Robust Inside-Outside Segmentation using Generalized Winding Numbers

Alec Jacobson
Ladislav Kavan
Olga Sorkine-Hornung

ETH Zurich
University of Pennsylvania
ETH Zurich

October 9, 2013
Processing solid shapes requires volumetric representation

Input triangle mesh

Surface-based

Volume-based
Processing solid shapes requires volumetric representation

Input triangle mesh  Surface-based  Volume-based
Explicit representations are essential

triangle mesh
tetrahedral mesh
Explicit representations are essential

triangle mesh
watertight

tetrahedral mesh
made by TETGEN
Explicit representations are essential

triangle mesh \textit{watertight}

tetrahedral mesh \textit{made by TETGEN}

quality elements varying density conform to input
Apparent surface descriptions of solids are *unmeshable* with current tools.
Apparent surface descriptions of solids are *unmeshable* with current tools.

- Self-intersections
- Nonmanifold edges
- Multiple connected components
Apparent surface descriptions of solids are *unmeshable* with current tools.
Meshes are often output of human creativity

only 4000 vertices
Treating as scanned objects is inappropriate

only 4000 vertices

[Kazhdan et al. 2006] over 130000 vertices!
Treating as scanned objects is inappropriate only 4000 vertices

over 130000 vertices!

[Kazhdan et al. 2006]
Volume mesh should conform to input only 4000 vertices

our output tet mesh only 4500 vertices
Volume mesh should conform to input

Output vertices are superset of input: no data interpolation/extrapolation problems

Only 4000 vertices

Our output tet mesh
Only 4500 vertices
Can mesh the entire convex hull, but what’s inside? What’s outside?
Can mesh the entire convex hull, but what’s inside? What’s outside?
Can mesh the entire convex hull, but what’s inside? What’s outside?

open boundaries
Generalized function indicates *insideness*

open boundaries
Generalized function indicates *insideness*.

open boundaries
Generalized function indicates *insideness*
Generalized function indicates *insideness*

open boundaries
Generalized function indicates *insideness*
Generalized function indicates *insideness*

open boundaries
Function guides a crisp segmentation

open boundaries
Function guides a crisp segmentation
Function guides a crisp segmentation

open boundaries
Output is minimal, ripe for post-processing

Refined mesh using TETGEN, STELLAR, etc.

open boundaries
Idea: mesh entire convex hull, segment inside tets from outside ones
Idea: mesh entire convex hull, segment inside tets from outside ones

input triangles → CDT
Idea: mesh entire convex hull, segment inside tets from outside ones.

input triangles → CDT → measure of insideness
Idea: mesh entire convex hull, segment inside tets from outside ones

input triangles → CDT → measure of insideness → segment CDT
Idea: mesh entire convex hull, segment inside tets from outside ones

input triangles → CDT → measure of insideness → segment CDT → refine, etc.
Idea: mesh entire convex hull, segment inside tets from outside ones

input triangles → CDT → measure of insideness → segment CDT → refine, etc.
If shape is watertight, winding number is perfect measure of inside.

\[ w(p) = \frac{1}{2\pi} \oint_C d\theta \]
If shape is watertight, winding number is perfect measure of inside

\[ w(p) = \frac{1}{2\pi} \oint_C d\theta \]
Winding number uses orientation to treat insideness as signed integer

\[ w(p) = \frac{1}{2\pi} \oint_C d\theta \]
Naive discretization is simple and exact

\[ w(p) = \frac{1}{2\pi} \int_C d\theta \]

\[ w(p) = \frac{1}{2\pi} \sum_{i=1}^{n} \theta_i \]
Generalizes elegantly to 3D via solid angle

$$w(p) = \frac{1}{4\pi} \int \int_S \sin(\phi) d\theta d\phi$$

$$w(p) = \frac{1}{4\pi} \sum_{f=1}^{m} \Omega_f$$
What happens if the shape is open?

\[ w(p) = \frac{1}{2\pi} \oint_C d\theta \]
What happens if the shape is open?

\[ w(p) = \frac{1}{2\pi} \oint_C d\theta \]
What happens if the shape is open?

\[ w(p) = \frac{1}{2\pi} \oint_C d\theta \]
What happens if the shape is open?

\[ w(p) = \frac{1}{2\pi} \int_C d\theta \]
What happens if the shape is open?

\[ w(p) = \frac{1}{2\pi} \oint_{C} d\theta \]

Gracefully tends toward perfect indicator as shape tends towards watertight.
What if shape is self-intersecting? Non-manifold?

$$w(p) = \frac{1}{2\pi} \int_C d\theta$$

Jumps by ±1 across input facets
Winding number jumps across boundaries, otherwise harmonic!

\[ w(p) = \frac{1}{2\pi} \oint_C d\theta \]
Winding number jumps across boundaries, otherwise harmonic!

\[ w(p) = \frac{1}{2\pi} \oint_C d\theta \]

See MAPLE proof in paper or Rahul Narain’s recent proof http://goo.gl/5LJWf
Other interpolating implicit functions are confused by overlap...

