Fast Winding Numbers for Soups and Clouds

Gavin Barill
University of Toronto
gbarill@cs.toronto.edu

Neil Dickson
SideFX
ndickson@sidefx.com

Ryan Schmidt
Gradientspace
rms@rms80.com

David I.W. Levin
University of Toronto
di Levin@cs.toronto.edu

Alec Jacobson
University of Toronto
jacobson@cs.toronto.edu

Figure 1: In this paper, we further generalize the winding number to point clouds and propose a hierarchical algorithm for fast evaluation (up to 1000x speedup). This enables efficient answers to inside-outside queries for a wider class of shape representations (top) during a variety of tasks (bottom).

Abstract
Inside-outside determination is a basic building block for higher-level geometry processing operations. Generalized winding numbers provide a robust answer for triangle meshes, regardless of defects such as self-intersections, holes or degeneracies. In this paper, we further generalize the winding number to point clouds. Previous methods for evaluating the winding number are slow for completely disconnected surfaces, such as triangle soups or—in the extreme case—point clouds. We propose a tree-based algorithm to reduce the asymptotic complexity of generalized winding number computation, while closely approximating the exact value. Armed with a fast evaluation, we demonstrate the winding number in a variety of new applications: voxelization, signing distances, generating 3D printer paths, defect-tolerant mesh boolean operations and point set surfaces.

Index Terms: Computing Methodologies—Computer Graphics—Shape Modeling—Point-based models; Computing Methodologies—Computer Graphics—Shape Modeling—Mesh models;

1 Extended Abstract
Determining whether a point is inside or outside of a given shape is one of the most basic geometric questions. Inside-outside segmentation is crucial for: signing distance fields, tetrahedralizing or voxelizing volumes, representing smooth surfaces from point clouds, generating 3D printer path instructions, and surface repair (see figure 1). For analytic shapes and sufficiently clean discrete surface meshes, we can answer the question confidently and quickly. Unfortunately, most surface representations found in the wild are either completely unstructured (e.g., point clouds) or riddled with defects such as open boundaries, duplicated or degenerate geometry, self-intersections and non-manifold combinatorics.

The classic winding number determines how many times a planar curve encircles a query point (see, e.g., [3]). Generalized winding numbers [1] extend this concept to oriented triangle meshes suffering from the aforementioned defects. For oriented triangle meshes, this is computed as a sum of signed solid angles \( \Omega_t(q) \) of each triangle \( t \) subtended at a query point \( q \):

\[
w_S(q) = \frac{1}{4\pi} \sum_{t \in \text{Triangles}} \Omega_t(q). \tag{1}
\]

For closed watertight meshes, this perfectly reproduces the indicator function (1 inside, 0 outside). For overlapping regions, the winding number measures how many times the region is inside the surface. For holey or non-manifold surfaces, the winding number produces a smoothly varying function revealing a fractional measure of insideness. While simple and robust, a naive implementation of this definition: 1) is slow, requiring \( O(nm) \) computation for \( n \) queries and an \( m \)-triangle mesh; and 2) only applies to triangle meshes.

For large geometries and interactive applications, inside-outside queries need to be efficient. Existing optimizations for winding number computation either merely use parallelization or make heavy assumptions about mesh connectivity. For large, incoherent triangle “soups” often encountered during scanning or modeling existing methods are too slow.

Meanwhile, determining the smooth surface interpolating oriented point clouds is equivalent to extracting the boundary between what the points classify as inside or outside. Most existing point set surface methods are based on grid-dependent discretizations or custom tailored radial basis functions. These methods focus on level-set extraction, but knowing the answer to the inside-outside question has important applications away from the level set (e.g., for signed distances, voxelization, solid 3D printing, etc.).

In this work, we propose a fast method for computing generalized winding numbers on arbitrary triangle soups and point clouds. We begin by deriving a definition of the winding number for oriented point clouds. This directly enables a novel interpolating point set surface representation from the sum of winding number contributions from each point. This sum maintains physical units, avoiding parameter tuning or unnecessary blurring. We then asymptotically improve the performance of this sum with a tree-based algorithm for computing error-controlled approximations of far away points. Analytically integrating our definition for points leads to the familiar generalized winding number for triangles. By applying the same integration to our approximations, we generalize our fast evaluation algorithm to triangle soups as well.

We show evidence of the performance and approximation accuracy of our method on a large benchmark, where we achieve up to 1000 speedup for large triangle soups. We test our method in a variety of applications including: point set surfaces, voxelization, mesh cleanup, boolean operations on triangle soups, and signing distance fields. Quickly and robustly answering the inside-outside question allows raw point clouds and triangle soups to travel deeper into the geometry processing pipeline, avoiding lossy representation conversions.
To 3D print the shape represented by the winding field, we convert it to a stack of 2D polygons, which are then filled with toolpaths by the 3D printer software. We extract the polygons using “marching squares” \cite{2} again with a continuation approach. The 3D winding number is used for field evaluations – the 2D winding number along the slice is not the same for open geometry.

As a prototypical example, we show direct 3D printing of point clouds (see figure 2).

Our fast evaluation for triangle soups and point clouds will not only improve the performance of all applications already relying on generalized winding numbers, but opens the door to new opportunities. We foresee a rich topic of research exploring how to push raw, unstructured geometric data like soups and clouds farther along the geometry processing pipeline.

**REFERENCES**

