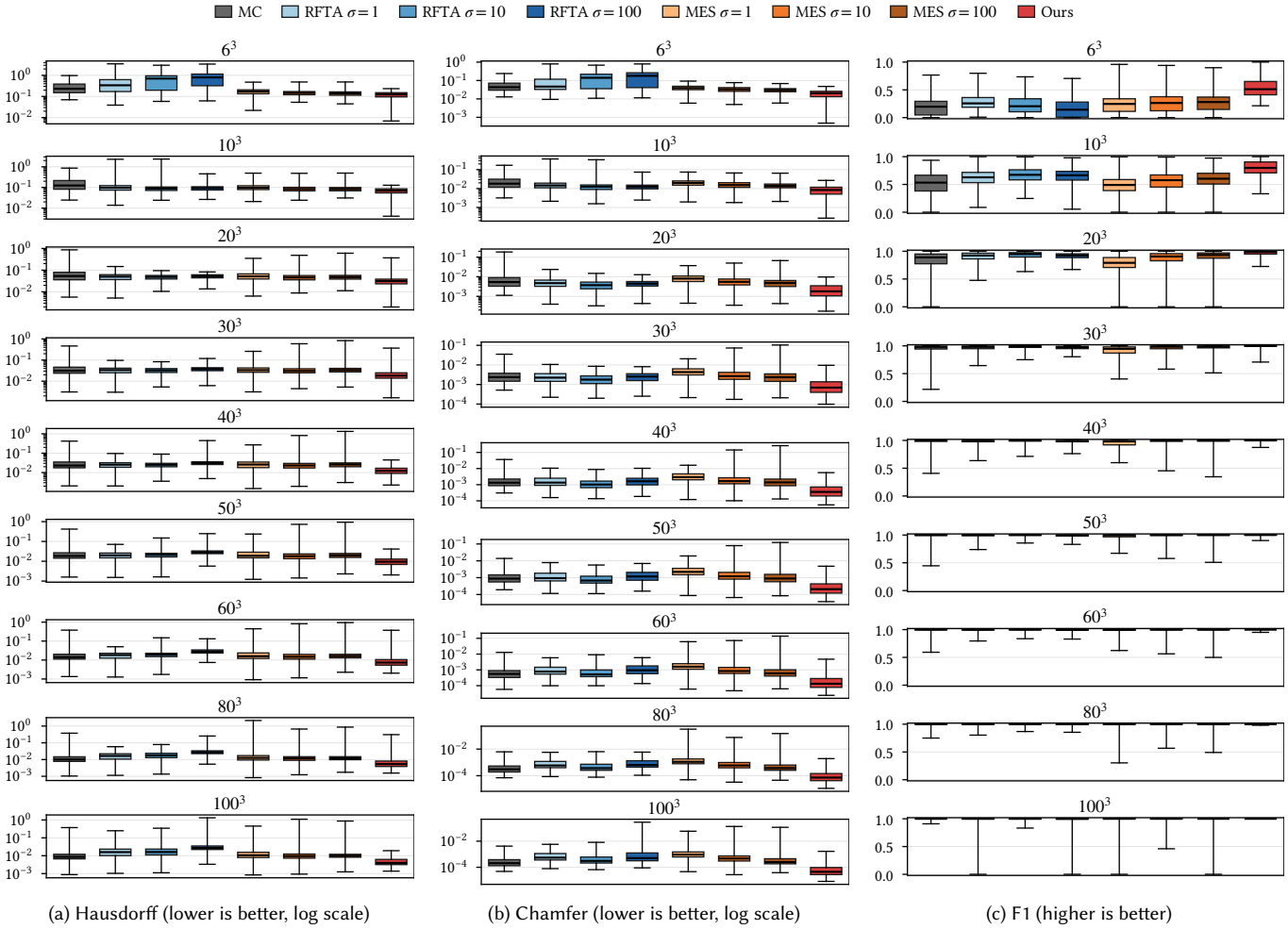
Data for every combination of tangent-point source and reconstruction method (Tables 7 and 8), plotted in Figure 7 of the main paper.

Table 5. Mean F1, Hausdorff, and Chamfer over successful cases on the 300-mesh Thingi10K subset, per (method, $\sqrt[3]{n}$). The best score in each column is bolded; ours is best in every column for every metric. RFTA and MES variants are shown for screening weights $\sigma \in \{1, 10, 100\}$. The means here are the values plotted in Figure 6 of the main paper.

