

Leaps and Bounds: An Improved Point Cloud Winding Number Formulation for Fast Normal Estimation and Surface Reconstruction

Chamin Hewa Koneputugodage Dylan Campbell Stephen Gould

The Australian National University

{chamin.hewa, dylan.campbell, stephen.gould}@anu.edu.au

Abstract

Recent methods for point cloud surface normal estimation predominantly use the generalized winding number field induced by the normals. Optimizing the field towards satisfying desired properties, such as the input points being on the surface defined by the field, provides a principled way to obtain globally consistent surface normals. However, we show that the existing winding number formulation for point clouds is a poor approximation near the input surface points, diverging as the query point approaches a surface point. This is problematic for methods that rely on the accuracy and stability of this approximation, requiring heuristics to compensate. Instead, we derive a more accurate approximation that is properly bounded and converges to the correct value. We then examine two distinct approaches that optimize for globally consistent normals using point cloud winding numbers. We show how the original unbounded formulation influences key design choices in both methods and demonstrate that substituting our formulation yields substantive improvements with respect to normal estimation and surface reconstruction accuracy.

1. Introduction

Surface normal estimation and 3D reconstruction are fundamental tasks in computer vision and graphics. Given a point cloud, normal estimation assigns an outward pointing unit normal vector to each point, providing crucial geometric information. Surface reconstruction, on the other hand, aims to recover the underlying 3D surface from the point cloud. These techniques are crucial for various applications, including object recognition, scene understanding, rendering, and 3D printing [9, 24].

In order to guarantee that the 3D representation is watertight, most methods use implicit fields, where the surface of the object is represented as the level set of a continuous field defined on the domain. Often these fields are parameterized by some function space, e.g., basis functions defined

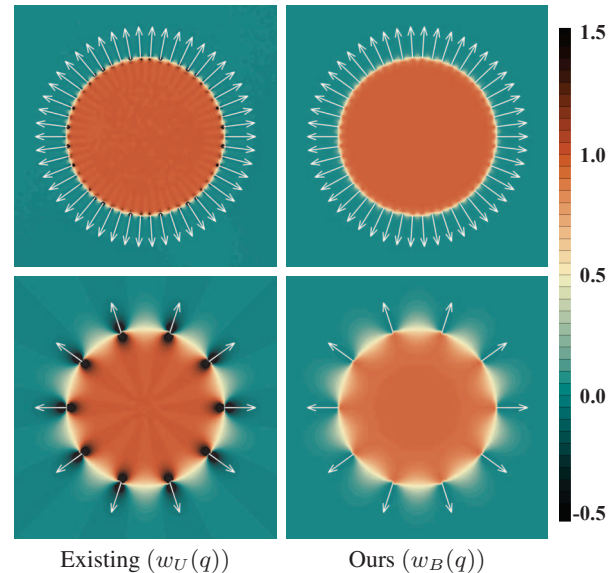


Figure 1. The winding number field robustly determines whether points are inside or outside a shape. It resembles an indicator function: 1 inside, 0 outside, and intermediate values where uncertain. Here, we plot the 2D winding number field induced by 50 (**top**) and 10 (**bottom**) input surface points given their ground-truth attributes such as normals (white arrows). The existing formulation $w_U(q)$ (**left**) has values far outside $[0, 1]$, while our corrected formulation $w_B(q)$ (**right**) stays close to this range and better resembles an indicator function. See Fig. 4 (a–b) for histograms of the winding number values of these 2D examples.

on some spatial data structure or coordinate neural networks [9, 26, 44]. These methods optimize the implicit field to best represent the surface based on the input point cloud and desirable properties of the field.

Recently, methods that directly parameterize a field with normals rather than neural network parameters have become popular for the unoriented reconstruction task [17, 31, 32, 49]. Similar to the previous approaches, they optimize towards appropriate properties of the parameterized field. However, instead of optimizing the parameters of some function space, they optimize the normals directly.

This enables more efficient optimization, since modifying local occupancy around the surface is easier when using a normal-based parameterization. The most commonly used field is the generalized winding number field [4, 23], which has many desirable properties: it does not need extra basis function parameters, its values have clear meaning, and there are fast ways to approximate it. However, we identify a key problem with the generalized winding number for point clouds: it is a poor approximation near input points, where it diverges rather than remaining finite, as it ought.

We derive a better approximation for the winding number for point clouds, and show that our formulation is bounded and has correct limiting value (visualized in Fig. 1). We then investigate two very different approaches to optimizing the winding number field for globally consistent surface normal estimation: GCNO [49] and WNNC [32]. The former constructs multiple losses over the field and optimizes with L-BFGS [42] while the latter solves an underdetermined linear system with alternating steepest descent and normal update steps. We demonstrate that substituting our formulation into each approach leads to significant improvements. Our main contributions are

1. a more accurate approximation for point cloud winding numbers that better aligns with key properties of the true winding number; and
2. integrating our improved approximation into two distinct winding number-based normal estimation methods, demonstrating its effectiveness.

We further analyze key components of both approaches, emphasizing that both methods resort to particular design choices to alleviate the divergent nature of the original point cloud winding number formulation.

2. Related Work

Surface normal estimation. Surface normal estimation can be divided into local and global methods. Local methods estimate the normals locally by fitting a local surface and then propagate for a consistent orientation. Seminal work by Hoppe et al. [16] used PCA for local normal estimation and a minimum spanning tree for propagation. Following this, many methods improved either (or both) steps [11, 13, 19, 20, 29, 38, 39, 48]. Global methods, on the other hand, optimize a global function to determine all normal [1, 21, 36, 40, 47]. More recently, many neural network based approaches have been proposed [6, 7, 14, 30].

Surface reconstruction with implicit fields. Surface reconstruction via implicit fields represents surfaces using an implicit function optimized to satisfy desired properties. Early methods relied on structured basis functions and required ground truth normals [10, 25, 26, 41, 43], with Poisson Surface Reconstruction (PSR) [26] being a notable example. Recent neural implicit methods [37, 44] offer greater

flexibility and regularization, enabling reconstruction without normals [2, 3, 8, 12, 27, 28, 45]. However, while robust to defects, neural methods can be less reliable than classical approaches on clean point clouds [22].

Generalized Winding Numbers. Jacobson et al. [23] generalize the classic winding number from 2D curves to 3D meshes, enabling robust inside-outside segmentation. For watertight meshes it matches the occupancy indicator function, and when defects are present it smoothly transitions, enabling many volumetric applications for previously problematic meshes. Barill et al. [4] noted its high computational cost for large meshes and unstructured triangle soups, such as scanned data. Thus, they adapt the winding number formulation to work for point clouds—an extreme case of unstructured data—and introduce a fast tree-based approximation to compute this. Since point clouds lack a well-defined winding number, they also introduce point areas for their adaptation. However, their formulation remains an approximation of the true winding number. Although effective in many applications, it lacks key theoretical properties of a true winding number. In this work, we propose a more accurate approximation that preserves these properties.

