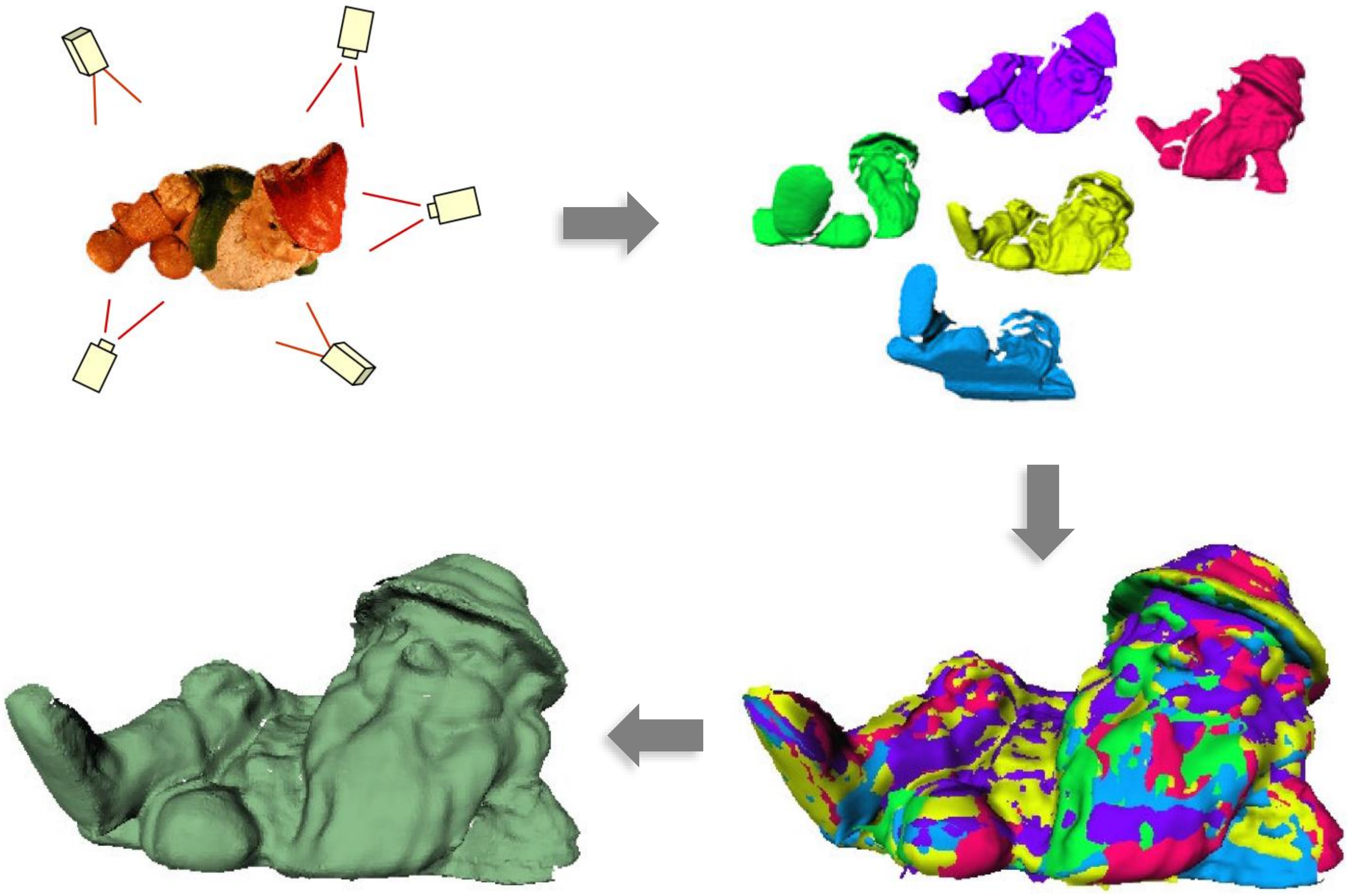
Multi-View Matching & Mesh Generation

Qixing Huang

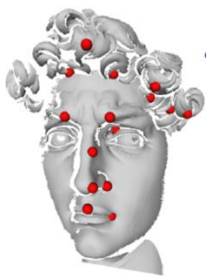
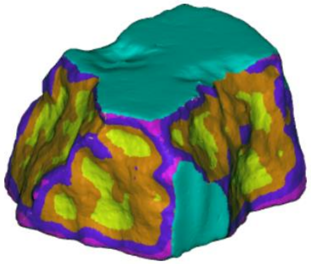
Feb. 13th 2017

Geometry Reconstruction Pipeline



RANSAC --- facts

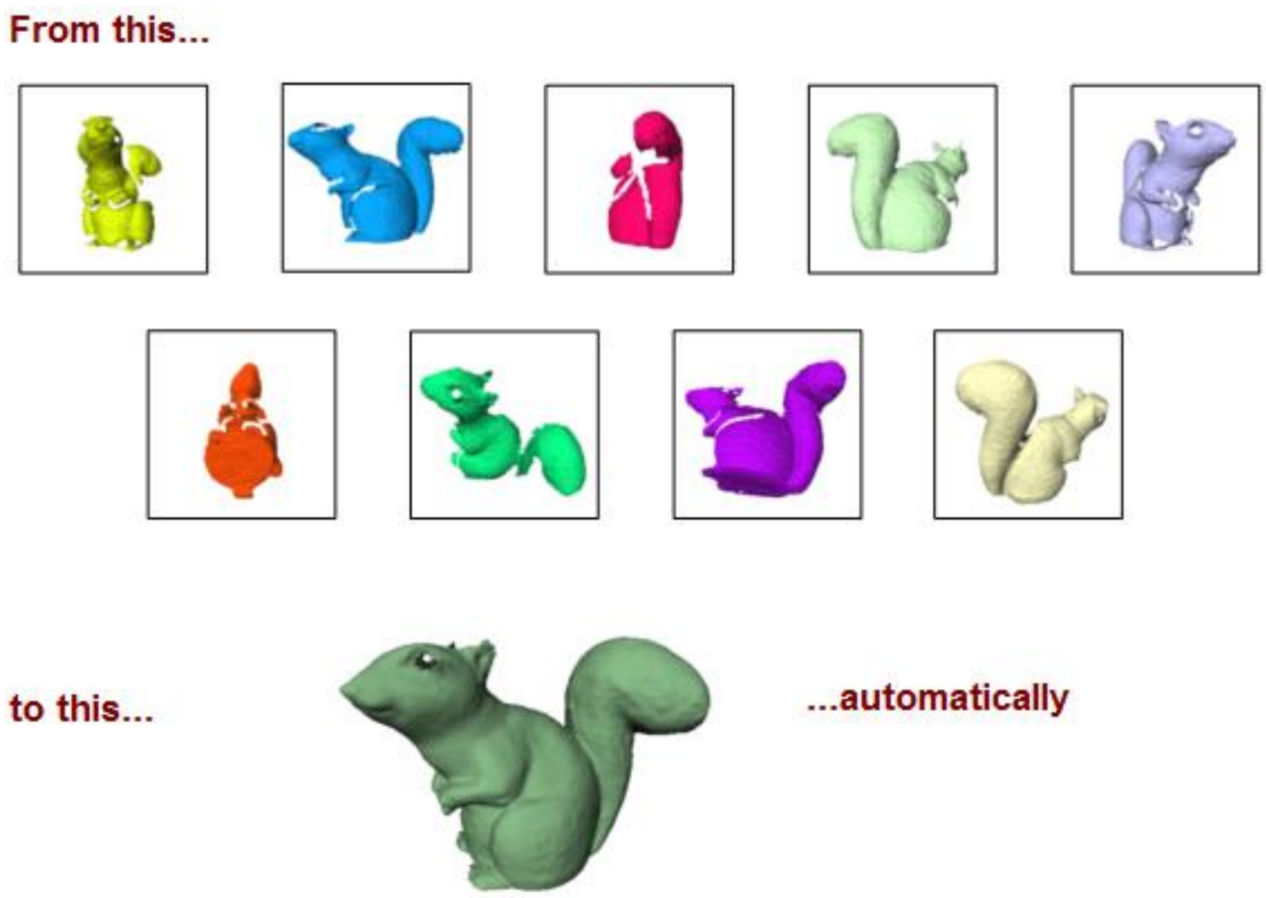
- Sampling
  - Feature point detection
    [Gelfand et al. 05, Huang et al. 06]

- Correspondences
  - Use feature descriptors $m << O(n^2)$
  - The candidate correspondences
  - Denote the success rate $p \approx \frac{n}{m}$
- *Basic* analysis
  - The probability of having a valid triplet p^3
  - The probability of having a valid triplet in N trials is $1-(1-p^3)^N$