Method	$\sqrt[3]{n}$								
	6	10	20	30	40	50	60	80	100
Mean F1 (↑)									
RFTA σ=1	0.279	0.629	0.898	0.959	0.976	0.985	0.990	0.993	0.992
RFTA σ=10	0.230	0.678	0.927	0.977	0.988	0.994	0.996	0.997	0.997
RFTA σ=100	0.171	0.663	0.912	0.965	0.981	0.985	0.989	0.990	0.986
MES σ=1	0.299	0.505	0.785	0.914	0.950	0.970	0.980	0.988	0.992
MES σ=10	0.327	0.579	0.882	0.959	0.980	0.984	0.987	0.991	0.992
MES σ=100	0.331	0.610	0.913	0.970	0.984	0.988	0.989	0.990	0.991
MC	0.242	0.518	0.844	0.943	0.972	0.983	0.992	0.995	0.998
Ours	**0.537**	**0.789**	**0.957**	**0.989**	**0.996**	**0.998**	**0.999**	**1.000**	**1.000**
Mean Hausdorff (↓)									
RFTA σ=1	0.439	0.129	0.0515	0.0344	0.0263	0.0211	0.0188	0.0174	0.0190
RFTA σ=10	0.694	0.0987	0.0481	0.0328	0.0254	0.0221	0.0203	0.0190	0.0197
RFTA σ=100	0.871	0.0941	0.0522	0.0374	0.0328	0.0310	0.0302	0.0300	0.0353
MES σ=1	0.174	0.109	0.0620	0.0408	0.0331	0.0300	0.0252	0.0259	0.0177
MES σ=10	0.154	0.0953	0.0580	0.0439	0.0389	0.0349	0.0295	0.0259	0.0236
MES σ=100	0.147	0.0909	0.0579	0.0495	0.0497	0.0419	0.0356	0.0314	0.0277
MC	0.299	0.174	0.0705	0.0425	0.0316	0.0236	0.0187	0.0146	0.0121
Ours	**0.122**	**0.0696**	**0.0329**	**0.0210**	**0.0136**	**0.0105**	**0.00983**	**0.00758**	**0.00518**
Mean Chamfer (↓)									
RFTA σ=1	8.7×10^{-2}	2.0×10^{-2}	5.2×10^{-3}	2.7×10^{-3}	1.9×10^{-3}	1.4×10^{-3}	1.2×10^{-3}	9.6×10^{-4}	9.3×10^{-4}
RFTA σ=10	1.5×10^{-1}	1.4×10^{-2}	4.1×10^{-3}	2.1×10^{-3}	1.4×10^{-3}	1.0×10^{-3}	8.3×10^{-4}	6.3×10^{-4}	5.7×10^{-4}
RFTA σ=100	1.9×10^{-1}	1.3×10^{-2}	4.6×10^{-3}	2.6×10^{-3}	1.9×10^{-3}	1.5×10^{-3}	1.3×10^{-3}	1.0×10^{-3}	2.0×10^{-3}
MES σ=1	4.2×10^{-2}	2.1×10^{-2}	8.6×10^{-3}	4.9×10^{-3}	3.6×10^{-3}	2.9×10^{-3}	2.4×10^{-3}	2.6×10^{-3}	1.5×10^{-3}
MES σ=10	3.5×10^{-2}	1.6×10^{-2}	6.1×10^{-3}	3.8×10^{-3}	3.0×10^{-3}	2.6×10^{-3}	2.2×10^{-3}	1.7×10^{-3}	1.6×10^{-3}
MES σ=100	3.1×10^{-2}	1.4×10^{-2}	5.3×10^{-3}	3.7×10^{-3}	3.5×10^{-3}	2.7×10^{-3}	2.4×10^{-3}	2.1×10^{-3}	1.8×10^{-3}
MC	5.8×10^{-2}	2.6×10^{-2}	7.6×10^{-3}	3.4×10^{-3}	2.1×10^{-3}	1.4×10^{-3}	8.8×10^{-4}	5.4×10^{-4}	3.8×10^{-4}
Ours	$\mathbf{2.0\times10^{-2}}$	$\mathbf{8.8\times10^{-3}}$	$\mathbf{2.5\times10^{-3}}$	$\mathbf{1.2\times10^{-3}}$	$\mathbf{6.5\times10^{-4}}$	$\mathbf{4.3\times10^{-4}}$	$\mathbf{3.1\times10^{-4}}$	$\mathbf{1.5\times10^{-4}}$	$\mathbf{9.2\times10^{-5}}$