Normal-parametrized implicit fields. Recent work has improved normal estimation and surface reconstruction by directly optimizing normals via implicit fields parameterized by those normals. Iterative Poisson surface reconstruction (iPSR) [17] iteratively alternates between running PSR on the current normals and smoothing the implicit field computed by PSR to generate new normals, while diffusing winding gradients (DWG) [34] use the winding number field instead. Parametric Gauss reconstruction (PGR) [31] and winding number normal consistency (WNNC) [32] optimize for a winding number of $\frac{1}{2}$ at input surface points, with WNNC adding normal guidance. Globally consistent normal orientation (GCNO) [49] optimize for the winding number field to be peaked at zero and one and have high variance around input surface points. The boundary integral method (BIM) [35] optimizes the boundary energy derived from the Dirichlet energy of the winding number field. Stochastic normal orientation (SNO) [18] optimizes for the signed uncertainty of the stochastic PSR field to be minimized. We investigate the effect of our bounded winding number on GCNO and WNNC.

3. Preliminaries

The classic winding number counts the number of times a closed curve C in 2D encircles a query point q , defined as

$$w(q) = \frac{1}{2\pi} \oint_C d\theta(q) \quad (1)$$

where $\theta(q)$ is the signed angle traversed in relation to query point q , with positive angles indicating counterclockwise

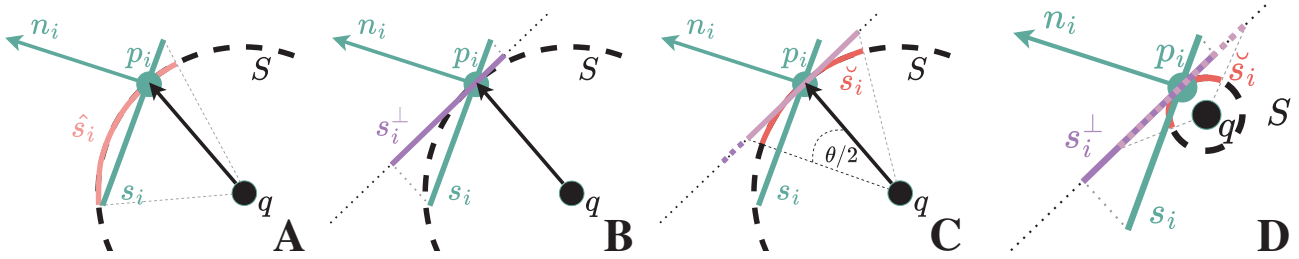


Figure 2. (A) In order to calculate the contribution of surface point p_i to the winding number at q we need the area of \hat{s}_i : the projection of the surface s_i at point p_i onto the sphere S . This is then normalized by the surface area of S to get the solid angle. (B) Eq. (4) approximates this by the area of s_i^\perp , which is the projection of s_i onto the tangent plane of S at p_i . (C) We instead approximate it by the area of \tilde{s}_i : we first project s_i onto the tangent plane, and then assume that this projected surface is distributed symmetrically around p_i and finally project that onto S , given by Eq. (12). (D) When the distance $\|p_i - q\|$ is small compared to the area of s_i^\perp , the latter can be much larger than the theoretical maximum: half of the surface area of S . Our approximation \tilde{a}_i is properly bounded by this.

rotation. It is commonly used to solve the point-in-polygon problem [15], *i.e.*, whether a point lies inside a polygon.

In order to determine whether a point lies inside a 3D triangle mesh M , Jacobson et al. [23] generalize the winding number to 3D by replacing the signed angle with the signed solid angle. Note that the angle (in radians) of a line segment at a query point $q \in \mathbb{R}^2$ can be defined by projecting it onto the unit circle around q and measuring the length of the projection. Similarly, the solid angle (in steradians) of a triangle t at a query point $q \in \mathbb{R}^3$, $\Omega_t(q)$, is the area of its projection onto the unit sphere around q . The solid angle is then signed based on whether the triangle's face normal points towards or away from the query point. The generalized winding number is then defined as

$$w(q) = \frac{1}{4\pi} \sum_{i=1}^N \Omega_{f_i}(q), \quad (2)$$

where the mesh has N faces $\{f_i\}_{i=1}^N$. If q is inside the mesh then $w(q) = 1$, otherwise $w(q) = 0$.

Barill et al. [4] extend this formulation to point clouds. They start with the integral definition of the generalized winding number and reformulate it as

$$w(q) := \frac{1}{4\pi} \int_M d\Omega(q) = \int_M \frac{(x-q)^\top n_x}{4\pi \|x-q\|_2^3} dx \quad (3)$$

where n_x denotes the unit normal at x . They then discretize this to define the winding number for a point cloud in 3D with N points p_i and unit normals n_i

$$w_U(q) := \sum_{i=1}^N a_i \frac{(p_i - q)^\top n_i}{4\pi \|p_i - q\|_2^3} \quad (4)$$

where a_i is the geodesic Voronoi area associated with point p_i on the surface of the shape. Barill et al. approximate this by fitting a plane to p_i and its k nearest neighbors, projecting the $k+1$ points onto the plane, and computing the Voronoi area associated with the projection of p_i .

We introduce notation for an individual term in the winding number formula, which will be used throughout:

$$w_U(q) = \sum_{i=1}^N w_U(q; p_i, n_i, a_i) \quad (5)$$

$$w_U(q; p_i, n_i, a_i) := a_i \frac{(p_i - q)^\top n_i}{4\pi \|p_i - q\|_2^3}, \quad (6)$$

where $w_U(q; p_i, n_i, a_i)$ denotes the contribution from a single surface point p_i with normal n_i and point area a_i to the winding number at query point q , $w_U(q)$.

4. A Bounded Point Cloud Winding Number

The winding number term $w_U(q; p_i, n_i, a_i)$ should represent the solid angle of area a_i at the query point q , signed by the orientation of the normal vector n_i . The area a_i corresponds to an unknown surface s_i lying in a plane orthogonal to n_i that contains the point p_i (see Fig. 2 A). To calculate the solid angle, we project s_i onto a unit sphere centered at q and compute the area of the projection. Equivalently, as shown in Fig. 2 A, we can project s_i onto the sphere S centered at q that passes through p_i , producing \hat{s}_i , and divide the projected area by the surface area of S , $4\pi \|p_i - q\|_2^2$.

To better understand Eq. (6), we rearrange it as

$$w_U(q; p_i, n_i, a_i) = \frac{a_i^\perp}{4\pi \|p_i - q\|_2^2}, \quad (7)$$

where $a_i^\perp := a_i ((\widehat{p_i - q})^\top n_i)$. Here, a_i^\perp is the signed area of s_i^\perp , the projection of s_i onto the tangent plane of S at p_i (see Fig. 2 B). This is because the angle between s_i and s_i^\perp is the same as the angle between n_i and $p_i - q$. The sign is given by the relative direction of n_i to $p_i - q$. Note that this area is trying to approximate the area of \hat{s}_i , it assumes that the tangent plane is a good approximation of the sphere.