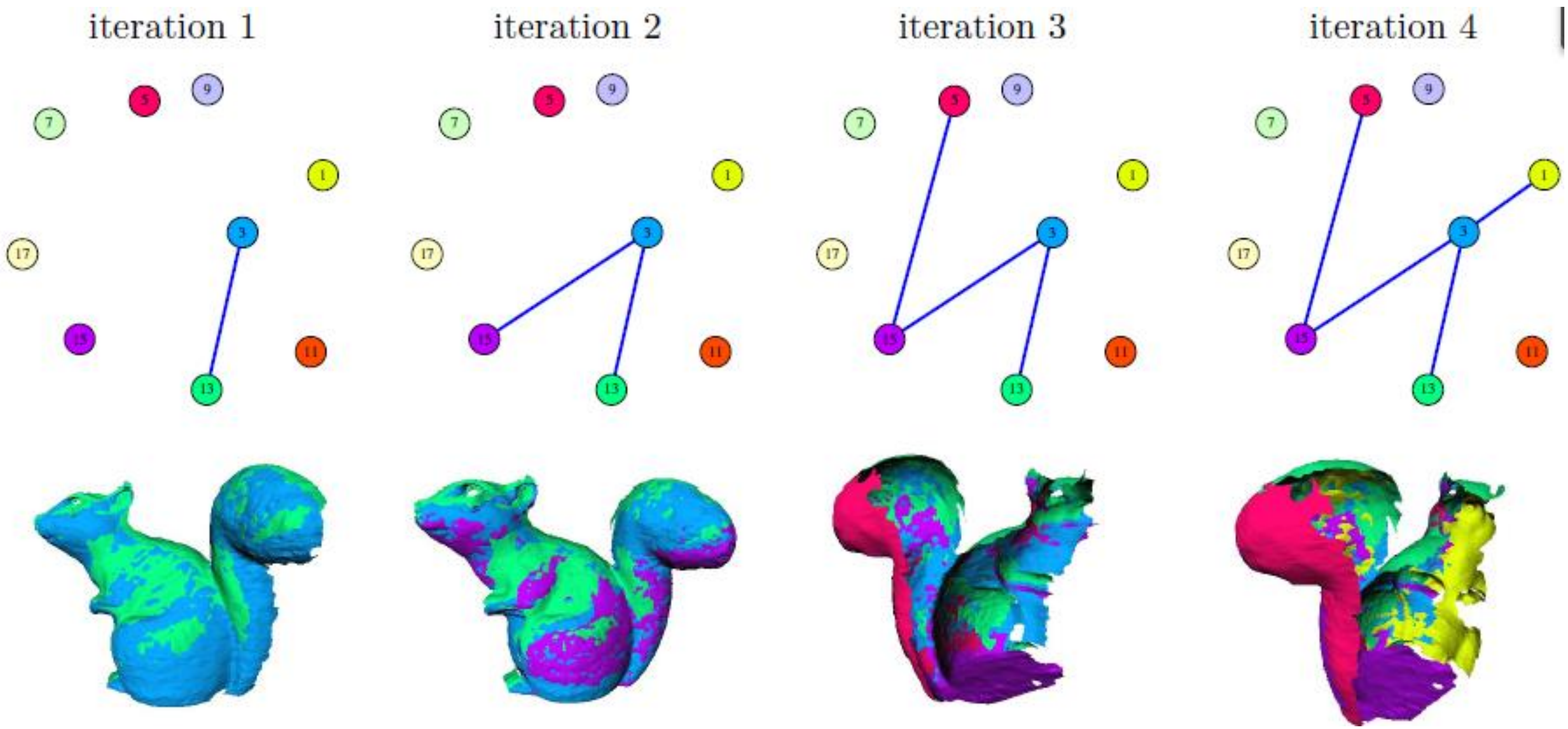
Multiple Matching

Spanning tree based

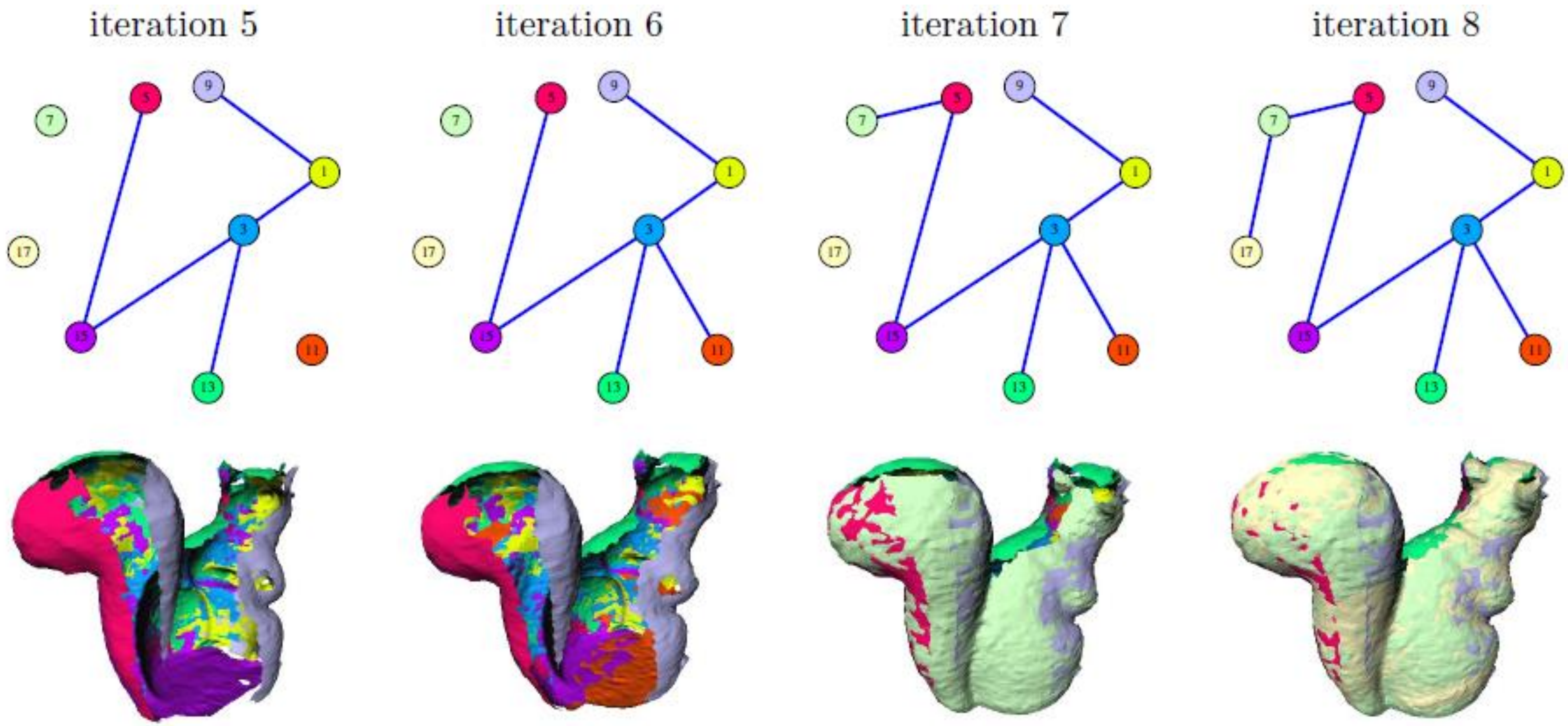


Automatic Three-dimensional Modeling from Reality, PhD thesis, D. Huber, Robotics Institute, Carnegie Mellon University, 2002

Spanning tree based

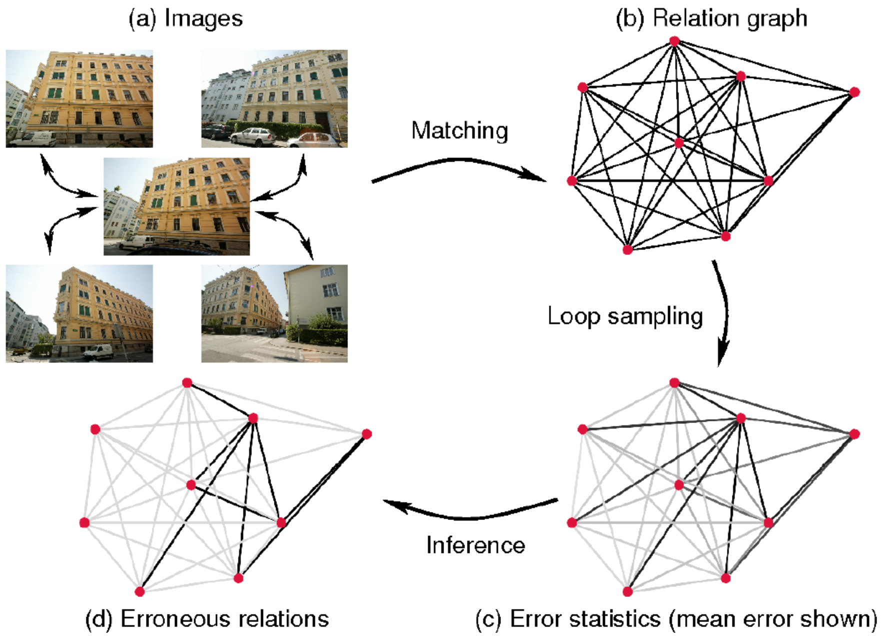


Spanning tree based



Issue: A single incorrect match can destroy everything

Detecting inconsistent cycles



large for inconsistent cycles

maximize $\sum_L \rho_L x_L$

subject to $x_L \geq x_e \; \forall e \in L,$

$x_L \leq \sum_{e \in L} x_e,$

$x_L \in [0, 1],$

$x_e \in [0, 1].$

Disambiguating visual relations using loop constraints, C. Zach, M. Klopscjotz, and M. POLLEFEYS, *CVPR'10*

Rotation

[Wang and Singer'13]

$$\mathbf{R} = \begin{bmatrix} I_3 & \cdots & \mathbf{R}_{1n} \\ \vdots & \ddots & \vdots \\ \mathbf{R}_{1n}^T & \cdots & I_3 \end{bmatrix}$$

minimize $\sum_{(i,j)\in\mathcal{G}} \|\mathbf{R}_{ij} - \mathbf{R}_{ij}^{init}\|_{\mathcal{F}}$

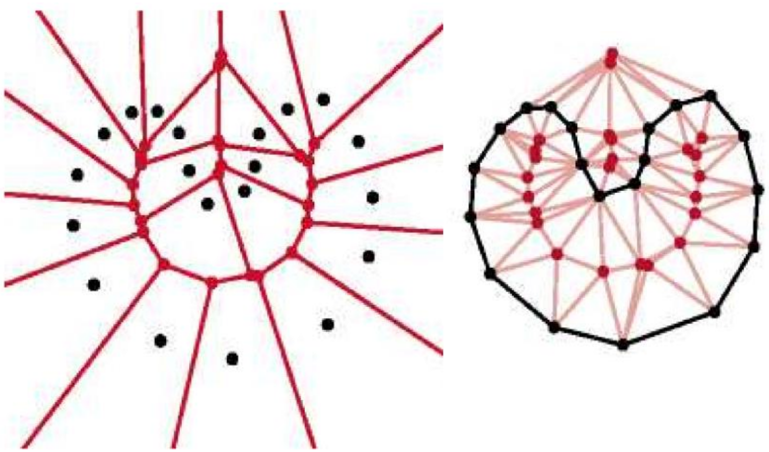
subject to $\mathbf{R} \succeq 0$

$$\mathbf{R}_{ii} = I_3, \quad 1 \leq i \leq n$$

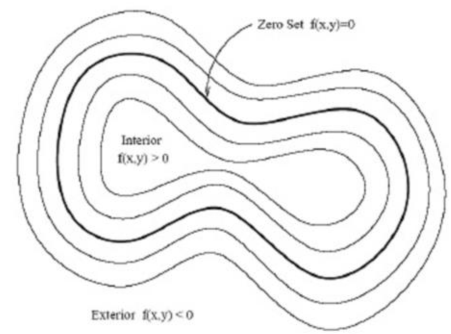
$$\mathbf{R}_{ij} \in convex - hull(SO(3)), \quad 1 \leq i < j \leq n$$