Table 6. Sensitivity of the algorithm to the total arc length threshold ϵ_{degen} on the EIFFEL mesh at two grid resolutions. Candidates are the midpoints of short-arcs. *Candidate Quality:* Per-candidate surface distance statistics. $\overline{d}_{\text{surf}}$ is the mean. The four ≤ columns report precision as the fraction of candidates within the listed surface-distance threshold. The surface-distance measures the distance to the ground truth surface. *Reconstruction:* The final reconstruction quality of the extracted mesh against ground truth. Although the number of candidates shrinks with ϵ, up to jitter from parallelism, the reconstruction is stable across the swept range, as measured by Chamfer and F1. Hausdorff fluctuates, as is typical for a worst-case metric among similar outputs.

		Candidate Quality						Reconstruction		
$\sqrt[3]{n}$	ϵ_{degen}	#cand.	$\overline{d}_{\text{surf}}$	$\le 10^{-8}$	$\le 10^{-7}$	$\le 10^{-6}$	$\le 10^{-5}$	Hausdorff	Chamfer	F1@10^{-2}
20	10^{-4}	2202	3.7×10^{-8}	99.5%	99.5%	99.5%	100.0%	0.0403	0.00490	0.9052
20	10^{-5}	2181	3.1×10^{-12}	100.0%	100.0%	100.0%	100.0%	0.0339	0.00436	0.9347
20	10^{-6}	2184	3.1×10^{-12}	100.0%	100.0%	100.0%	100.0%	0.0354	0.00454	0.9242
20	10^{-7}	2187	3.1×10^{-12}	100.0%	100.0%	100.0%	100.0%	0.0403	0.00491	0.9077
20	10^{-8}	2120	3.2×10^{-12}	100.0%	100.0%	100.0%	100.0%	0.0359	0.00445	0.9329
50	10^{-4}	50655	6.2×10^{-6}	99.0%	99.2%	99.5%	99.6%	0.0140	0.00099	1.0000
50	10^{-5}	50114	6.2×10^{-6}	99.3%	99.4%	99.5%	99.7%	0.0141	0.00097	1.0000
50	10^{-6}	50055	6.2×10^{-6}	98.8%	98.9%	99.2%	99.7%	0.0140	0.00097	1.0000
50	10^{-7}	50065	6.2×10^{-6}	98.6%	98.7%	99.2%	99.7%	0.0140	0.00097	1.0000
50	10^{-8}	48934	6.3×10^{-6}	98.6%	98.8%	99.2%	99.7%	0.0140	0.00097	1.0000



(a) Hausdorff (lower is better, log scale) (b) Chamfer (lower is better, log scale) (c) F1 (higher is better)

Fig. 14. Per-method, per-resolution distributions of all three metrics on the 300-mesh Thingi10K subset. Each column is one metric; each row within a column is one grid resolution, titled n, from 6^3 at the top to 100^3 at the bottom. Boxes span Q1–Q3 with a median bar, and whiskers span the full range over the meshes (minimum to maximum). The F1 axis is fixed to $[0, 1]$ on every row so the rows are directly comparable; the Chamfer and Hausdorff axes are logarithmic and scaled per row. Our method's box sits below every baseline's median for Chamfer and Hausdorff at every resolution, and dominates F1 most visibly at coarse sampling ($\sqrt[3]{n} \in \{6, 10, 20\}$).

Table 7. Mean Hausdorff distance ($\times 10^{-3}$, ↓) over successful cases for every combination of tangent-point source and reconstruction method, on the first 50 models of our Thingi10K subset. GT are the ground truth closest surface points for each sample.

	$\sqrt[3]{n}$=10		$\sqrt[3]{n}$=20		$\sqrt[3]{n}$=40		$\sqrt[3]{n}$=60		$\sqrt[3]{n}$=80	
Tangent points	sPSR	RBF	sPSR	RBF	sPSR	RBF	sPSR	RBF	sPSR	RBF
RFTA	139	96.8	50.2	51.9	25.8	31.5	19.8	29.3	18.7	31.2
MES	87.8	82.5	45.4	48.3	32.3	24.0	23.2	15.0	19.1	11.4
GT	**65.5**	**54.0**	**32.3**	**23.6**	15.7	**9.75**	12.3	**6.03**	11.1	**4.38**
Ours	76.0	70.3	33.2	30.0	**15.4**	11.9	**12.0**	7.35	**10.4**	5.22

Table 8. Mean Chamfer distance ($\times 10^{-3}$, ↓) over successful cases for the same combinations and inputs as Table 7.