As a_i^\perp approaches zero, this approximation becomes more accurate, hence its valid use in the integral definition

in Eq. (3). However, when q is very close to p_i relative to a_i^\perp , as shown in Fig. 2 D, the approximation becomes highly inaccurate. In such cases, a_i^\perp can exceed half the surface area of S , which is the upper bound of the projected area from a flat surface. In fact, as $q \rightarrow p_i$, $|a_i^\perp| \rightarrow \infty$. This is demonstrated in Fig. 1, where the winding number at query points close to p_i are far outside $[0, 1]$. This causes problems in optimization (see Fig. 5).

Ideally we would compute the true projection \hat{s}_i , but this requires knowing how s_i is distributed around p_i . One can make the assumption that s_i is symmetric around p_i , but then projecting this onto S is difficult to compute efficiently.

Our solution involves a two-step projection approach, shown in Fig. 2 C. First, we project onto the tangent plane as before. Then, we project this onto the sphere S , making the assumption that the projected surface s^\perp is symmetric around p_i . Note that the second projection ensures that the result is bounded by half the surface area of S . We denote this bounded winding number formulation as w_B , which we now derive (the derivation in 2D is given in the supplement).

Define a_i^\perp as the area of a circle centered at p_i on the tangent plane of S with radius $\sqrt{\pi^{-1}|a_i^\perp|}$. This circle, made to have area $|a_i^\perp|$, subtends an angle θ_i at q , related by

$$t_i := \tan\left(\frac{\theta_i}{2}\right) = \frac{\sqrt{a_i^\perp}}{\sqrt{\pi}\|p_i - q\|_2}. \quad (8)$$

We compute the signed angle θ_i by retaining the sign of a_i^\perp , $\theta_i = 2 \mathbf{sign}(a_i^\perp) \arctan(t_i)$. Then the (signed) area of the projection of this circle onto S , \check{a}_i , is the surface area of the spherical cap with angle $\frac{\theta_i}{2}$. This is given by $2\pi r^2 (1 - \cos(\frac{\theta_i}{2}))$ where r is the radius of the sphere. Retaining the sign again,

$$\check{a}_i = 2 \mathbf{sign}(a_i^\perp) \pi \|p_i - q\|_2^2 \left(1 - \frac{1}{\sqrt{t_i^2 + 1}}\right). \quad (9)$$

Thus, the bounded winding number becomes:

$$w_B(q) = \sum_{i=1}^N w_B(q; p_i, n_i, a_i) \quad (10)$$

$$w_B(q; p_i, n_i, a_i) := \frac{\check{a}_i}{4\pi \|p_i - q\|_2^2} \quad (11)$$

$$= \frac{\mathbf{sign}(a_i^\perp)}{2} \left(1 - \frac{1}{\sqrt{t_i^2 + 1}}\right). \quad (12)$$

4.1. Simplification and Analysis

We can further simplify the formulation by noting that $\mathbf{sign}(w_U(q; p_i, n_i, a_i)) = \mathbf{sign}(a_i^\perp)$:

$$w_B(q; p_i, n_i, a_i) = c(w_U(q; p_i, n_i, a_i)) \quad (13)$$

$$c(x) := \frac{\mathbf{sign}(x)}{2} \left(1 - \frac{1}{\sqrt{|4x| + 1}}\right). \quad (14)$$

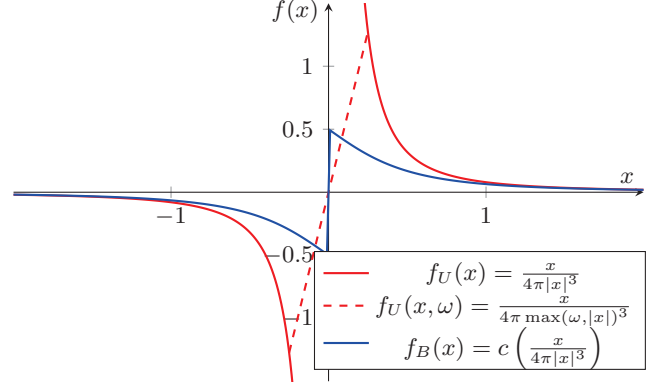


Figure 3. Plot of the signed distance kernels $w_U(q; p_i, n_i, a_i)$ (red) and $w_B(q; p_i, n_i, a_i)$ (blue) defined in Sec. 4.1. The truncated variant $w_U(q, \omega; p_i, \mu_i)$ from PGR [31] and WNNC [32], defined in Sec. 4.2.2, is also plotted, for $\omega = 0.25$ (dashed red). The kernel used in existing work w_U diverges in the zero-distance limit, whereas ours w_B is bounded and attains the correct value.

Note that when $\|p - q\|_2$ limits to 0, $w_U(q; p_i, n_i, a_i)$ diverges to $\pm\infty$ making the inner fraction in Eq. (14) limit to 0, and thus $w_B(q; p_i, n_i, a_i)$ limit to $\pm\frac{1}{2}$.

We plot the distance kernel of $w_U(q; p_i, n_i, a_i)$ and $w_B(q; p_i, n_i, a_i)$ in Fig. 3, denoted as $f_U(x)$ and $f_B(x)$. To do this, we set $a_i = 1$ and $q = p_i - xn_i$, $x \in \mathbb{R}$ so $p_i - q = xn_i$. As expected for a signed solid angle approximation, both functions are antisymmetric and decrease in magnitude as $|x|$ increases. However, as $x \rightarrow 0$, the solid angle should approach $\frac{1}{2}$, since when q is infinitesimally close to p_i , the latter's surface projects onto half of the unit sphere at q . Furthermore at $x = 0$, the signed solid angle should be 0. To see this, again assume q is infinitesimally close to p_i in its inside direction ($x > 0$). Then the contribution from p_i is $\frac{1}{2}$. Since it is inside the shape, its overall winding number must be 1, so the contribution from all other points should be $\frac{1}{2}$. Since moving from $x = \varepsilon$ to $x = 0$ does not affect contributions from other points, and the overall winding number at the surface should be $\frac{1}{2}$, p_i must have zero contribution at $x = 0$. These two properties only hold for $f_B(x)$, and the latter property is crucial for Sec. 4.2.2.

In Fig. 4, we evaluate the winding number on evenly sampled point clouds of a circle in 2D and a sphere in 3D at dense grid locations, using ground-truth normals and areas. Both $w_U(q)$ and $w_B(q)$ have modes at zero and one (note the log y scale), but $w_U(q)$ has many outliers outside of $[0, 1]$ (full range given in legend). As sampling becomes denser, the number and range of outliers decrease, but extreme outliers remain for the dense 3D case (100k points). This makes sense as the projection approximation is much worse in 3D. In contrast, $w_B(q)$ has few outliers in both 2D and 3D, and has better convergence to an indicator function as the sampling density increases.