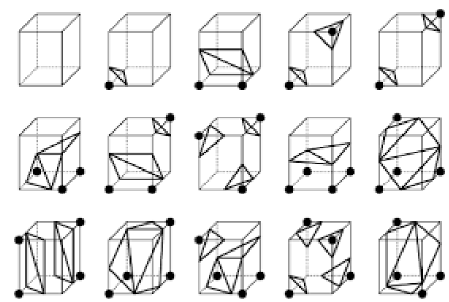
Meshing

Two Approaches



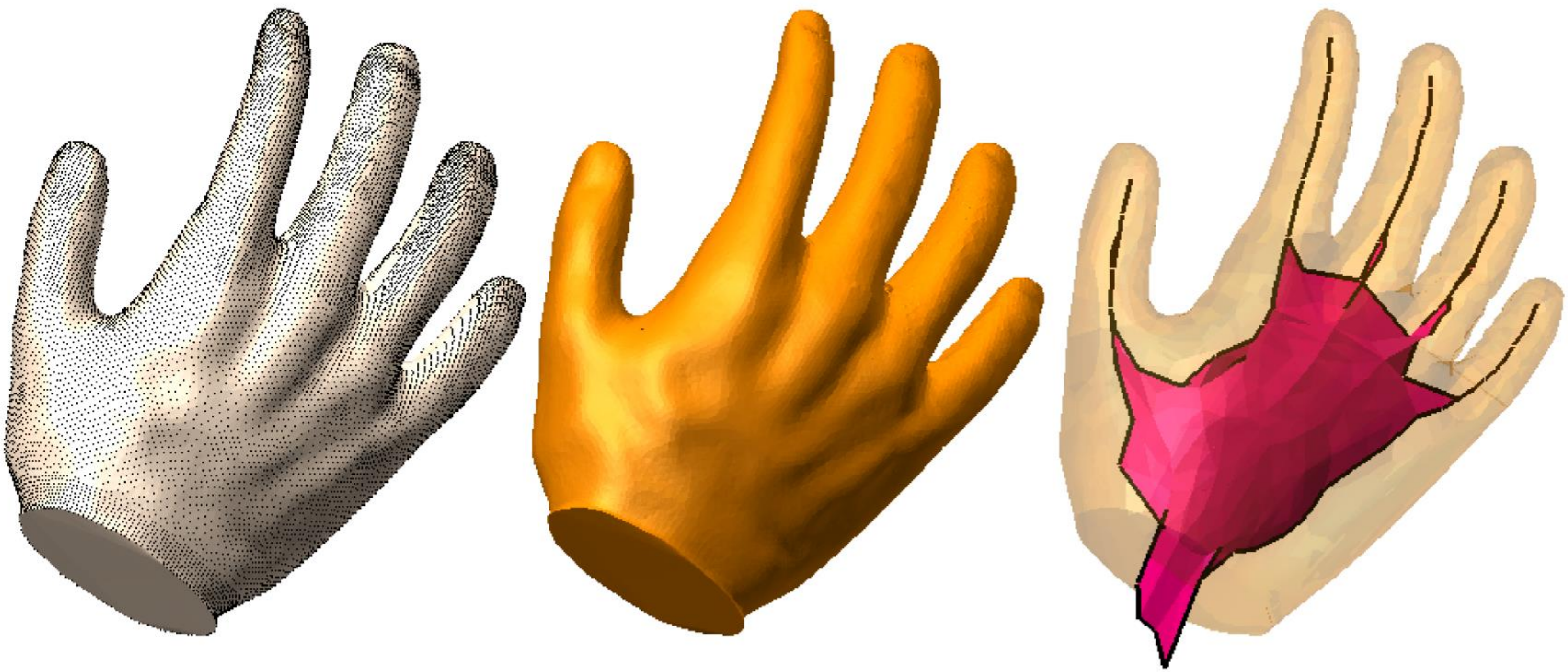
Computational Geometry Based



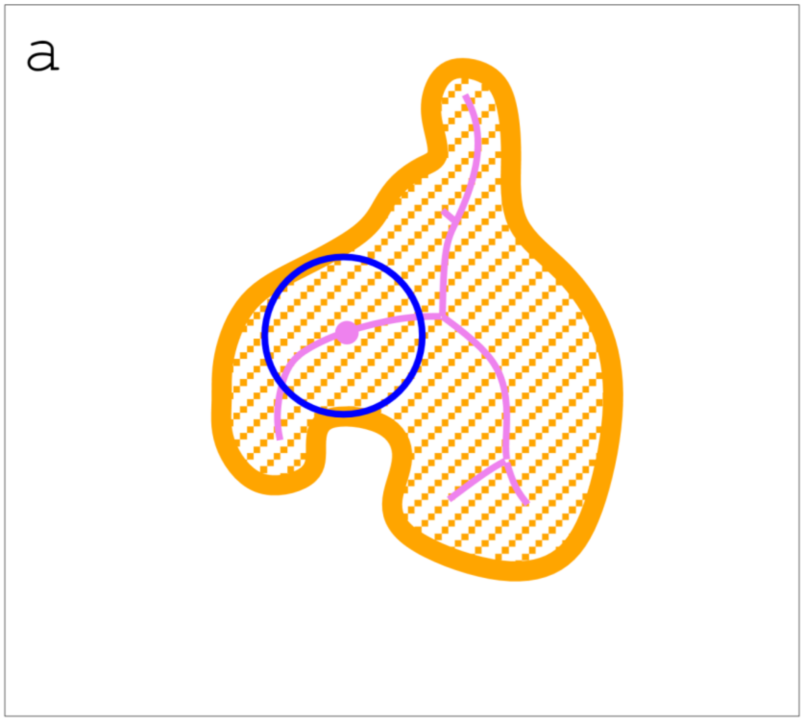


Implicit Surface -> Contouring

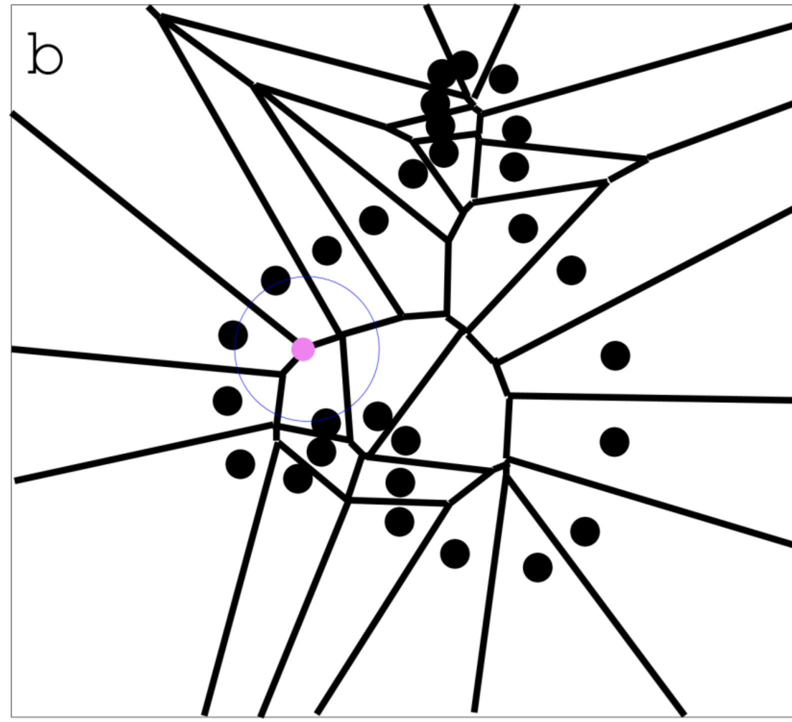
Power Crust [Amenta et al. 02]



Power Crust [Amenta et al. 02]

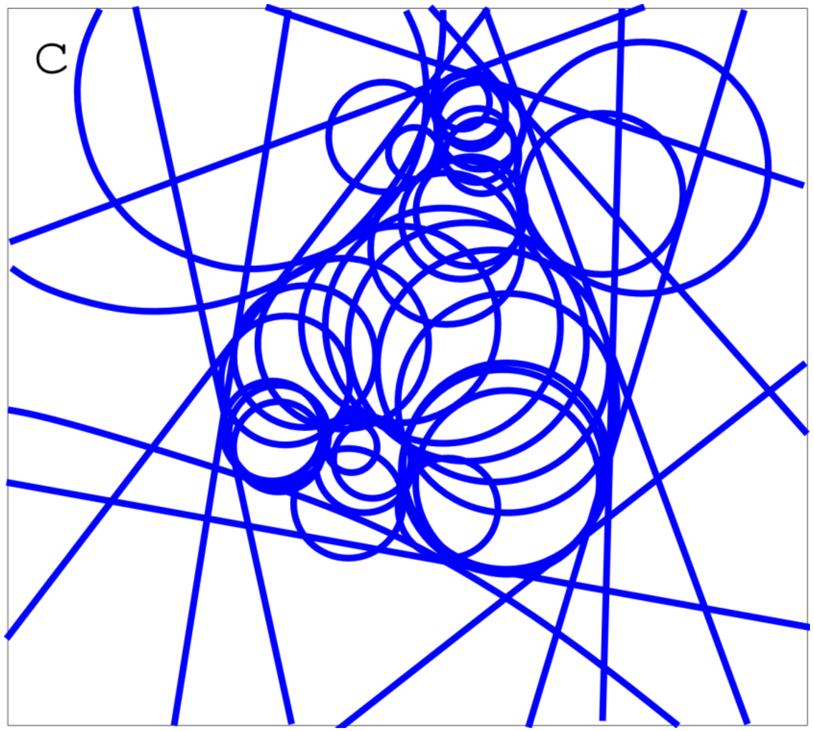


(a) An object with its medial axis; one maximal interior ball is shown.

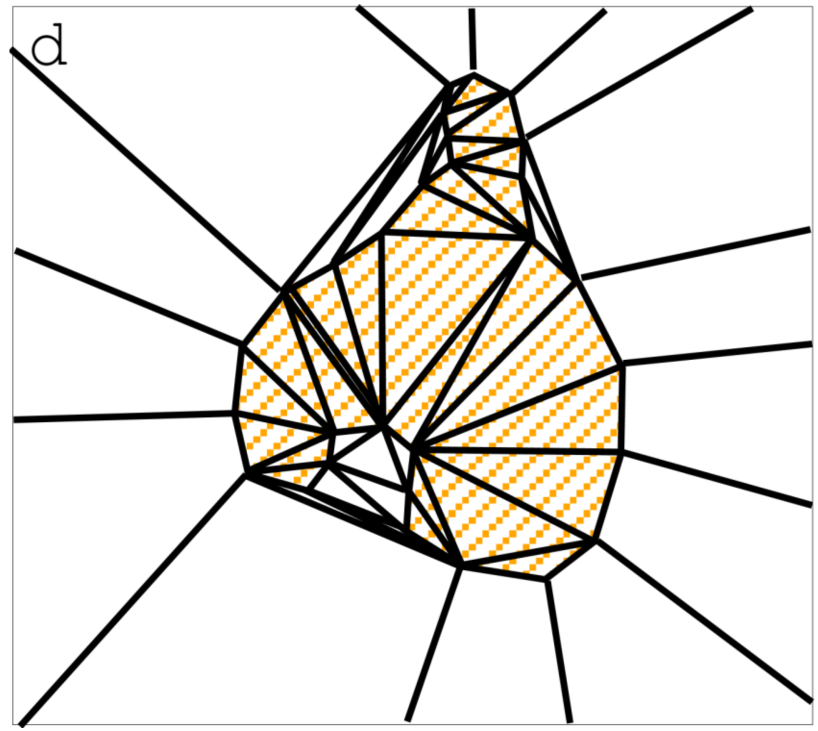


b) The Voronoi diagram of S, with the Voronoi ball surrounding one pole shown. In 2D, we can select all Voronoi vertices as poles, but not in 3D

Power Crust [Amenta et al. 02]

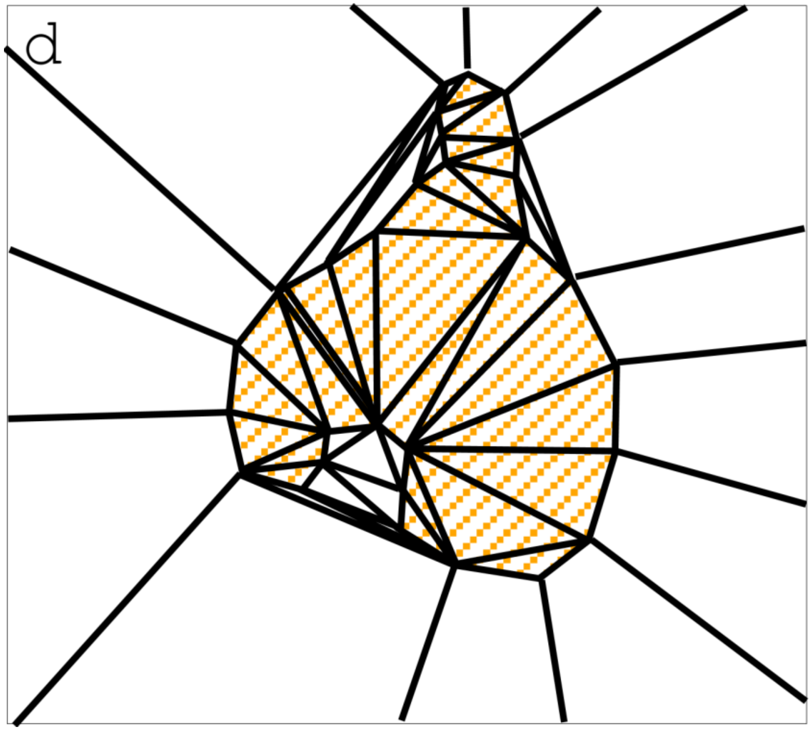


c) The inner and outer polar balls. Outer polar balls with centers at infinity degenerate to halfspaces on the convex hull.

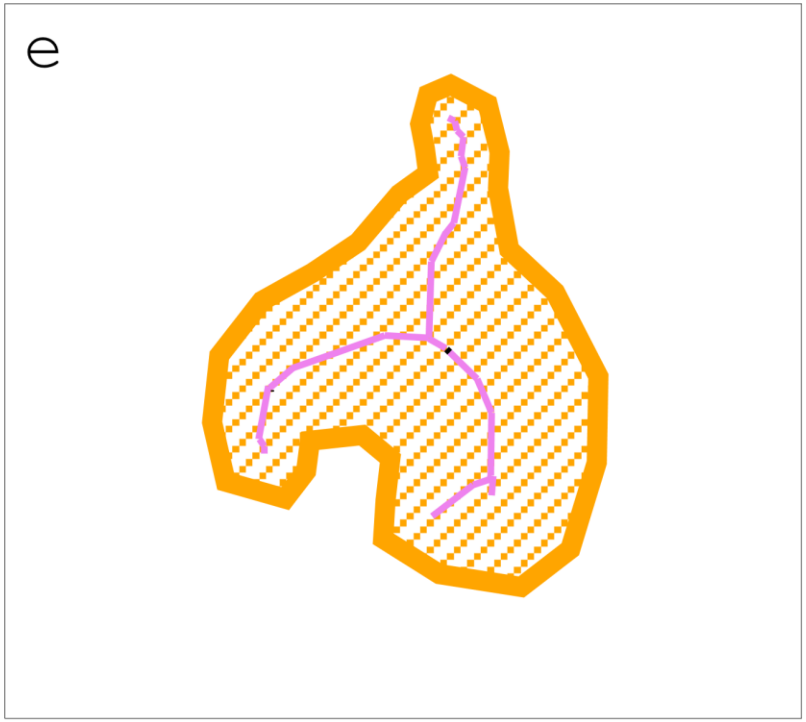


d) The power diagram cells of the poles, labeled inner and outer.