	$\sqrt[3]{n}$=10		$\sqrt[3]{n}$=20		$\sqrt[3]{n}$=40		$\sqrt[3]{n}$=60		$\sqrt[3]{n}$=80	
Tangent points	sPSR	RBF	sPSR	RBF	sPSR	RBF	sPSR	RBF	sPSR	RBF
RFTA	18.8	11.6	3.83	3.85	1.18	1.29	0.645	0.780	0.460	0.687
MES	14.4	12.1	5.18	4.19	2.19	1.10	1.28	0.486	0.941	0.260
GT	**9.63**	**7.11**	**3.12**	**1.93**	0.991	0.526	0.634	0.241	0.453	0.136
Ours	10.8	8.29	3.22	2.23	**0.849**	**0.442**	**0.481**	**0.176**	**0.331**	**0.0939**

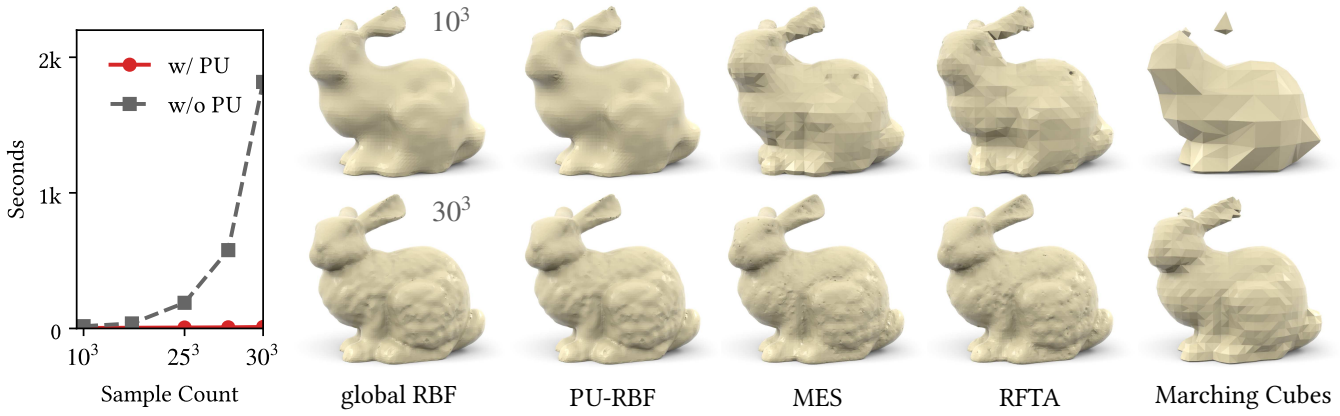


Fig. 15. The effect of partition of unity (PU) on the STANFORD BUNNY. **Left:** Without PU, our algorithm takes 1818 seconds (s) at $n = 30^3$. With PU, our algorithm takes ranges from 3.45 s to 11.04 s as n increases from 10^3 to 30^3. **Right:** Qualitative comparisons for $K = 10$ (top) and $K = 30$ (bottom).

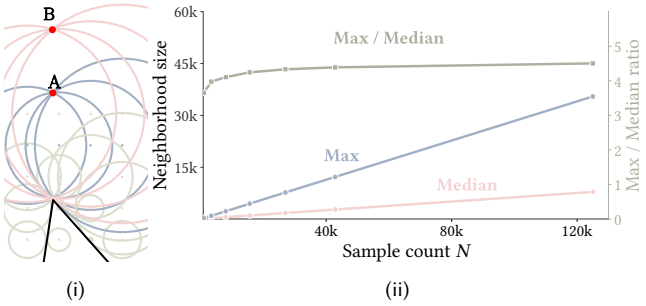


Fig. 16. (i) Phantom points illustrated in 2D. When the centers are collinear, circles passing through a shared corner produce a short arc at the true corner and a mirror short arc across the line of centers. The interior mirror (A) is clipped by surrounding circles; the mirror at the sampling boundary (B) escapes clipping and emits a spurious tangent-point candidate. (ii) Median and maximum number of intersecting neighbors for each sample vs. sample count n across the BUNNY test mesh. Both grow linearly with n, so the unbounded per-sphere cost scales as $O(n^2)$ and the total cost as $O(n^3)$.