[Shen et al. 2004]
...or resort to approximation

[Shen et al. 2004]
Sharp discontinuity across input eases precise, *conformal* segmentation
Sharp discontinuity across input eases precise, *conformal* segmentation
Sharp discontinuity across input eases precise, conformal segmentation
Sharp discontinuity across input eases precise, *conformal* segmentation
Sharp discontinuity across input eases precise, *conformal* segmentation
Sharp discontinuity across input eases precise, *conformal* segmentation
Sharp discontinuity across input eases precise, conformal segmentation
Sharp discontinuity across input eases precise, *conformal* segmentation
Winding number is sum of winding numbers: $O(m)$
Interesting fact reveals asymptotic speedup
Interesting fact reveals asymptotic speedup
Interesting fact reveals asymptotic speedup
Interesting fact reveals asymptotic speedup

\[ C \oplus \bar{C} = C \cup \bar{C} \]

\[ w_{C \cup \bar{C}}(p) = 0 \]
Interesting fact reveals asymptotic speedup

\[ w_C(p) + w_{\overline{C}}(p) = w_{C \cup \overline{C}}(p) = 0 \]
Interesting fact reveals asymptotic speedup

\[ w_C(p) = -w_{\bar{C}}(p) \]
Interesting fact reveals asymptotic speedup

\[ w_C(p) = -w_{\bar{C}}(p) \]
Interesting fact reveals asymptotic speedup

\[ w_C(p) = -w_{\bar{C}}(p) \]
Interesting fact reveals asymptotic speedup

\[ w_C(p) = -w_{\overline{C}}(p) \]
Interesting fact reveals asymptotic speedup

\[ w_C(p) = -w_{\bar{C}}(p) \]
Divide and conquer!
Divide and conquer!
Divide and conquer!
Divide and conquer!

\[ \sim m = \sim \frac{m}{2} - \sim \sqrt{\frac{m}{2}} \]
Divide and conquer!

\[ \sim m - \sim \frac{m}{2} = \sim \sqrt{\frac{m}{2}} \]
Divide and conquer!

\[ \sim m \]

= \[
\begin{array}{c}
\sim m \\
\sim \sqrt{m}
\end{array}
\]
Divide-and-conquer evaluation performs asymptotically better.
Divide-and-conquer evaluation performs asymptotically better
Divide-and-conquer evaluation performs asymptotically better

Winding number computation time (SHREC Dataset)

Seconds

1e-2
1e-3
1e-4

1e2
1e3
1e4
1e5

Naive
Hierarchical

Number of input facets, $m$

log-log plot!
Idea: mesh entire convex hull, segment inside tets from outside ones

input triangles → CDT → measure of insideness → segment CDT → refine, etc.
Segmentation is a labeling problem, labels should agree with w.n.

input triangles $\rightarrow$ CDT $\rightarrow$ measure of inside/outside $\rightarrow$ segment CDT $\rightarrow$ refine, etc.

graphcut energy optimization with nonlinear coherency term + optional facet or surface-manifoldness constraints
Preprocessing and meshing convex hull dominates runtime

input triangles $\rightarrow$ CDT $\rightarrow$ measure of insideness $\rightarrow$ segment CDT $\rightarrow$ refine, etc.

$60K$ facet mesh $\rightarrow$ $~22$ secs $\rightarrow$ $~10$ secs $\rightarrow$ $~0.1$ secs
Winding number degrades gracefully
CDT maintains small features

Open boundaries

Input triangle mesh

Winding number
We rely heavily on orientation

input mesh

backside of ear penetrates front (inside-out region)
We rely heavily on orientation

input mesh

backside of ear penetrates front (inside-out region)
We rely heavily on orientation

input mesh

our output
Brings a new level of robustness to volume meshing for a variety of shapes

http://goo.gl/m0oL9
Future work

- Even faster approximation
- Relationship to: diffusion curves, Mean Value Coordinates, etc.

![Diagram showing winding number and diffusion curves](image)
Acknowledgements


Marco Attene for MESHFIX

Hang Si for TETGEN

This work was supported in part by the ERC grant iModel (StG-2012-306877), by an SNF award 200021 137879 and the Intel Doctoral Fellowship.
Robust Inside-Outside Segmentation using Generalized Winding Numbers

http://igl.ethz.ch/projects/winding-number/
(paper, code, video)

Alec Jacobson
gleb@inf.ethz.ch
Ladislav Kavan
Olga Sorkine-Hornung

October 9, 2013
Additional material
Surface processing is distinct from volumetric distance.
Brings a new level of robustness to volume meshing for a variety of shapes
We rasterize the winding number, rather than ray cast.
We rasterize the winding number, rather than ray cast.
We rasterize the winding number, rather than ray cast.
We rasterize the winding number, rather than ray cast.
We rasterize the winding number, rather than ray cast.
We rasterize the winding number, rather than ray cast
We rasterize the winding number, rather than ray cast