Power Crust [Amenta et al. 02]

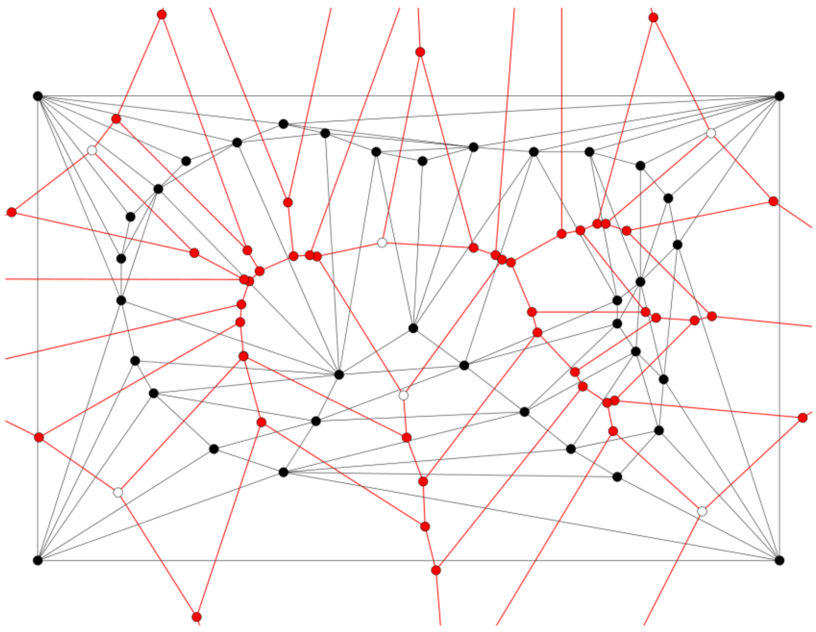


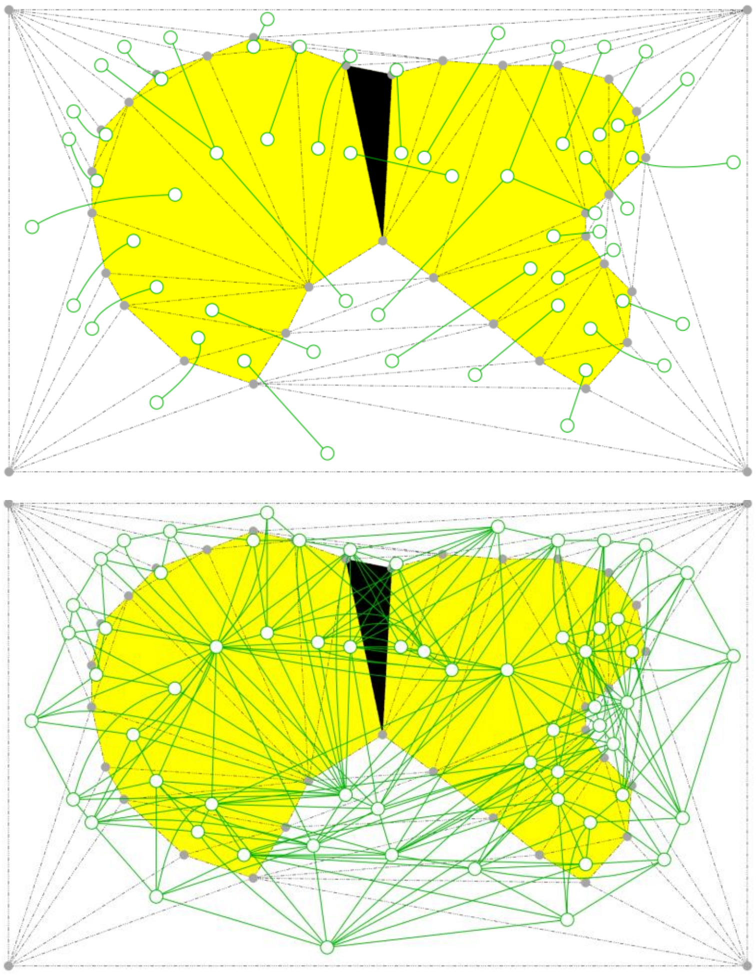
d) The power diagram cells of the poles, labeled inner and outer.



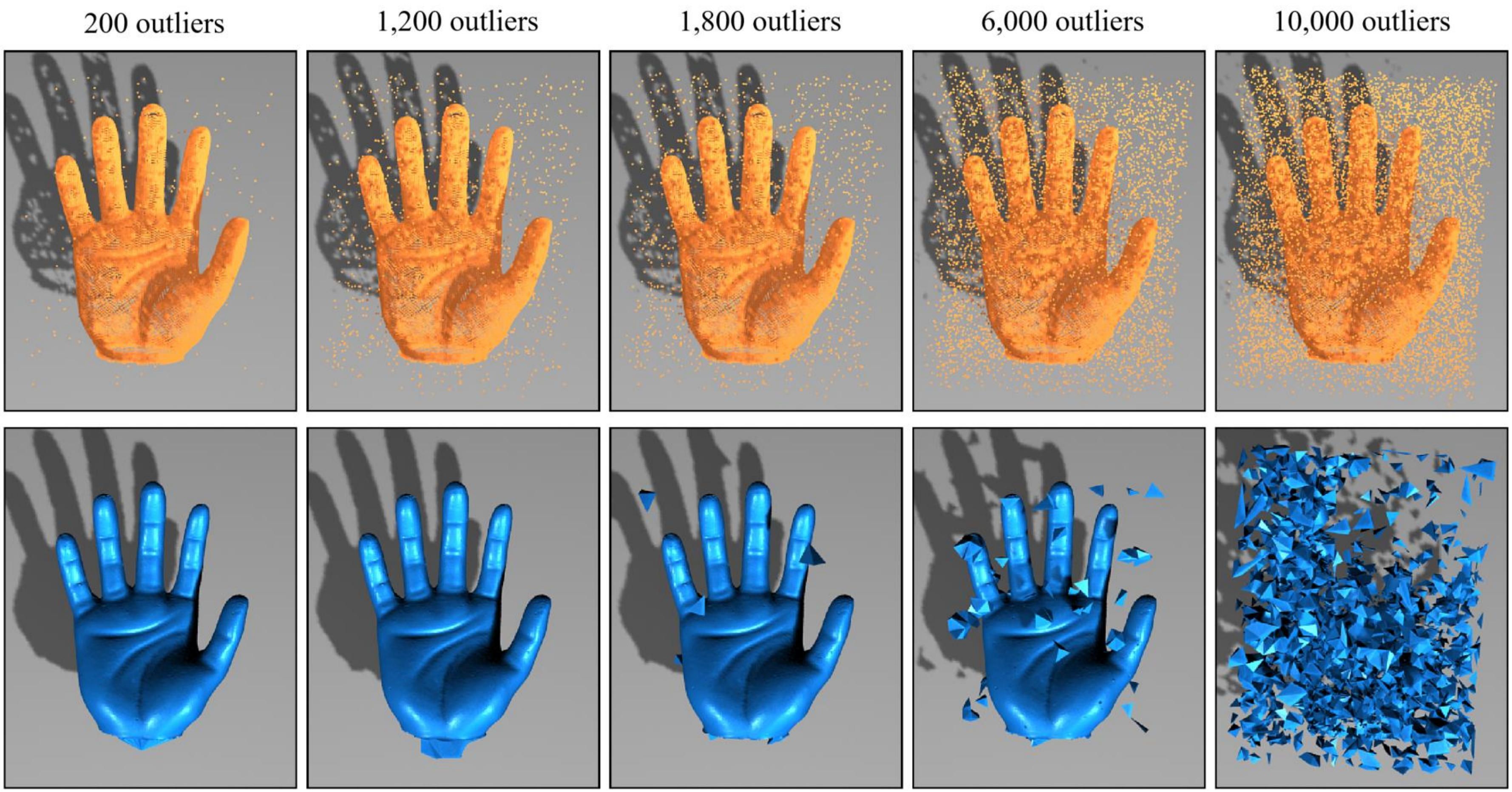
e) The power crust and the power shape of its interior solid.

Eigen Crust [Kolluri et al. 04]



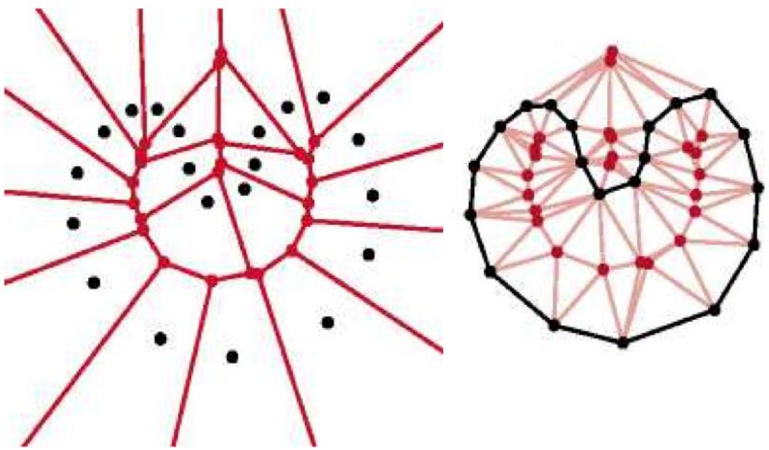


Eigen Crust [Kolluri et al. 04]

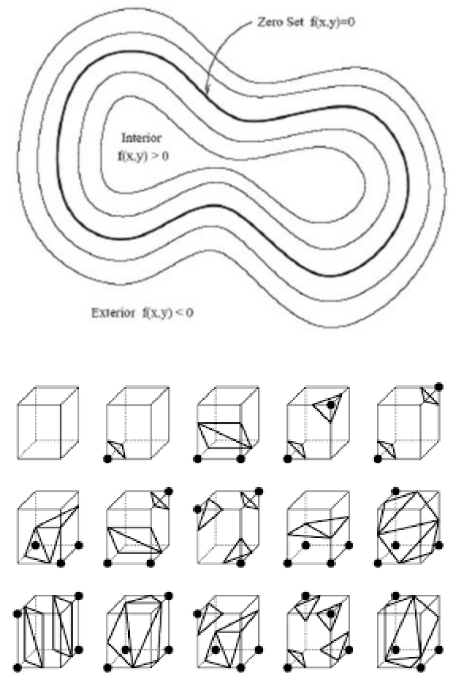


Eigencrust

Two Approaches

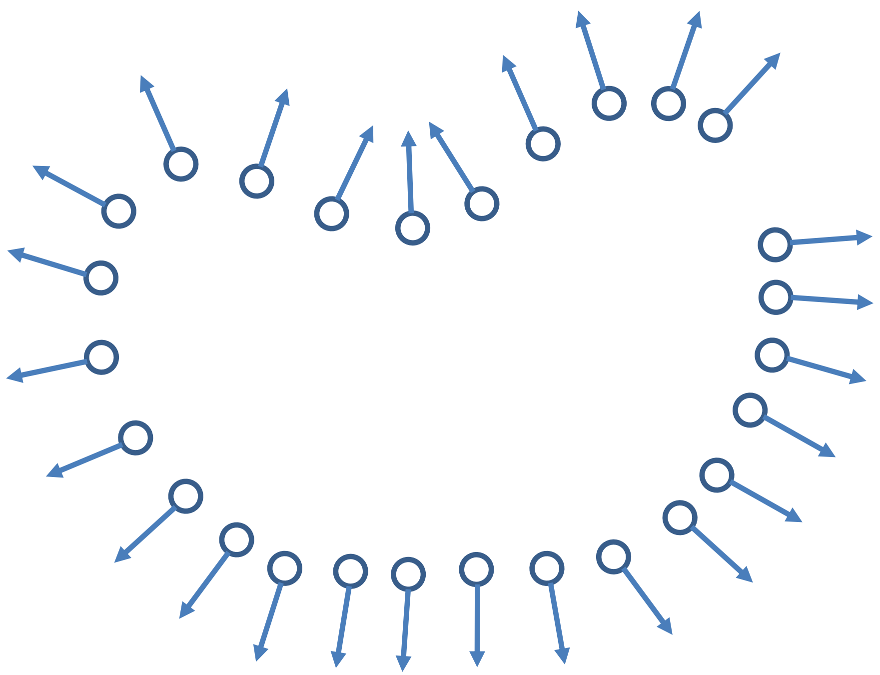


Computational Geometry Based



Implicit Surface -> Contouring

Point Cloud -> Implicit Surface?