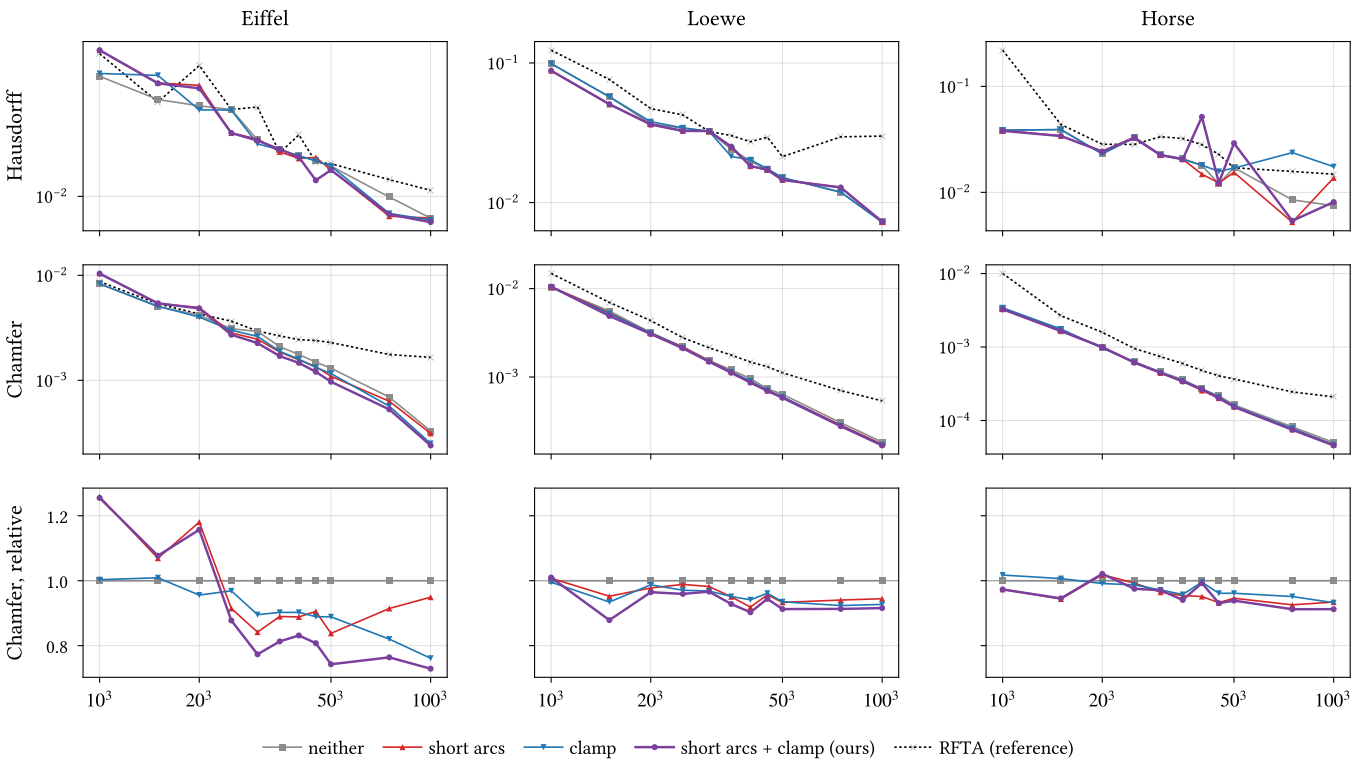


Fig. 17. Short-arc seeding and clamping ablated independently. The top two rows show absolute Hausdoff and Chamfer distance, respectively, versus RFTA [Sellán et al. 2024]. All four variants of our algorithm have lower error than RFTA for most models and resolutions. The bottom row shows relative Chamfer distance versus our algorithm without short-arc seeding or clamping. At higher sampling rates, both switches improve performance. Clamping is neutral on coarse samples, whereas short-arc seeding harms a sharp model like Eiffel at low resolution. We show Hausdorff in absolute form only; which variant is lowest is essentially random.