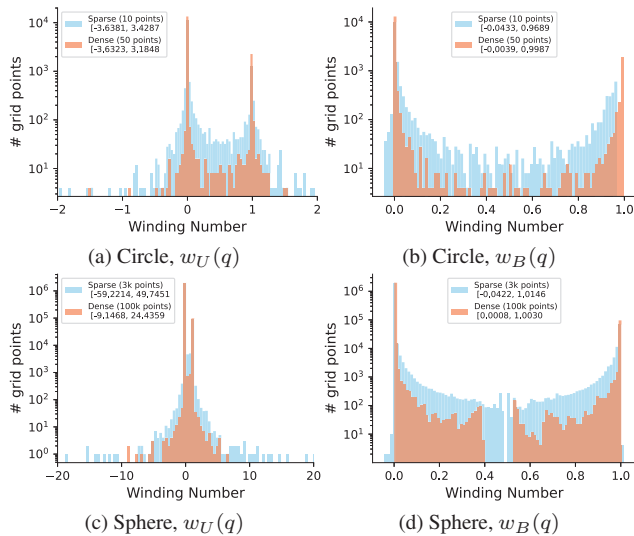


Figure 4. Histograms of the winding number field values of a circle in 2D (top row) and of a sphere in 3D (bottom row) with a logarithmic scale. The shapes are sampled evenly both sparsely and densely, and the ground-truth point normals and areas are used. We query the field in a regular grid of the domain. For $w_U(q)$ we truncate the x-axis, but give the full range of the values in the legend. The corresponding 2D fields are visualized in Fig. 1.

4.2. Normal Estimation with Winding Numbers

We now examine two normal estimation approaches that use point cloud winding numbers and analyze how they mitigate the poor approximation of $w_U(q)$. We then show how we can integrate our bounded formulation $w_B(q)$.

4.2.1. GCNO

Xu et al. [49] propose globally consistent normal orientation (GCNO), a method for normal estimation by regularizing the winding number field. They represent normals in spherical coordinates, use precomputed point areas, and optimize the normals with L-BFGS.

A key part of their method is that they compute a Voronoi diagram of the input points, which is used throughout their method. In preprocessing, point areas are estimated from Voronoi cell cross-sections, and during optimization, winding numbers are queried only at Voronoi vertices.

Their objective function has three terms: (1) a 0–1 term encouraging winding numbers at the Voronoi vertices to be zero or one; (2) a balance term maximizing variance in winding numbers around each input point (ideally half zero, half one); and (3) an alignment term encouraging normals to align with a pole of the Voronoi cell, a common heuristic for surface normal estimation. More details, including equations, are in the supplement.

Their motivation for using Voronoi vertices is that the vertices are far away from the input points, and thus the winding number at those vertices is likely to be close to either zero or one as required for their first loss term. They

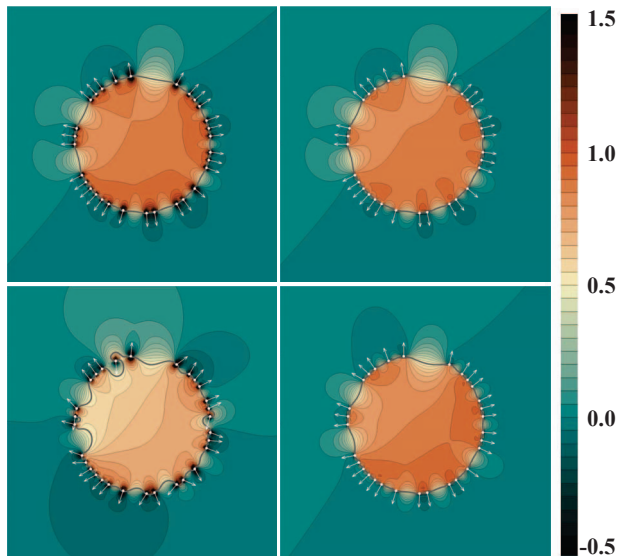


Figure 5. **(Top)** Comparing w_U (left) to w_B (right) for irregular samples from a sphere with ground-truth point normals and areas. Close to the surface points, w_U is significantly outside the range of $[0, 1]$, while w_B is not. **(Bottom)** We optimize the normals with sGCNO’s loss and dense grid samples as query points. Optimization with w_B (right) results in normals close to the ground truth, while optimization with w_U (left) fixates on reducing the loss terms arising from winding numbers outside $[0, 1]$, obtaining a poor normal estimation result.

observe that when there is slight input noise, the winding number at the Voronoi vertices still remains close to zero or one. Furthermore, they show that when sampling query points with Gaussian random noise around the input points, as usually done in the literature [12, 33], their optimization fails to produce consistent normals (their Fig. 23).

We argue that Voronoi sampling effectively mitigates the poor approximation of $w_U(q)$ close to the surface. Note that the Voronoi vertices are points that are equidistant to multiple input surface points, such that if you move in any direction you will be closer to one of those surface points. Thus, they are using points that are well distributed around each surface point, yet are maximally far from any surface point. We also demonstrate this through an experiment in which we sample densely near the input surface points, shown in Fig. 5. As shown in the top row, $w_U(q)$ has values far outside of $[0, 1]$. As a result, even when starting from ground-truth normals, optimizing with $w_U(q)$ leads to extreme normal direction changes (bottom left), especially for points with large areas, as they attempt to push those winding numbers toward zero or one. In contrast, optimizing with $w_B(q)$ preserves the ground-truth normals.

Stochastic GCNO (sGCNO). GCNO’s main issue is its runtime: 100–1000× slower than competitors. This arises from computing the Voronoi diagram, which is expensive

for large point clouds, and using L-BFGS, which is known to be slow due to requiring full batch optimization and repeated function evaluations for line search. We also hypothesize that relying only on Voronoi vertices as query points slows optimization as they are a sparse representation of the entire winding number field.

Encouraged by our results in Fig. 5, we propose stochastic GCNO (sGCNO), a stochastic approach based on GCNO that uses our improved approximation $w_B(q)$ and random sampling without a Voronoi diagram. We use stochastic gradient descent (SGD), and randomly sample query points at each iteration following common practice [3, 8, 12, 28, 46]. Note that replacing L-BFGS with SGD improves speed but risks poor local minima. However, the random sampling gives optimization a much denser representation of the entire winding number field. Point areas are estimated with a simple KNN scheme and optimized. We also make improvements to the loss terms from GCNO, which we explain in detail in the supplement.