Challenges:

Irregular Samples

Holes

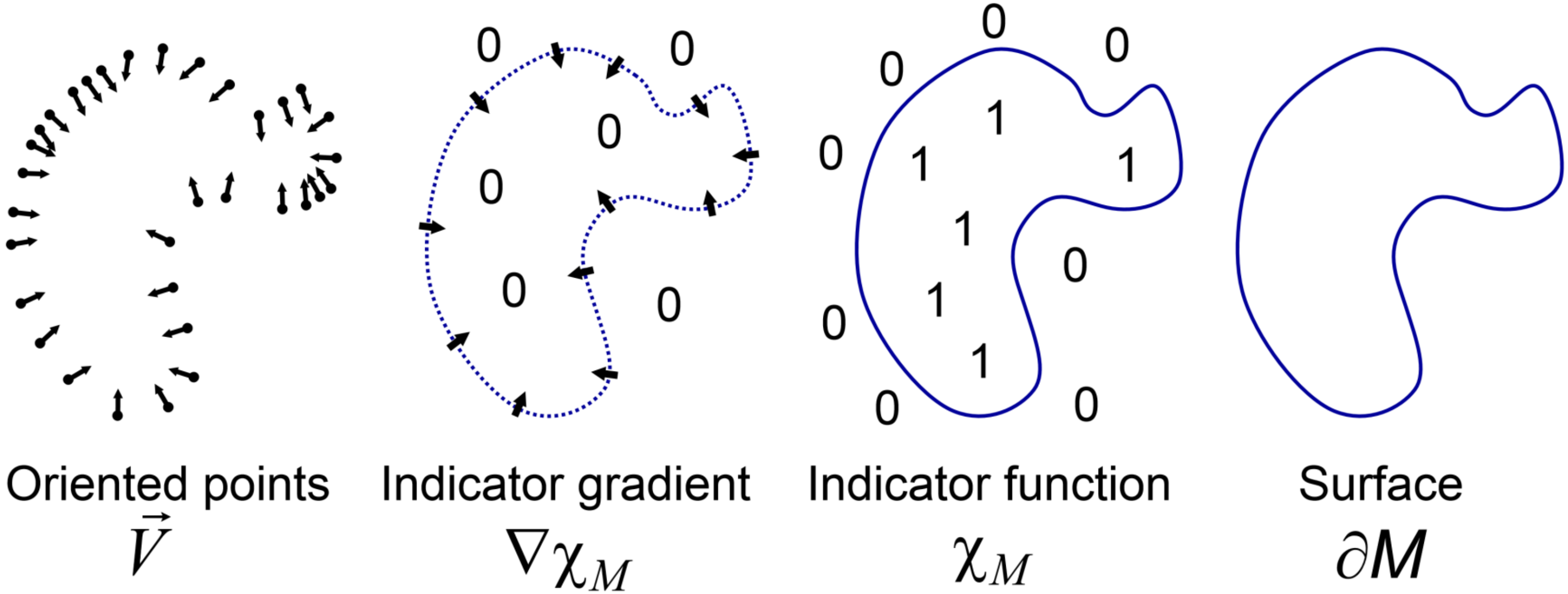
Noise

$$\sum_{i=1}^{N} w_i(\boldsymbol{x})(\boldsymbol{x}-\boldsymbol{p}_i)^T \boldsymbol{n}_i = 0$$

$$w_i(\boldsymbol{x}) = \frac{\exp\left(-\frac{\|\boldsymbol{x}-\boldsymbol{p}_i\|^2}{2\sigma^2}\right)}{\sum_{i=1}^{n} \exp\left(-\frac{\|\boldsymbol{x}-\boldsymbol{p}_i\|^2}{2\sigma^2}\right)}$$

Defining point-set surfaces [Amenta et al. 05]

Poisson surface reconstruction [Kazhdan et al. 06]



Poisson surface reconstruction [Kazhdan et al. 06]

Define the vector field:

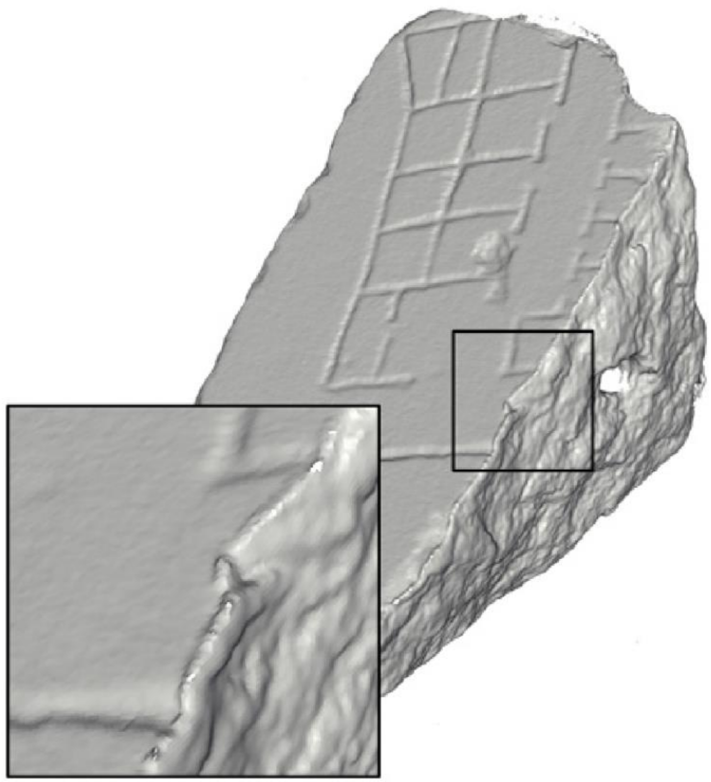
$$\nabla(\chi_M * \tilde{F})(q) = \sum_{s \in S} \int_{\mathscr{P}_s} \tilde{F}_p(q) \vec{N}_{\partial M}(p) dp$$

$$\approx \sum_{s \in S} |\mathscr{P}_s| \, \tilde{F}_{s.p}(q) \, s.\vec{N} \equiv \vec{V}(q)$$

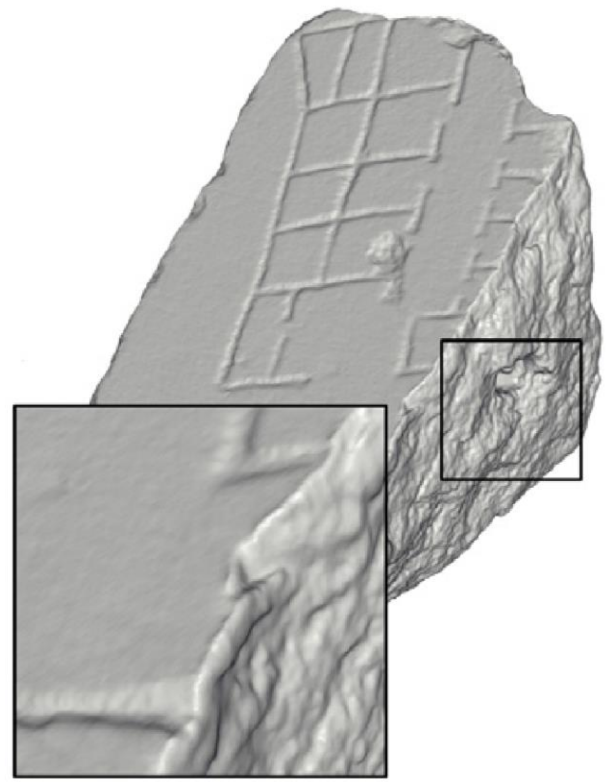
Solve the Poisson equation:

$$\Delta \tilde{\chi} = \nabla \cdot \vec{V}.$$

Poisson surface reconstruction [Kazhdan et al. 06]



VRIP

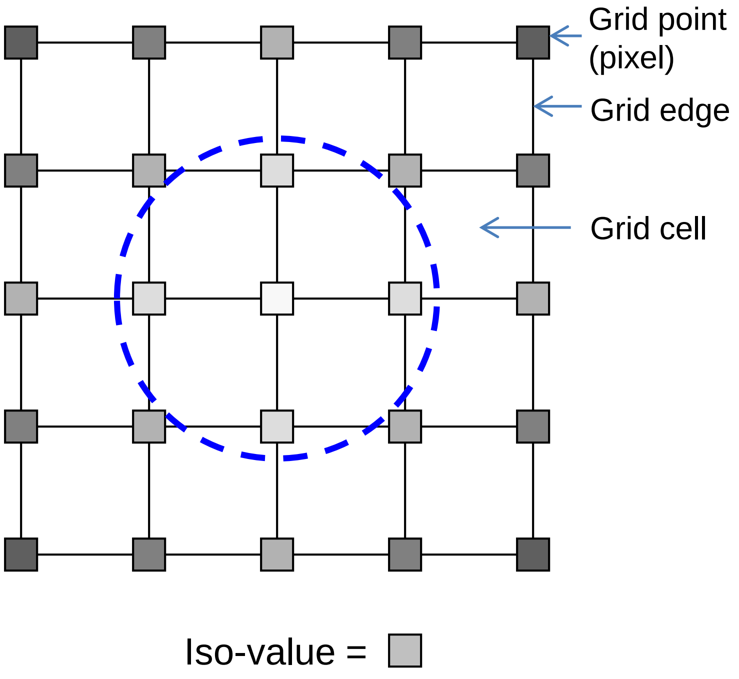


Poisson Surface Reconstruction

Contouring

Contouring (On A Grid)

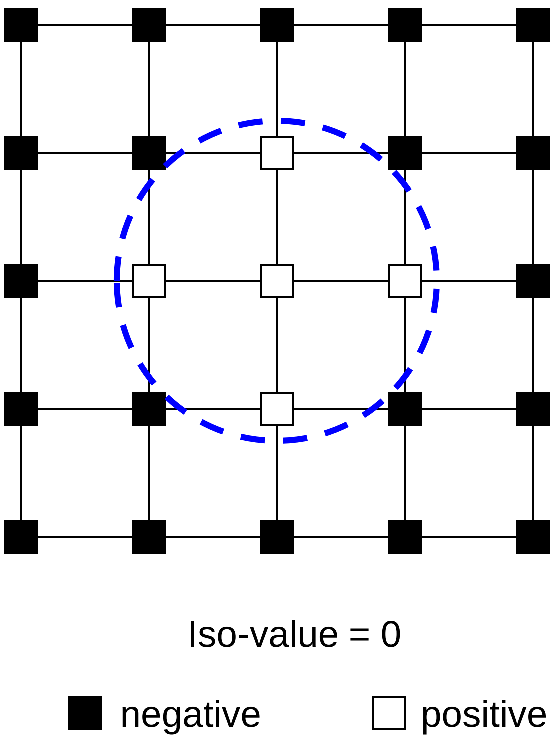
- Input
  - A grid where each grid point (pixel or voxel) has a value (color)
  - An iso-value (threshold)
- Output
  - A closed, manifold, non-intersecting polyline (2D) or mesh (3D) that separates grid points above the iso-value from those that are below the iso-value.