63 rays
We rasterize the winding number, rather than ray cast
We rasterize the winding number, rather than ray cast

255 rays
We rasterize the winding number, rather than ray cast.
We rasterize the winding number, rather than ray cast.
We rasterize the winding number, rather than ray cast
Surface cleanup methods modify the input too much
Surface cleanup methods modify the input too much

[Attene 2010]
Winding number tells more than just inside: *how many times inside*
Winding number tells more than just inside: *how many times inside*
Duplicate any multiply inside parts: consistently overlapping tet mesh
Duplicate any multiply inside parts: consistently overlapping tet mesh
Some ambiguities are just semantics
Some ambiguities are just semantics
Some ambiguities are just semantics

(a)  
(b)  
(c)
Simple thresholding is not enough

\[ \text{is\_outside}(e_i) = \begin{cases} \text{true} & \text{if } w(e_i) < 0.5 \\ \text{false} & \text{otherwise} \end{cases} \]

Each element in CDT
Graphcut encourages coherency

\[ E = \sum_{i=1}^{m} \left[ u(x_i) + \gamma \frac{1}{2} \sum_{j \in N(i)} v(x_i, x_j) \right] \]

data coherency

winding number  threshold

October 9, 2013  Alec Jacobson  #114
Graphcut encourages coherency

\[ E = \sum_{i=1}^{m} \left[ u(x_i) + \gamma \frac{1}{2} \sum_{j \in N(i)} v(x_i, x_j) \right] \]

\[ u(x_i) = \begin{cases} 
  \max(w(e_i) - 0, 0) & \text{if } x_i = \text{outside} \\
  \max(1 - w(e_i), 0) & \text{otherwise}
\end{cases} \]
Graphcut encourages coherency

\[ E = \sum_{i=1}^{m} \left[ u(x_i) + \gamma \frac{1}{2} \sum_{j \in N(i)} v(x_i, x_j) \right] \]

\[ v(x_i, x_j) = \begin{cases} 
0 & \text{if } x_i = x_j \\
\alpha_{ij} \exp \left( \frac{|w(e_i) - w(e_j)|^2}{2\sigma^2} \right) & \text{otherwise}
\end{cases} \]
Graphcut encourages coherency

\[
E = \sum_{i=1}^{m} \left[ u(x_i) + \gamma \frac{1}{2} \sum_{j \in N(i)} v(x_i, x_j) \right]
\]

\[
\arg\min_{x \mid x_i \in [0,1]} E(x) \quad \text{use graphcut (maxflow)}
\]

winding number

threshold
Graphcut encourages coherency

$$E = \sum_{i=1}^{m} \left[ u(x_i) + \frac{\gamma}{2} \sum_{j \in N(i)} v(x_i, x_j) \right]$$

$$\arg\min_{x_i \in [0,1]} E(x) \quad \text{use graphcut (maxflow)}$$

subject to hard *facet constraints*
Graphcut encourages coherency

\[ E = \sum_{i=1}^{m} \left[ u(x_i) + \gamma \frac{1}{2} \sum_{j \in N(i)} v(x_i, x_j) \right] \]

\[ \text{argmin}_{x \mid x_i \in [0,1]} E(x) \]

subject to hard \textit{facet constraints}

“nonregular” [Kolmogorov & Zabin 2004]
Graphcut encourages coherency

\[ E = \sum_{i=1}^{m} \left[ u(x_i) + \gamma \frac{1}{2} \sum_{j \in N(i)} v(x_i, x_j) \right] \]

\[ \text{argmin}_{x|x_i \in [0,1]} E(x) \quad \text{use graphcut (maxflow)} \]

subject to hard *facet constraints*
Graphcut encourages coherency

\[
E = \sum_{i=1}^{m} \left[ u(x_i) + \gamma \frac{1}{2} \sum_{j \in N(i)} v(x_i, x_j) \right]
\]

\[
\begin{align*}
\text{argmin}_{x \mid x_i \in [0,1]} & \quad E(x) \\
\text{use graphcut (maxflow)}
\end{align*}
\]

subject to hard \textit{facet constraints}

\text{use heuristic $\rightarrow$ local min.}
Graphcut encourages coherency

\[
E = \sum_{i=1}^{m} \left[ u(x_i) + \frac{1}{2} \sum_{j \in N(i)} v(x_i, x_j) \right]
\]

\[
\arg\min_{x \mid x_i \in [0,1]} E(x) \quad \text{use graphcut (maxflow)}
\]

subject to hard \textit{facet constraints}

+subject to hard \textit{manifoldness constraints}
Hard constraints are optional: outliers
Even failure to create beautiful *surface*, can be success as volume representation.

*Input triangle mesh*
Even failure to create beautiful surface, can be success as volume representation
Even failure to create beautiful *surface*, can be success as volume representation.
Cleanup methods modify input too much, ...

input mesh  [Attene 2010]
Cleanup methods modify input too much, ...
... but we rely heavily on orientation

input mesh  [Attene 2010]  our output