4.2.2. PGR and WNNC

A different approach to optimizing the winding number field was proposed by Lin et al. [31], called Parametric Gauss Reconstruction (PGR), and was improved in their subsequent work Winding Number Normal Consistency (WNNC) [32]. Their approach leverages the fact that the winding number is equivalent to the Gauss formula from potential theory and should be $\frac{1}{2}$ on the surface. Thus, they optimize the winding number field by solving the system of equations where $w_U(p_i) = \frac{1}{2}$ for all input points p_i . When forming this system, they combine point areas and normals into a single parameter, area-weighted normals $\mu_j = a_j n_j$. Thus winding numbers are expressed as

$$w_U(q) = \sum_{j=1}^N w_U(q; p_j, \mu_j) \quad (15)$$

$$w_U(q; p_j, \mu_j) := \frac{(p_j - q)^\top \mu_j}{4\pi \|p_j - q\|_2^3}. \quad (16)$$

This leads to a linear system $A\mu = b$ where $A \in \mathbb{R}^{N \times 3N}$, $\mu \in \mathbb{R}^{3N}$ and $b \in \mathbb{R}^N$ are such that

$$A_{i,3j-2:3j} = \frac{(p_j - p_i)}{4\pi \|p_j - p_i\|_2^3} \quad (17)$$

$$\mu_{3j-2:3j} = \mu_j \quad (18)$$

$$b_i = \frac{1}{2}. \quad (19)$$

However, as shown in Fig. 3, $w_U(q)$ is undefined at input points, so they introduce a *smoothing width* ω that truncates small distances in the denominator, replacing $\|p_j - q\|_2^3$ with $\max(\omega, \|p_j - q\|_2)^3$. When this truncation is used, we will add ω as an input, e.g., $w_U(q, \omega; p_j, \mu_j)$, $w_U(q, \omega)$ and similarly for w_B . The effect of this truncation is shown in Fig. 3.

As explained in Sec. 4.1, it is crucial that this function pass through the origin so that the winding number at surface points can be $\frac{1}{2}$. However, the system is underdetermined and ill-conditioned, so PGR solves for a regularized least-norm solution using conjugate gradient steps on the GPU. This optimization is very efficient, but forming A explicitly is quadratic in the number of points, making it memory-intensive.

WNNC improves efficiency by using the fast approximation for winding numbers from Barill et al. [4], which approximates the sum over all N input points p_j using the Barnes–Hut treecode algorithm [5]. They thus replace the explicit matrix A with an operator $A(\mu, \omega)$, which is theoretically still linear but computationally faster and less memory intensive. They solve for $A(\mu, \omega) = b$ using steepest descent on the least squares formulation, which has a closed form for the optimal step size [42].

They further note that with ground-truth parameters μ , the negative gradient of the winding number field with respect to the query points evaluated at the input points aligns with the normals, i.e., $-\nabla_q w_U(q, \omega)|_{q=p_i} = \lambda_i n_i$ for some $\lambda_i > 0$. They call this the winding number normal consistency (WNNC) property, and propose WNNC steps

$$\mu_i = \|\mu_i\|_2 \frac{G(\mu, \omega)}{\|G(\mu, \omega)\|_2} \quad (20)$$

$$G(\mu, \omega)_{3i-2:3i} = -\nabla_q w_U(q, \omega)|_{q=p_i}. \quad (21)$$

This maintains the current area estimates $\|\mu_i\|_2$, but changes the direction to be the current field’s negative gradient direction. Due to the smoothing width ω , they zero out gradient terms in $G(\mu, \omega)$ that are affected by smoothing, so $\nabla_q w_U(q, \omega)$ is set to zero when $\|p_j - q\|_2 < \omega$.

They start with $\mu = \mathbf{0}$ and run for 40 iterations, where in each iteration they first do a steepest descent step and then do a WNNC step. An important part of their method is their schedule for the smoothing width ω . For multiple noise levels, they define an initial and final values of ω for the first and last iteration, and linearly interpolate between these two for intermediate iterations. For example, for zero noise they recommend $\omega \in \Omega_0 := [0.002, 0.016]$, and for low noise they recommend $\omega \in \Omega_1 := [0.01, 0.04]$. When WNNC uses a specific range for the smoothing width Ω , we denote it as WNNC (Ω).

Integrating our bounded formulation. Replacing the winding number terms from w_U to w_B converts the original system $A(\mu, \omega) = b$ to $A_B(\mu, \omega) = b$, and the negative gradient in the WNNC step from $G(\mu, \omega)$ to $G_B(\mu, \omega)$, where

$$A_B(\mu, \omega)_i = w_B(p_i, \omega) \quad (22)$$

$$G_B(\mu, \omega)_{3i-2:3i} = -\nabla_q w_B(q, \omega)|_{q=p_i}. \quad (23)$$

Note that we still apply the smoothing width ω , and evaluate it if it is needed with our bounded formulation in Sec. 5.3.

Method	PGP \uparrow	\angle RMSE \downarrow	CD \downarrow	Time (s)
iPSR [17]	99.702	15.826	0.173	<u>27.8</u>
PGR [31]	99.889	14.534	0.141	3.3
GCNO [49]	99.783	15.926	0.204	9123
WNNC (SD, A, G, Ω_0) [32]	98.700	43.259	0.224	0.4
WNNC (SD, A, G, Ω_1) [32]	<u>99.944</u>	14.516	0.143	0.4
sGCNO $w_U(q)$	49.191	98.836	4.499	32.0
sGCNO $w_B(q)$	99.913	12.772	<u>0.140</u>	32.0
WNNC (SD, A, G_B, Ω_0)	99.728	21.844	0.169	0.4
WNNC (SD, A, G_B, Ω_1)	99.954	<u>10.798</u>	0.144	0.4
WNNC (LM, $A_B, G_B, 0$)	99.880	12.515	0.139	<u>1.5</u>
WNNC (LM, A_B, G_B, Ω_0)	99.904	11.705	0.139	<u>1.5</u>
WNNC (LM, A_B, G_B, Ω_1)	<u>99.944</u>	10.158	<u>0.140</u>	<u>1.5</u>

Table 1. Normal estimation and surface reconstruction results on the GCNO [49] dataset of 18 shapes. We randomly sample 3k points from each shape. The chamfer distance is computed after surface reconstruction with SPSR [25]. SD denotes steepest descent and LM denotes Levenberg–Marquardt.

However, $A_B(\mu, \omega) = b$ is now a nonlinear system of equations, making steepest descent inefficient as we no longer have an analytical optimal step size. However, we can easily change the WNNC step to use $G_B(\mu, \omega)$. The analytical form of the gradient is provided in the supplement.

To solve the nonlinear system of equations, we use the Levenberg–Marquardt optimizer [42], which is memory-intensive since it requires the formation of an $N \times 3N$ Jacobian matrix. However, for small point clouds, it optimizes effectively, with comparable or better results than steepest descent on the initial linear system. It takes steps using

$$r \leftarrow A_B(\mu, \omega) - b \quad (24)$$

$$J_r \leftarrow \nabla_{\mu} r \quad (25)$$

$$\delta \leftarrow J_r^T (J_r J_r^T + \lambda \mathbf{diag}(J_r J_r^T))^{-1} r \quad (26)$$

$$\mu \leftarrow \mu - \delta. \quad (27)$$

Overall, for small point clouds we propose taking gradient steps on the nonlinear system formed from $w_B(q, \omega)$ using Levenberg–Marquardt, and taking WNNC steps using $w_B(q, \omega)$. We denote this method as WNNC (LM, A_B, G_B, Ω), where $\omega \in \Omega$ as with WNNC. For large point clouds, since Levenberg–Marquardt is too expensive we propose taking steepest descent steps on the original linear system formed from $w_U(q, \omega)$, and taking WNNC steps using $w_B(q, \omega)$, which we denote WNNC (SD, A, G_B, Ω).