Slide Credit: Tao Ju

Contouring (On A Grid)

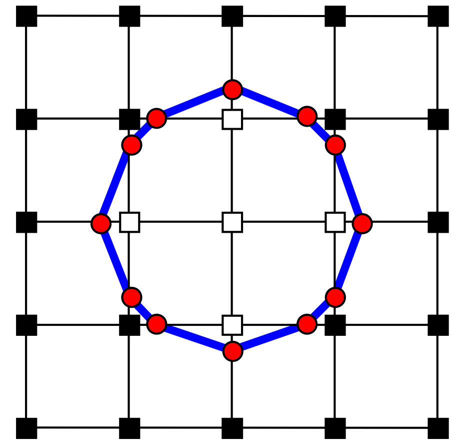
- Input
  - A grid where each grid point (pixel or voxel) has a value (color)
  - An iso-value (threshold)
- Output
  - Equivalently, we extract the zero-contour (separating negative from positive) after subtracting the iso-value from the grid points



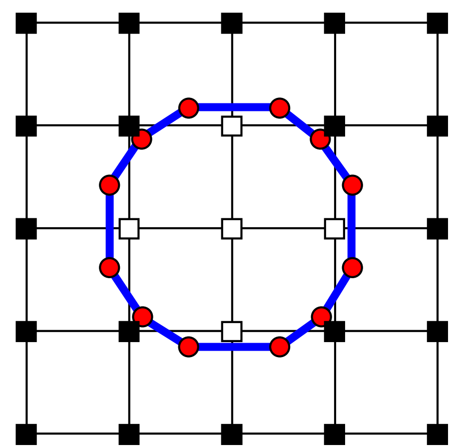
Slide Credit: Tao Ju

Algorithms

- Primal methods
  - Marching Squares (2D), Marching Cubes (3D)
  - Placing vertices on grid edges



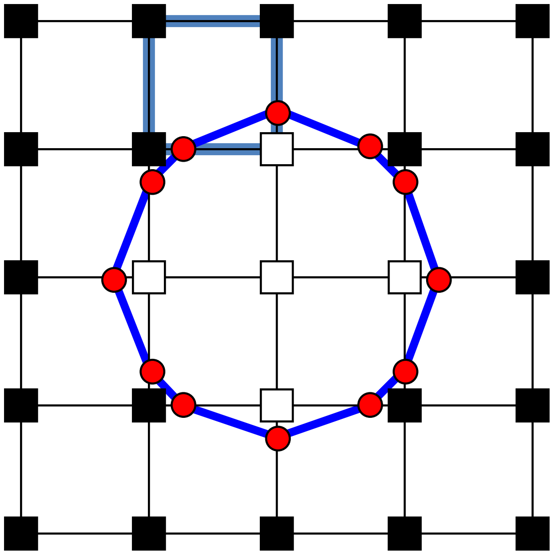
- Dual methods
  - Dual Contouring (2D,3D)
  - Placing vertices in grid cells



Slide Credit: Tao Ju

Marching Squares (2D)

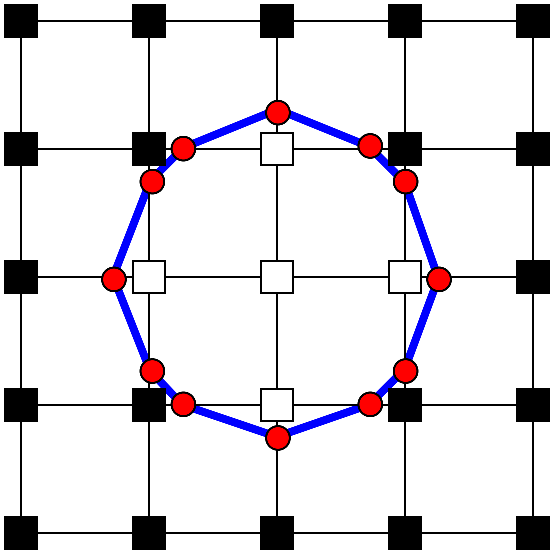
- For each grid cell with a sign change
  - Create one vertex on each grid edge with a sign change
  - Connect vertices by lines



Slide Credit: Tao Ju

Marching Squares (2D)

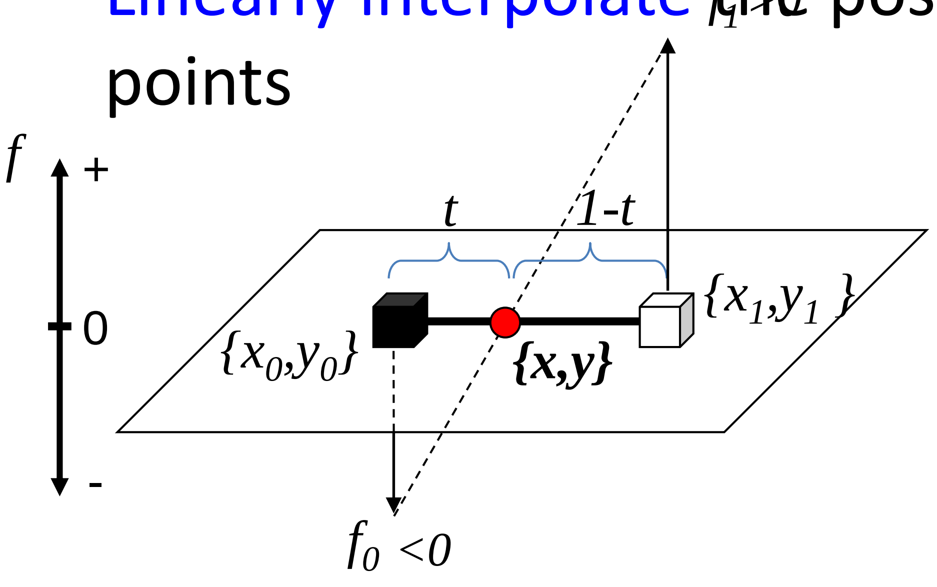
- For each grid cell with a sign change
  - Create one vertex on each grid edge with a sign change
  - Connect vertices by lines



Slide Credit: Tao Ju

Marching Squares (2D)

- Creating vertices: linear interpolation
  - Assuming the underlying, continuous function is linear on the grid edge
  - Linearly interpolate the positions of the two grid points



$$t = \frac{f_0}{f_0 - f_1}$$

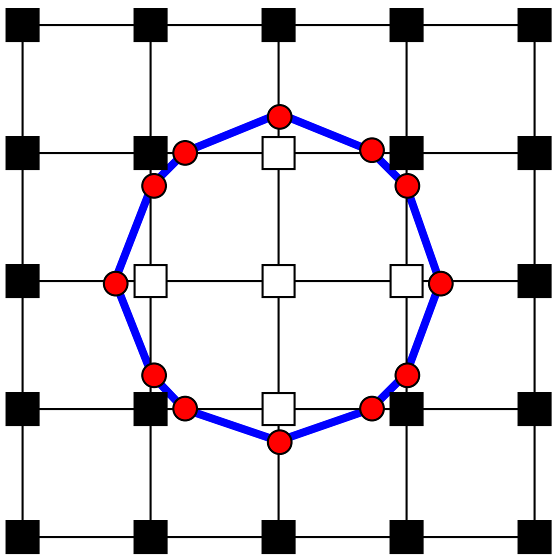
$$x = x_0 + t(x_1 - x_0)$$

$$y = y_0 + t(y_1 - y_0)$$

Slide Credit: Tao Ju

Marching Squares (2D)

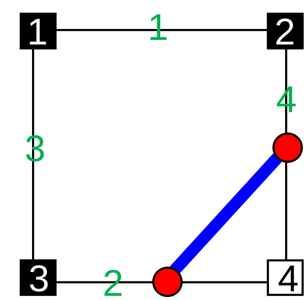
- For each grid cell with a sign change
  - Create one vertex on each grid edge with a sign change
  - Connect vertices by lines



Slide Credit: Tao Ju

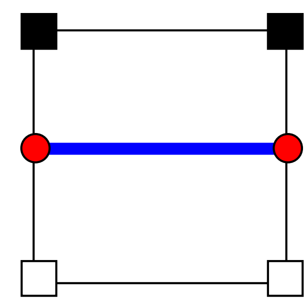
Marching Squares (2D)

- Connecting vertices by lines
  - Lines shouldn't intersect
  - Each vertex is used once
    - So that it will be used exactly twice by the two cells incident on the edge
- Two approaches
  - Do a walk around the grid cell
    - Connect consecutive pair of vertices
  - Or, using a pre-computed look-up table
    - 2^4=16 sign configurations
    - For each sign configuration, it stores the indices of the grid edges whose vertices make up the lines.



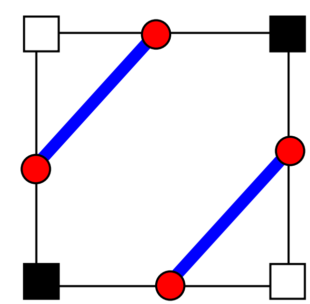
Key: 0 0 0 1

Data: {{2,4}}



Key: 0 0 1 1

Data: {{3,4}}



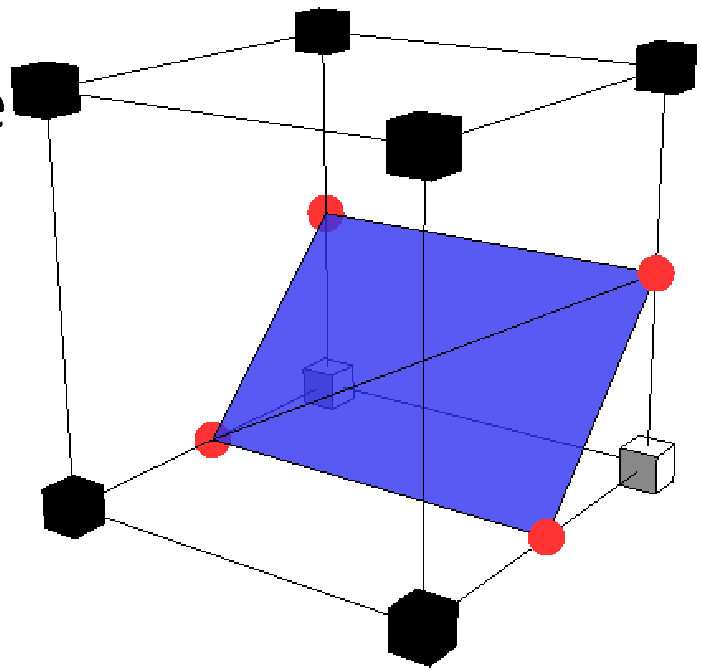
Key: 1 0 0 1

Data: {{1,3}, {2,4}}

Slide Credit: Tao Ju

Marching Cubes (3D)

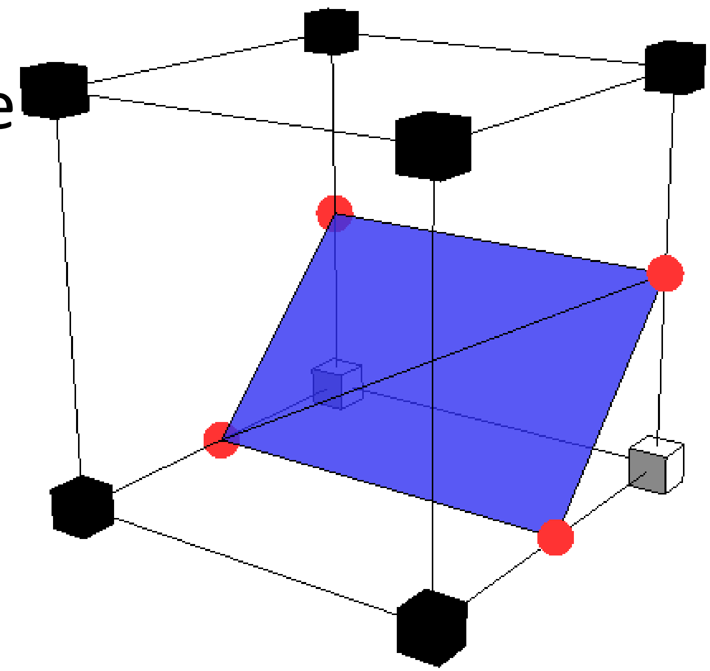
- For each grid cell with a sign change
  - Create one vertex on each grid edge with a sign change (using linear interpolation)
  - Connect vertices into triangles