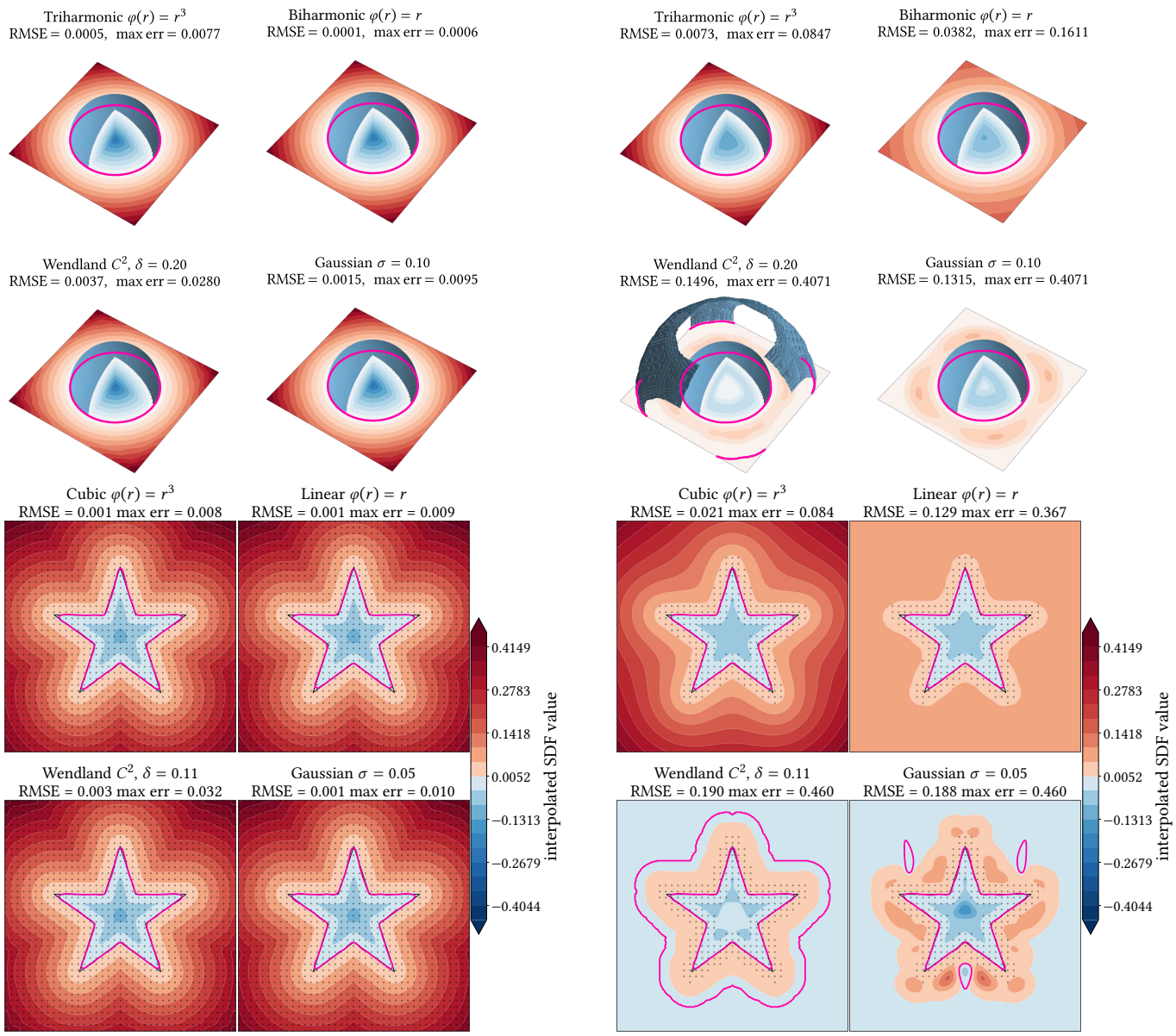


Fig. 18. Heatmaps of the reconstructed implicit field under four RBF kernels on a sphere in 3D and a star in 2D: 3D triharmonic (r^3, ours), 3D biharmonic (r), Gaussian, and Wendland's compactly supported C^2. Left: With dense samples throughout the domain, all kernels have low error. Right: With only a narrow band of samples, only the cubic kernel yields a smooth field whose far-field gradient stays closest to unit norm. In 2D, we also evaluated the triharmonic ($r^4 \log r$) and biharmonic ($r^2 \log r$) kernels; they produced higher error than the cubic kernel.