5. Experiments

We evaluate the original winding number-based normal estimation methods and our modifications on two datasets. The first dataset, from GCNO [49], consists of 18 shapes of varying complexity, with 3k randomly sampled points per shape. The second dataset, Top20SION [28], contains 20 challenging shapes from ShapeNet (100k points per

Method	\angle RMSE \downarrow			IoU \uparrow		
	Mean	Median	Std	Mean	Median	Std
SAP [45]	-	-	-	0.7616	0.8872	0.2665
PG-SDF [28]	-	-	-	0.9153	0.9598	0.0906
iPSR [17]	<u>22.288</u>	<u>18.447</u>	14.447	0.9709	0.9836	0.0391
PGR [31]	Did not fit in GPU memory					
GCNO [49]	Did not finish within 12h					
WNNC (Ω_0)	22.518	19.703	11.076	<u>0.9723</u>	<u>0.9815</u>	0.0292
sGCNO $w_B(q)$	38.714	34.703	11.472	0.9460	0.9758	0.0804
WNNC (SD, G_B, Ω_0)	21.100	18.162	11.230	0.9724	<u>0.9820</u>	0.0284

Table 2. Surface reconstruction results on the Top20SION dataset, a subset of ShapeNet (100k points).

Method	$\Omega = 0$		$\Omega = \Omega_0$		$\Omega = \Omega_1$		Time (s)
	PGP	RSME	PGP	RSME	PGP	RSME	
SD A	87.754	60.750	99.019	28.344	98.989	25.155	0.4
SD A, G	90.657	63.122	98.835	42.180	99.930	14.516	0.4
SD A, G_B	90.970	61.289	99.728	21.844	<u>99.954</u>	10.798	0.4
LM A	99.389	28.321	93.500	49.658	86.430	65.732	1.5
LM A, G	99.800	16.084	99.630	16.892	99.909	12.070	1.5
LM A_B	99.307	27.946	99.152	27.198	97.056	30.761	1.5
LM A, G_B	99.907	11.633	99.907	11.284	99.957	10.073	1.5
LM A_B, G_B	<u>99.880</u>	<u>12.515</u>	<u>99.904</u>	<u>11.705</u>	<u>99.944</u>	<u>10.158</u>	1.5

Table 3. Ablation study on components of our WNNC variants on the GCNO [49] dataset of 18 shapes. We randomly sample 3k points from each shape. The chamfer distance is computed after surface reconstruction with SPSR. SD denotes steepest descent and LM denotes Levenberg–Marquardt.

shape), selected based on their SION metric, a measure of shape complexity. We report two normal estimation metrics, percentage of good points (PGP) and angular RMSE (\angle RMSE), and two surface reconstruction metrics, chamfer distance (d_C) and intersection over union (IoU). For all normal estimation methods, when computing surface reconstruction metrics (chamfer and IoU) we use SPSR [25] to reconstruct a mesh on the optimized normals. In the supplement, we provide full details on the datasets and metrics and present additional results and visualizations, including for corrupted inputs.

5.1. GCNO Dataset Results

Results are shown in Tab. 1. All methods perform well on this dataset, although WNNC requires adjusting its smoothing width setting from no noise ($\Omega = \Omega_0$) to low noise ($\Omega = \Omega_1$). Our sGCNO approach slightly outperforms GCNO in PGP and Chamfer Distance (CD) and shows a significant improvement in angular RMSE. As expected, when using $w_U(q)$ it fails to converge. We find that using our bounded formulation with WNNC—for the WNNC step alone (SD, A, G_B) and for both steps (LM, A_B, G_B)—achieves better results than the original WNNC. This is consistent across both width settings. We visualize the reconstructed shapes in Fig. 6.

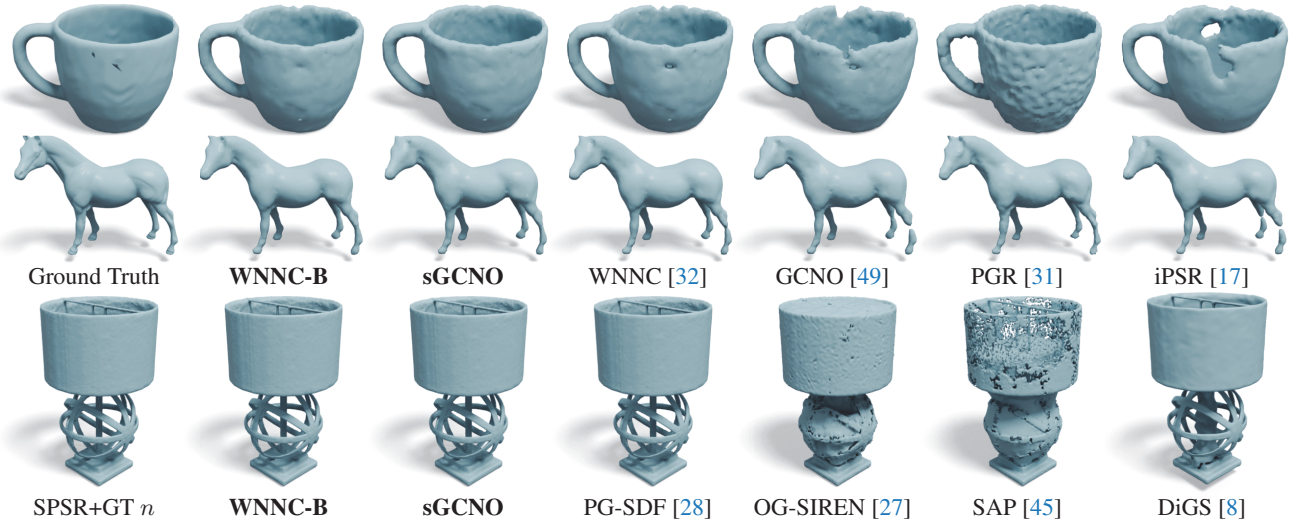


Figure 6. **(Top)** Comparison of normal estimation methods on two shapes from GCNO’s dataset. WNNC-B denotes our bounded formulation WNNC (LM, A_B , G_B , Ω_1); sGCNO denotes our stochastic bounded formulation of GCNO; and WNNC denotes WNNC (Ω_1). **(Bottom)** Comparison of surface reconstruction methods on a shape from Top20SION. WNNC-B denotes our bounded formulation with no smoothing WNNC (SD, A , G_B , Ω_0). SPSR+GT n denotes using SPSR with ground truth normals.