Slide Credit: Tao Ju

Marching Cubes (3D)

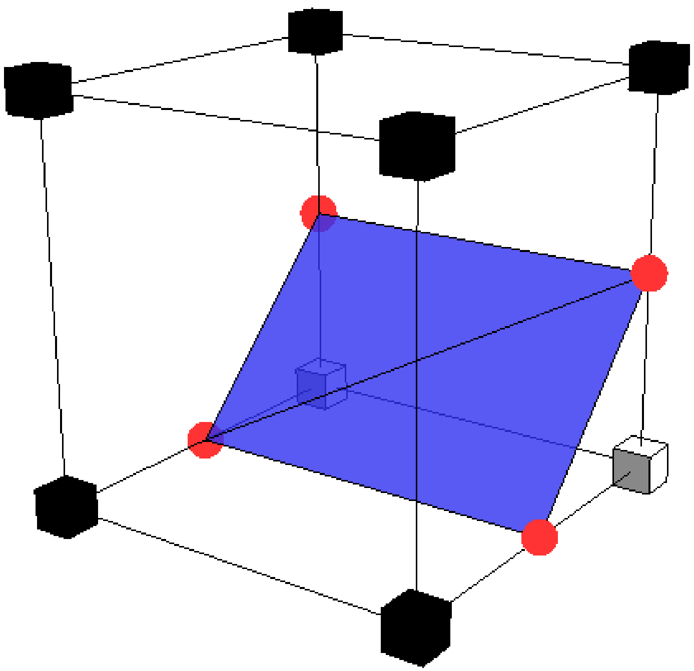
- For each grid cell with a sign change
  - Create one vertex on each grid edge with a sign change (using linear interpolation)
  - Connect vertices into triangles



Slide Credit: Tao Ju

Marching Cubes (3D)

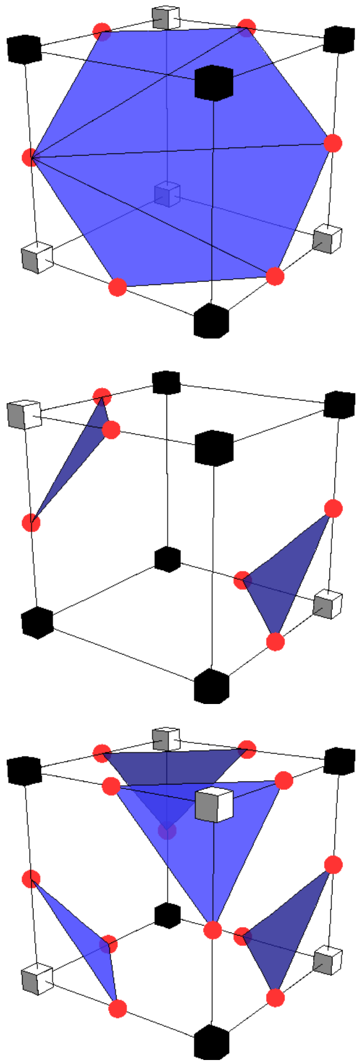
- Connecting vertices by triangles
  - Triangles shouldn't intersect
  - To be a closed manifold:
    - Each vertex used by a triangle "fan"
    - Each mesh edge used by 2 triangles (if inside grid cell) or 1 triangle (if on a grid face)



Slide Credit: Tao Ju

Marching Cubes (3D)

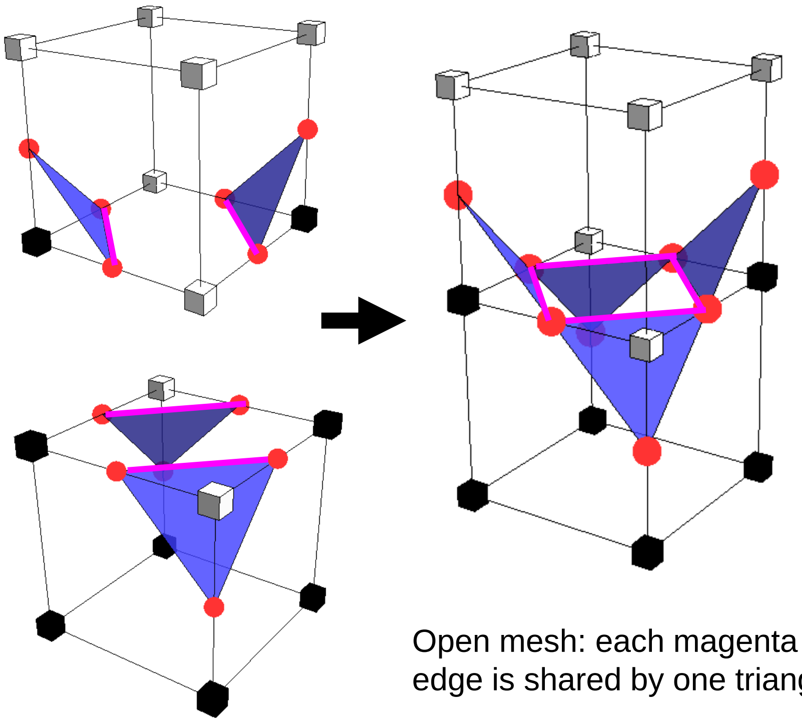
- Connecting vertices by triangles
  - Triangles shouldn't intersect
  - To be a closed manifold:
    - Each vertex used by a triangle "fan"
    - Each mesh edge used by 2 triangles (if inside grid cell) or 1 triangle (if on a grid face)



Slide Credit: Tao Ju

Marching Cubes (3D)

- Connecting vertices by triangles
  - Triangles shouldn't intersect
  - To be a closed manifold:
    - Each vertex used by a triangle "fan"
    - Each mesh edge used by 2 triangles (if inside grid cell) or 1 triangle (if on a grid face)

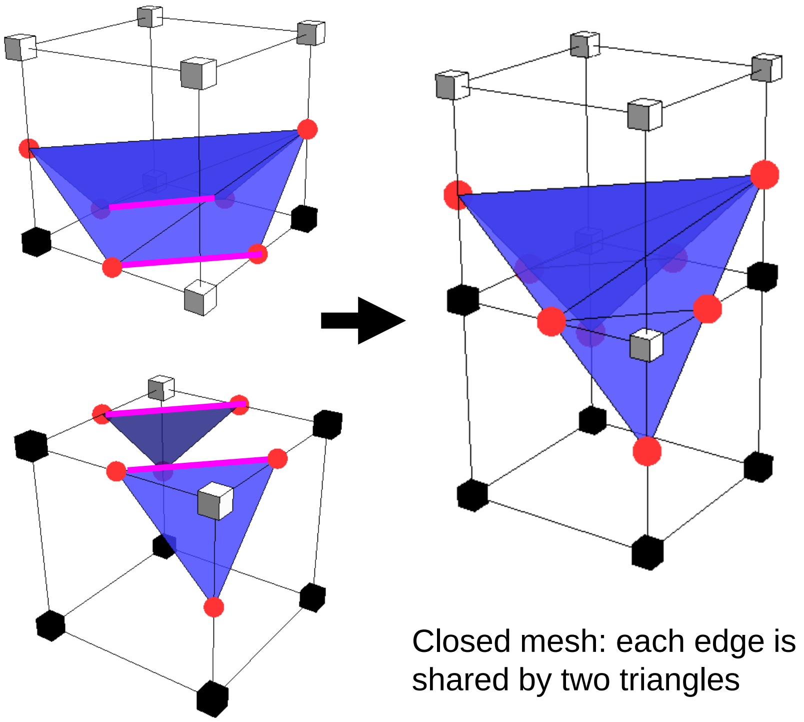


Open mesh: each magenta edge is shared by one triangle

Slide Credit: Tao Ju

Marching Cubes (3D)

- Connecting vertices by triangles
  - Triangles shouldn't intersect
  - To be a closed manifold:
    - Each vertex used by a triangle "fan"
    - Each mesh edge used by 2 triangles (if inside grid cell) or 1 triangle (if on a grid face)
    - Each mesh edge on the grid face is shared between adjacent cells

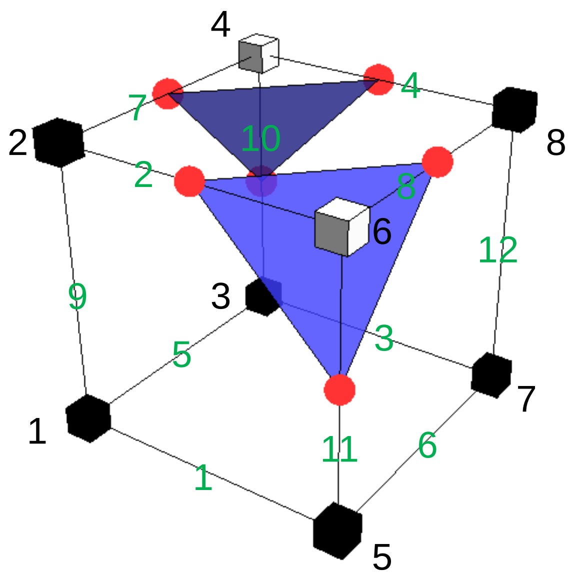


Closed mesh: each edge is shared by two triangles

Slide Credit: Tao Ju

Marching Cubes (3D)

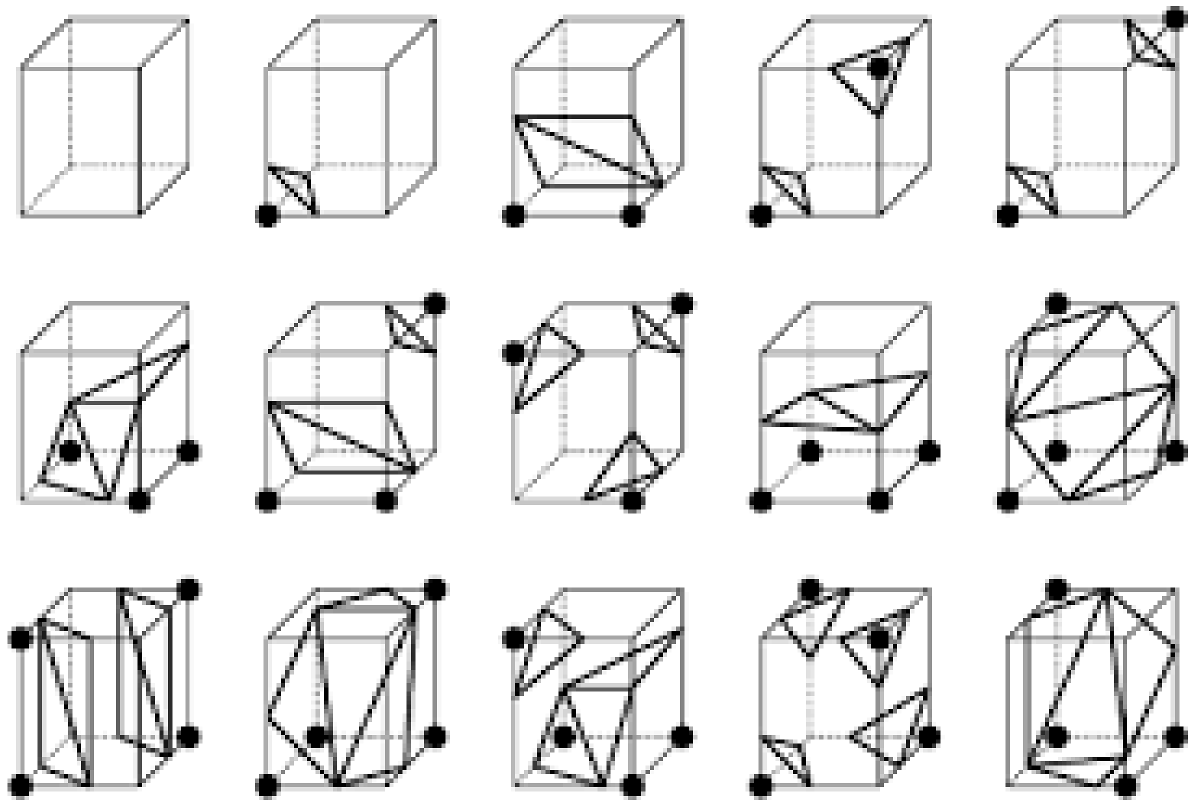
- Connecting vertices by triangles
  - Triangles shouldn't intersect
  - To be a closed manifold:
    - Each vertex used by a triangle "fan"
    - Each mesh edge used by 2 triangles (if inside grid cell) or 1 triangle (if on a grid face)
    - Each mesh edge on the grid face is shared between adjacent cells
- Look-up table
  - 2^8=256 sign configurations
  - For each sign configuration, it stores indices of the grid edges whose vertices make up the triangles



