Spectral Surface Reconstruction from Noisy Point Clouds

Ravi Kolluri
Jonathan Shewchuk
James O’Brien

Computer Science Division
University of California
Berkeley, California
Surface Reconstruction from 3D Point Clouds

Input:  Point cloud

Output:  Surface triangulation
Previous Work

Pioneers:

Boissonnat (1984)
Curless–Levoy (1996)

Implicit Surfaces:

Carr–Beatson–Cherrie–Mitchell–Fright–et al. (2001)

Delaunay:

Amenta–Choi–Kolluri “Powercrust” (2001)
Noise, Outliers, and Undersampling

Powercrust reconstruction of hand with outliers.

Tight Cocone reconstruction of Stanford Bunny with random noise in point coordinates.
Our Approach

Add bounding box
Our Approach  Form Delaunay triangulation
Our Approach

▲ : Inside  △ : Outside
Our Approach (Boissonnat 1984)

Output surface (Always watertight!)
Why Use Delaunay?
Why Use Delaunay?

- Effortless watertightness & outlier removal.
Why Use Delaunay?

- Effortless watertightness & outlier removal.
- Many Delaunay algorithms are provably correct.
Why Use Delaunay?

- Effortless watertightness & outlier removal.
- Many Delaunay algorithms are provably correct...in the absence of noise, outliers, undersampling.
Why Use Delaunay?

- Effortless watertightness & outlier removal.
- Many Delaunay algorithms are provably correct...in the absence of noise, outliers, undersampling.

Our Goal

- Achieve same results (in practice) as Cocone algorithm on “clean” point clouds; better results otherwise.
Why Use Delaunay?

- Because we can make it robust against noise, outliers, and undersampling.
Central Idea

Use spectral graph partitioning to decide which Delaunay tetrahedra are inside/outside the object.
Central Idea

Use spectral graph partitioning to decide which Delaunay tetrahedra are inside/outside the object.

And One Little Idea

Use negative edge weights to make the partitioner robust and fast.
The Partitioner Has a Global View

Inside or outside?
The Partitioner Has a Global View

Eigencrust reconstruction of undersampled hand.
The Partitioner Has a Global View

and can make better sense of outliers.
The Partitioner Has a Global View
Eigencrust
Some tetrahedra are easy to classify.

Some are ambiguous.

Obviously inside.

Could be labeled inside or outside.
Eigencrust Algorithm

Stage 1:

- Identify non-ambiguous tetrahedra called “poles”.
- Form a graph whose vertices are the poles.
- Assign edge weights based on geometry.
Eigencrust Algorithm

Stage 1:
- Identify non-ambiguous tetrahedra called “poles”.
- Form a graph whose vertices are the poles.
- Assign edge weights based on geometry.
- Partition graph.
Eigencrust Algorithm

Stage 1:
- Identify non-ambiguous tetrahedra called “poles”.
- Form a graph whose vertices are the poles.
- Assign edge weights based on geometry.
- Partition graph.

Stage 2:
- Form a graph whose vertices are the ambiguous tetrahedra (non-poles).
- Form graph, partition.
Voronoi Diagram
Voronoi Cell
Poles

Poles of a sample point are likely to be on opposite sides of surface. (Amenta–Bern 1999.)
Poles

Poles of a sample point are likely to be on opposite sides of surface.
Poles

Poles of a sample point are likely to be on opposite sides of surface.

Connect them with a negative-weight edge.
Poles

Weight of edge is 

\[ -e^{4 \pm 4 \cos \phi} \]
Negative weight edges
Positive weight edges
Positive Weight Edges

If two samples are connected by Delaunay edge, hook their poles together with positive weights.
Positive Weight Edges

Weight is large if the circumscribing spheres intersect deeply.

Weight of edge is $e^{4 - 4 \cos \phi}$
Pole graph
Supernode
Graph Partitioning with (Modified) Normalized Cuts

- Balances two criteria:
  - Minimizing sum of weights of cut edges.
  - Cutting graph into roughly “equal” pieces.
Graph Partitioning with (Modified) Normalized Cuts

- Balances two criteria:
  - Minimizing sum of weights of cut edges.
  - Cutting graph into roughly “equal” pieces.
- Pole Matrix $L$ is weighted adjacency matrix of pole graph.

\[
L = \begin{bmatrix}
a & b & c \\
-5 & -5 & 0 \\
0 & 6 & 6
\end{bmatrix}
\]
Graph Partitioning with (Modified) Normalized Cuts

- Balances two criteria:
  - Minimizing sum of weights of cut edges.
  - Cutting graph into roughly “equal” pieces.
- Pole Matrix $L$ is weighted adjacency matrix of pole graph. Diagonal $D$ of $L$ is the row sums of absolute off-diagonals.

\[
L = \begin{bmatrix}
  a & b & c \\
  5 & -5 & 0 \\
  -5 & 11 & 6 \\
  0 & 6 & 6 \\
\end{bmatrix}
\]
Graph Partitioning with (Modified) Normalized Cuts

- Balances two criteria:
  - Minimizing sum of weights of cut edges.
  - Cutting graph into roughly “equal” pieces.
- Pole Matrix $L$ is weighted adjacency matrix of pole graph. Diagonal $D$ of $L$ is the row sums of absolute off–diagonals.
- Compute eigenvector $x$ of $Lx = \lambda Dx$ with smallest eigenvalue (Lanczos iterations).

$$L = \begin{bmatrix}
5 & -5 & 0 \\
-5 & 11 & 6 \\
0 & 6 & 6 \\
\end{bmatrix} \quad x = \begin{bmatrix}
3 \\
2 \\
-4 \\
\end{bmatrix}$$
Graph Partitioning with (Modified) Normalized Cuts

- Balances two criteria:
  - Minimizing sum of weights of cut edges.
  - Cutting graph into roughly “equal” pieces.

- Pole Matrix $L$ is weighted adjacency matrix of pole graph. Diagonal $D$ of $L$ is the row sums of absolute off–diagonals.

- Compute eigenvector $x$ of $Lx = \lambda Dx$ with smallest eigenvalue (Lanczos iterations).

- Each component of $x$ corresponds to a pole/tetrahedron. Positive = inside; negative = outside.
End of Stage 1

\[ \triangle : \text{Inside} \quad \triangle : \text{Outside} \]

\[ \triangle : \text{Not pole} \]
Results
A Clean Point Cloud

327,323 input points.
654,496 output triangles.
2.8 minutes triangulation.
9.3 minutes eigenvectors.
A Noisy Point Cloud

362,272 input points.
679,360 output triangles.
1.5 minutes triangulation.
17.5 minutes eigenvectors.
Stanford Dragon

1,769,513 input points.
2,599,114 output triangles.
197 minutes.

Poles (tetrahedra)

+ Inside

- Outside
Artificial Outlier Test

- 200 outliers
- 1200 outliers
- 1800 outliers

- Eigencrust
- Powercrust
- Tight Cocone
Undersampled Goblet

2,714 input points.
36,538 output triangles.
1.63 seconds.
Conclusion

- Spectral partitioning is robust against noise, outliers, and undersampling.
- Handles raw data of real range finders.

Thanks

- Nina Amenta & Tamal Dey for their surface reconstruction programs.
- Chen Shen for rendering help.
- Stanford data repository.