5.2. Top20SION Dataset Results

Since Top20SION consists of shapes that are particularly difficult for direct surface reconstruction, as shown in Tab. 2 top-performing direct reconstruction methods like SAP [45] and PG-SDF [28] achieve high median IoUs but lower mean IoUs indicating they struggle with a few shapes. Normal estimation methods that optimize for global consistency perform better, with both iPSR and WNNC performing consistently across all shapes, though PGR and GCNO fail to run within reasonable time or memory constraints. Our sGCNO approach successfully runs on these large point clouds but does not reach the performance level of iPSR or WNNC, suggesting that SGD is not able to optimize well for these difficult shapes. Our LM-based WNNC variant runs out of GPU memory on this dataset, but our SD-based variant (SD, A , G_B) achieves slightly better IoU and significantly better angular RMSE compared to WNNC. iPSR, which does not use a winding number field, gets comparative performance.

5.3. Ablation Study

We show ablations on using our bounded winding numbers in Tab. 3. Here we vary using different smoothing widths, whether steepest descent (SD) or Levenberg–Marquardt (LM) is used for the gradient steps, whether WNNC steps are used, and using $w_B(q)$ in the two steps. We use A and A_B to refer to the gradient step using $w_U(q)$ and $w_B(q)$, respectively, and G and G_B to refer to the WNNC step using $w_U(q)$ and $w_B(q)$, respectively.

One key finding is that the smoothing width ω , introduced to avoid singularities in w_U , also helps SD find better minima, especially improving WNNC steps. On the other

hand, LM converges to better minima, even with no smoothing, mostly likely because of its inbuilt damping providing better regularization benefits than smoothing.

When WNNC steps were not used, the best results with LM and were obtained without the smoothing width. In all cases, replacing $w_U(q)$ with $w_B(q)$ resulted in lower RMSE and equal or better PGP. With WNNC steps included, using $w_B(q)$ consistently led to significantly better results in both PGP and RMSE compared to $w_U(q)$. Interestingly, the hybrid approach where $w_U(q)$ was used for the gradient step while $w_B(q)$ was used for the WNNC step performed slightly better than using $w_B(q)$ in both steps.

6. Conclusion

In this work, we introduced a better, bounded approximation for point cloud winding numbers, $w_B(q)$, and demonstrated its benefits for normal estimation. Our formulation aligns with key theoretical properties, ensuring that it is bounded and attains the correct value at the limit $q \rightarrow p_i$. By integrating $w_B(q)$ into two existing methods, we showed that it enhances normal estimation performance.

We also provided insights into both of these methods. We demonstrated that the reason the use of Voronoi vertices in GCNO is crucial is due to the poor approximation of the original winding number formulation. We showed that while WNNC’s smoothing width regularization is essential when using steepest descent, Levenberg–Marquardt’s built-in regularization is sufficient. Additionally, we found that our improved approximation is not necessary for solving for consistent normal alignment within WNNC’s framework, but is valuable for reducing angular error.

Acknowledgments

Dr Campbell is the recipient of an Australian Research Council Discovery Early Career Award (project number DE250100542) funded by the Australian Government.

References

- [1] Pierre Alliez, David Cohen-Steiner, Yiyang Tong, and Mathieu Desbrun. Voronoi-based variational reconstruction of unoriented point sets. In *Symposium on Geometry processing*, pages 39–48, 2007. 2
- [2] Matan Atzmon and Yaron Lipman. SAL: Sign Agnostic Learning of shapes from raw data. In *CVPR*, 2020. 2
- [3] Ma Baorui, Han Zhizhong, Liu Yu-Shen, and Zwicker Matthias. Neural-pull: Learning signed distance functions from point clouds by learning to pull space onto surfaces. In *ICML*, 2021. 2, 6
- [4] Gavin Barill, Nia Dickson, Ryan Schmidt, David I.W. Levin, and Alec Jacobson. Fast winding numbers for soups and clouds. *ACM Trans. Graph.*, 2018. 2, 3, 6
- [5] Josh Barnes and Piet Hut. A hierarchical O(N log N) force-calculation algorithm. *Nature*, 1986. 6
- [6] Yizhak Ben-Shabat and Stephen Gould. DeepFit: 3D surface fitting via neural network weighted least squares. In *ECCV*, 2020. 2
- [7] Yizhak Ben-Shabat, Michael Lindenbaum, and Anath Fischer. Nesti-net: Normal estimation for unstructured 3D point clouds using convolutional neural networks. In *CVPR*, 2019. 2
- [8] Yizhak Ben-Shabat, Chamin Hewa Koneputugodage, and Stephen Gould. DiGS: Divergence guided shape implicit neural representation for unoriented point clouds. In *CVPR*, 2022. 2, 6, 8
- [9] Matthew Berger, Andrea Tagliasacchi, Lee M Seversky, Pierre Alliez, Gael Guennebaud, Joshua A Levine, Andrei Sharf, and Claudio T Silva. A survey of surface reconstruction from point clouds. In *Comput. Graph. Forum*, 2017. 1
- [10] Jonathan C Carr, Richard K Beatson, Jon B Cherrie, Tim J Mitchell, W Richard Fright, Bruce C McCallum, and Tim R Evans. Reconstruction and representation of 3D objects with radial basis functions. In *Proc. of the Conference on Computer Graphics and Interactive Techniques*, 2001. 2
- [11] Frédéric Cazals and Marc Pouget. Estimating Differential Quantities using Polynomial fitting of Osculating Jets. *Computer Aided Geometric Design*, 22(2):121–146, 2005. 2
- [12] Amos Groppe, Lior Yariv, Niv Haim, Matan Atzmon, and Yaron Lipman. Implicit Geometric Regularization for learning shapes. In *ICML*, 2020. 2, 5, 6
- [13] Gaël Guennebaud and Markus H. Gross. Algebraic point set surfaces. *ACM SIGGRAPH 2007 papers*, 2007. 2
- [14] Paul Guerrero, Yanir Kleiman, Maks Ovsjanikov, and Niloy J Mitra. PCPNet: Learning local shape properties from raw point clouds. In *Computer Graphics Forum*, 2018. 2
- [15] Paul S. Heckbert, editor. *Graphics gems IV*. Academic Press Professional, Inc., USA, 1994. 3
- [16] Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle. Surface reconstruction from unorganized points. *SIGGRAPH Comput. Graph.*, 26(2):71–78, 1992. 2
- [17] Fei Hou, Chiyu Wang, Wencheng Wang, Hong Qin, Chen Qian, and Ying He. Iterative poisson surface reconstruction (ipsr) for unoriented points. *ACM Trans. Graph.*, 41:1–13, 2022. 1, 2, 7, 8
- [18] Guojin Huang, Qing Fang, Zheng Zhang, Ligang Liu, and Xiao-Ming Fu. Stochastic normal orientation for point clouds. *ACM Trans. Graph.*, 2024. 2
- [19] Hui Huang, Dan Li, Hao Zhang, Uri M. Ascher, and Daniel Cohen-Or. Consolidation of unorganized point clouds for surface reconstruction. *ACM SIGGRAPH Asia 2009 papers*, 2009. 2
- [20] Hui Huang, Shihao Wu, Minglun Gong, Daniel Cohen-Or, Uri M. Ascher, and Hao Zhang. Edge-aware point set resampling. *ACM Transactions on Graphics (TOG)*, 32:1 – 12, 2013. 2
- [21] Zhiyang Huang, Nathan A. Carr, and Tao Ju. Variational implicit point set surfaces. *ACM Transactions on Graphics (TOG)*, 38:1 – 13, 2019. 2
- [22] Zhangjin Huang, Yuxin Wen, Zihao Wang, Jinjuan Ren, and Kui Jia. Surface reconstruction from point clouds: A survey and a benchmark. *arXiv:2205.02413*, 2022. 2
- [23] Alec Jacobson, Ladislav Kavan, and Olga Sorkine-Hornung. Robust inside-outside segmentation using generalized winding numbers. *ACM Transactions on Graphics (proceedings of ACM SIGGRAPH)*, 32(4):33:1–33:12, 2013. 2, 3
- [24] Krzysztof Jordan and Philippos Mordohai. A quantitative evaluation of surface normal estimation in point clouds. *IEEE International Conference on Intelligent Robots and Systems*, pages 4220–4226, 2014. 1
- [25] Michael Kazhdan and Hugues Hoppe. Screened poisson surface reconstruction. In *ACM TOG*, 2013. 2, 7
- [26] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *SGP*, 2006. 1, 2
- [27] Chamin Hewa Koneputugodage, Yizhak Ben-Shabat, and Stephen Gould. Octree guided unoriented surface reconstruction. In *CVPR*, 2023. 2, 8
- [28] Chamin Hewa Koneputugodage, Yizhak Ben-Shabat, Dylan Campbell, and Stephen Gould. Small steps and level sets: Fitting neural surface models with point guidance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21456–21465, 2024. 2, 6, 7, 8
- [29] Sören König and Stefan Gumhold. Consistent propagation of normal orientations in point clouds. In *International Symposium on Vision, Modeling, and Visualization*, 2009. 2
- [30] Jan Eric Lenssen, Christian Osendorfer, and Jonathan Masci. Deep iterative surface normal estimation. In *CVPR*, 2020. 2
- [31] Siyou Lin, Dong Xiao, Zuoqiang Shi, and Bin Wang. Surface reconstruction from point clouds without normals by parametrizing the gauss formula. *ACM Trans. Graph.*, 42, 2022. 1, 2, 4, 6, 7, 8
- [32] Siyou Lin, Zuoqiang Shi, and Yebin Liu. Fast and globally consistent normal orientation based on the winding number