Sign: "0 0 0 1 0 1 0 0"
Triangles: {{2,8,11},{4,7,10}}

Slide Credit: Tao Ju

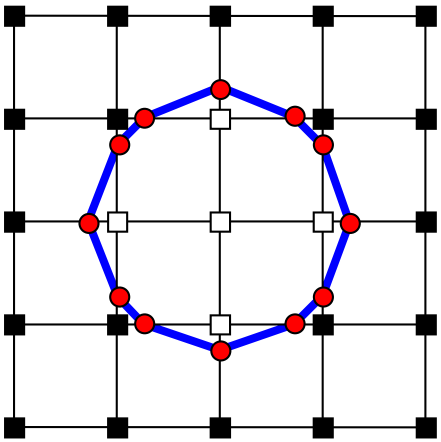
Lookup Table



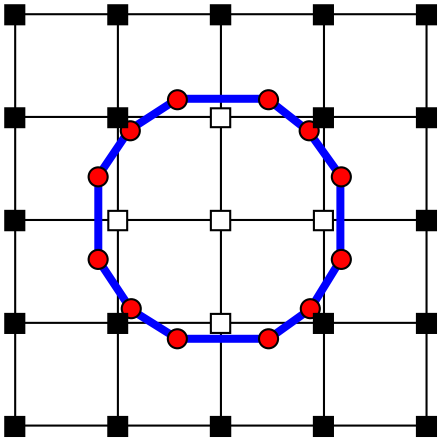
Slide Credit: Tao Ju

Algorithms

- Primal methods
  - Marching Squares (2D), Marching Cubes (3D)
  - Placing vertices on grid edges



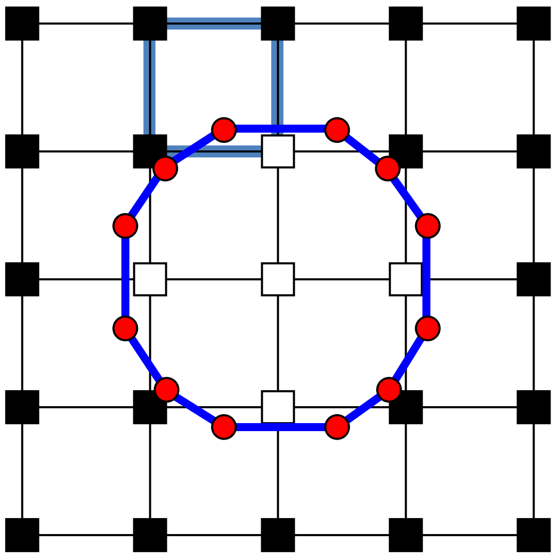
- Dual methods
  - Dual Contouring (2D,3D)
  - Placing vertices in grid cells



Slide Credit: Tao Ju

Dual Contouring (2D)

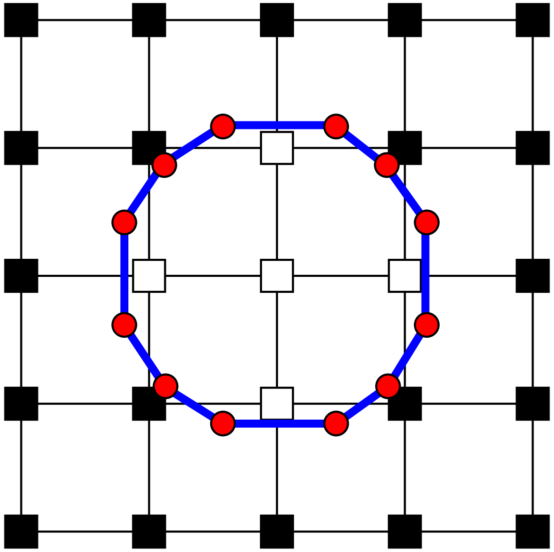
- For each grid cell with a sign change
  - Create one vertex
- For each grid edge with a sign change
  - Connect the two vertices in the adjacent cells with a line segment



Slide Credit: Tao Ju

Dual Contouring (2D)

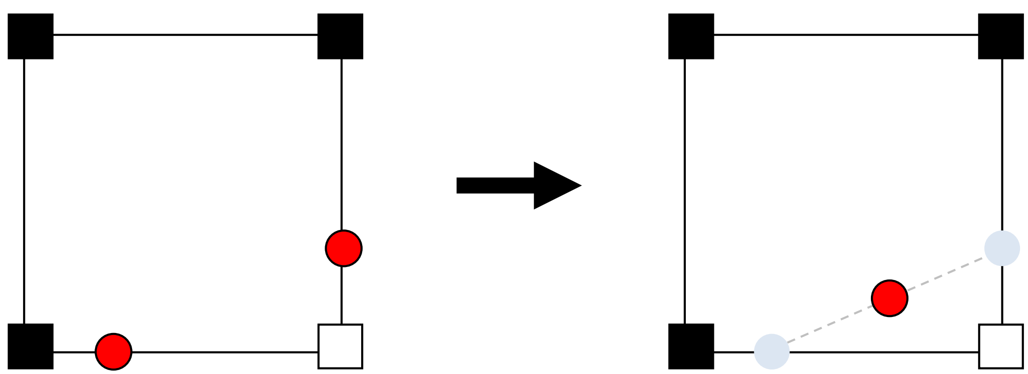
- For each grid cell with a sign change
  - Create one vertex
- For each grid edge with a sign change
  - Connect the two vertices in the adjacent cells with a line segment



Slide Credit: Tao Ju

Dual Contouring (2D)

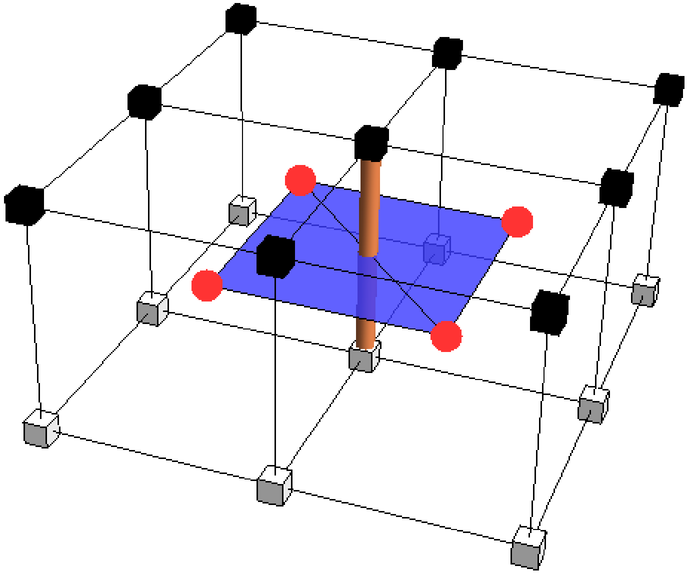
- Creating the vertex within a cell
  - Compute one point on each grid edge with a sign change (by linear interpolation)
    - There could be more than two sign-changing edges, so >2 points possible
  - Take the centroid of these points



Slide Credit: Tao Ju

Dual Contouring (3D)

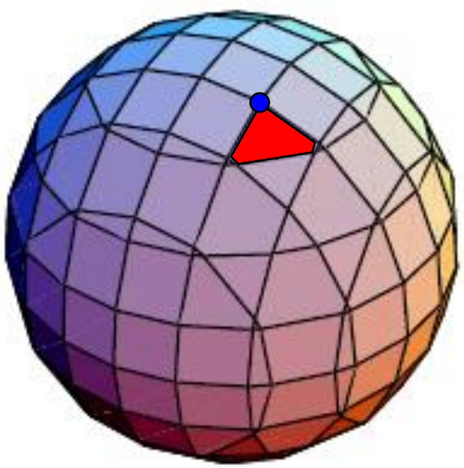
- For each grid cell with a sign change
  - Create one vertex (same way as 2D)
- For each grid edge with a sign change
  - Create a quad (or two triangles) connecting the four vertices in the adjacent grid cubes
  - No look-up table is needed!



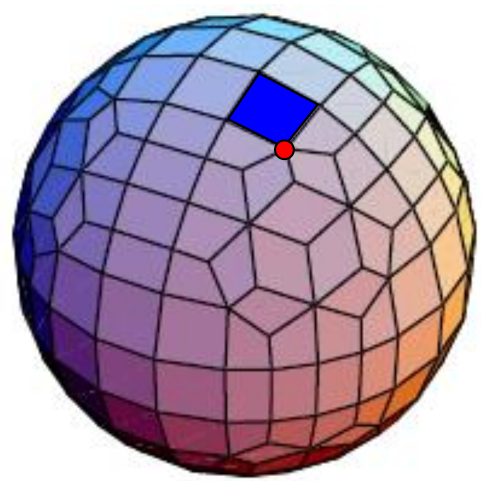
Slide Credit: Tao Ju

Duality

- The two outputs have a dual structure
  - Vertices and quads of Dual Contouring correspond (roughly) to un-triangulated polygons and vertices produced by Marching Cubes



Marching Cubes



Dual Contouring

Slide Credit: Tao Ju

Slide Credit: Tao Ju

Primal vs. Dual

- Marching Cubes
  - ✓ Always manifold
  - ✗ Requires look-up table in 3D
  - ✗ Often generates thin and tiny poly

Marching Cubes

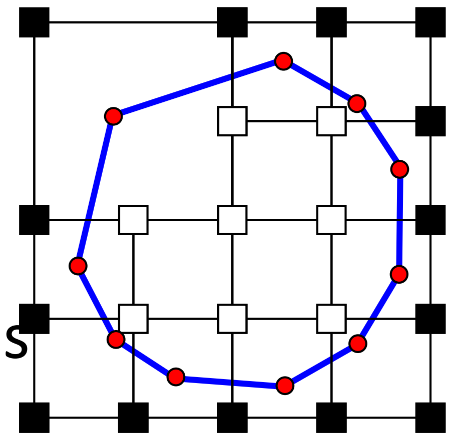
- Dual Contouring
  - ✗ Can be non-manifold
  - ✓ No look-up table needed
  - ✓ Generates better-shaped polygon

Dual Contouring

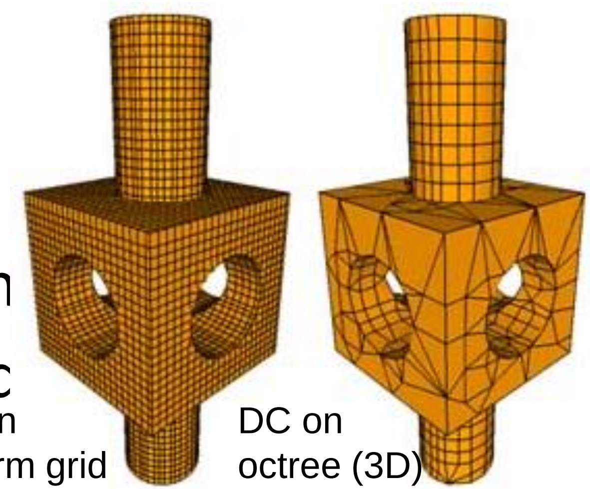
Slide Credit: Tao Ju

Primal vs. Dual

- Marching Cubes
  - ✓ Always manifold
  - ✗ Requires look-up table in 3D
  - ✗ Often generates thin and tiny polygons
  - ✗ Restricted to uniform grids
- Dual Contouring
  - ✗ Can be non-manifold
  - ✓ No look-up table needed
  - ✓ Generates better-shaped polygon
  - ✓ Can be applied to any type of gric



DC on a Quadtree (2D)



DC on uniform grid

DC on octree (3D)