- normal consistency. *ACM Trans. Graph.*, 2024. [1](#), [2](#), [4](#), [6](#), [7](#), [8](#)
- [33] David B. Lindell, Dave Van Veen, Jeong Joon Park, and Gordon Wetzstein. Bacon: Band-limited coordinate networks for multiscale scene representation. In *CVPR*, 2022. [5](#)
- [34] Weizhou Liu, Jiase Li, Xuhui Chen, Fei Hou, Shiqing Xin, Xingce Wang, Zhongke Wu, Chen Qian, and Ying He. Diffusing winding gradients (dwg): A parallel and scalable method for 3d reconstruction from unoriented point clouds, 2024. [2](#)
- [35] Weizhou Liu, Xingce Wang, Haichuan Zhao, Xingfei Xue, Zhongke Wu, Xuequan Lu, and Ying He. Consistent point orientation for manifold surfaces via boundary integration. In *ACM SIGGRAPH 2024 Conference Papers*, 2024. [2](#)
- [36] DanFeng Lu, HongKai Zhao, Ming Jiang, ShuLin Zhou, and Tie Zhou. A surface reconstruction method for highly noisy point clouds. In *International Workshop on Variational, Geometric, and Level Set Methods in Computer Vision*, pages 283–294. Springer, 2005. [2](#)
- [37] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3D reconstruction in function space. In *CVPR*, 2019. [2](#)
- [38] Gal Metzer, Rana Hanocka, Denis Zorin, Raja Giryes, Daniele Panozzo, and Daniel Cohen-Or. Orienting point clouds with dipole propagation. *ACM Trans. Graph.*, 40(4), 2021. [2](#)
- [39] Niloy Jyoti Mitra, An Thanh Nguyen, and Leonidas J. Guibas. Estimating surface normals in noisy point cloud data. *Int. J. Comput. Geom. Appl.*, 14:261–276, 2003. [2](#)
- [40] Patrick Mullen, Fernando Goes, Mathieu Desbrun, David Cohen-Steiner, and Pierre Alliez. Signing the unsigned: Robust surface reconstruction from raw pointsets. *Computer Graphics Forum*, 29, 2010. [2](#)
- [41] Yukie Nagai, Yutaka Ohtake, and Hiromasa Suzuki. Smoothing of partition of unity implicit surfaces for noise robust surface reconstruction. In *Comput. Graph. Forum*, 2009. [2](#)
- [42] Jorge Nocedal and Stephen J Wright. *Numerical optimization*. Springer, 1999. [2](#), [6](#), [7](#)
- [43] Yutaka Ohtake, Alexander Belyaev, and Marc Alexa. Sparse low-degree implicit surfaces with applications to high quality rendering, feature extraction, and smoothing. In *SGP*, 2005. [2](#)
- [44] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. DeepSDF: Learning continuous signed distance functions for shape representation. In *CVPR*, 2019. [1](#), [2](#)
- [45] Songyou Peng, Chiyu Jiang, Yiyi Liao, Michael Niemeyer, Marc Pollefeys, and Andreas Geiger. Shape As Points: A differentiable poisson solver. In *NeurIPS*, 2021. [2](#), [7](#), [8](#)
- [46] Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. In *NeurIPS*, 2020. [6](#)
- [47] Jun Wang, Zhouwang Yang, and Falai Chen. A variational model for normal computation of point clouds. *The Visual Computer*, 28:163–174, 2012. [2](#)
- [48] Hui Xie, Jianning Wang, Jing Hua, Hong Qin, and Arie E. Kaufman. Piecewise c1 continuous surface reconstruction of noisy point cloud via local implicit quadric regression. In *IEEE Visualization*, 2003. [2](#)
- [49] Rui Xu, Zhiyang Dou, Ningna Wang, Shiqing Xin, Shuangmin Chen, Mingyan Jiang, Xiaohu Guo, Wenping Wang, and Changhe Tu. Globally consistent normal orientation for point clouds by regularizing the winding-number field. *ACM Trans. Graph.*, 42(4), 2023. [1](#), [2](#), [5](#), [7](#), [8](#)