Analysis-ready 3D reconstructions of complex objects from planar cross-sectional slices

John Edwards

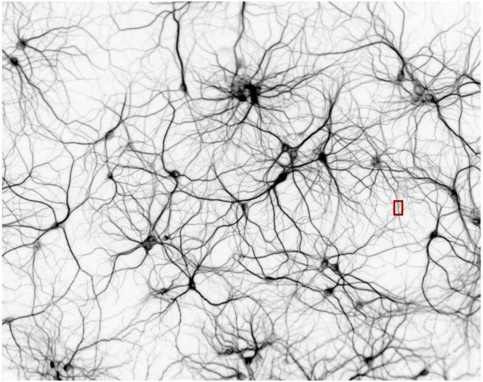
Department of Computer Science
Computational Visualization Center
Institute of Computational Engineering and Sciences
University of Texas at Austin

Research Preparation Exam, March 29, 2011

Outline

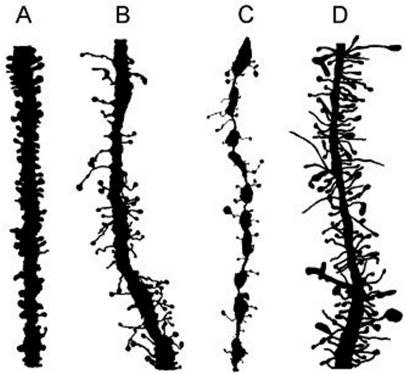
Outline

- Number of neurons in the human brain: 10^{10}
- Number of synaptic connections: 10^{14}
- Large amount of research on "connectome"
- Neurons have complex geometries



http://www.homepages.ucl.ac.uk/~sjjgnle/, [Fiala et al., 2002]

Neurons

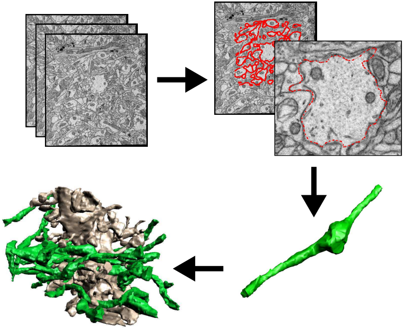


[Fiala et al., 2002]

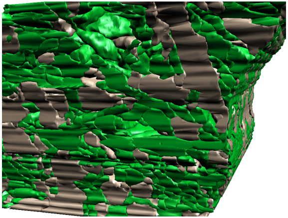
- Geometries play a role:
  - A Neurologically normal
  - B Mentally disabled
  - C Severe neurobehavioral failure
  - D Fragile X syndrome

- Electrophysiological simulations elucidate effects of geometries on
  - neuronal topology and combinatorics
  - learning, behavior, and memory

Neuronal reconstruction



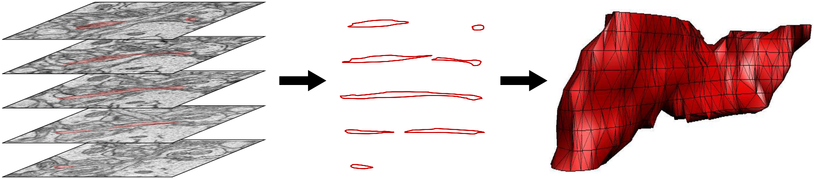
- Input: a series of "traced" 2D Electron Microscopy (EM) cross-sectional images
- Desired: 3D geometries suitable for simulation

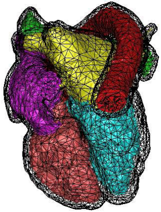


3D reconstruction

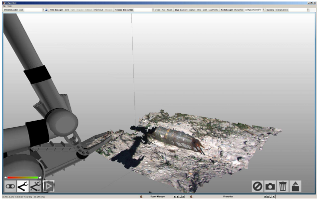
Problem statement

Given planar contours in parallel slices, build analysis-ready 3D surfaces

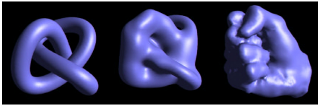




[Bajaj & Goswami, 2008]



[Alberts et al., 2009]



[Turk & O'Brien, 1999]

Other applications:

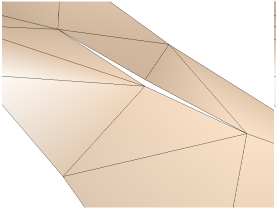
- Medical applications at organ and cellular level using imagery from
  - magnetic resonance imaging (MRI)
  - computed tomography (CT)
  - ultrasound
- geospatial information systems (GIS)
- robotics
- computer-aided design (CAD)
- special effects

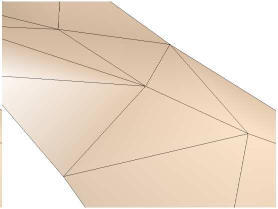
Outline

Analysis-ready requirements

- A surface is analysis-ready if it
    - is water-tight
    - has oriented surface normals
    - is non-intersecting
    - has no mesh irregularities
    - has manifold edges and vertices
    - is composed of low aspect ratio triangles
    - is topologically correct
    - is close to the true surface

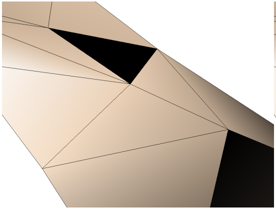
Analysis-ready requirements


not water-tight

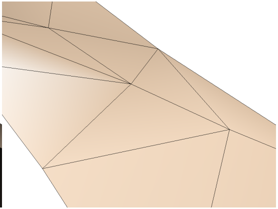

water-tight

- is water-tight
- has oriented surface normals
- is non-intersecting
- has no mesh irregularities
- has manifold edges and vertices
- is composed of low aspect ratio triangles
- is topologically correct
- is close to the true surface

Analysis-ready requirements



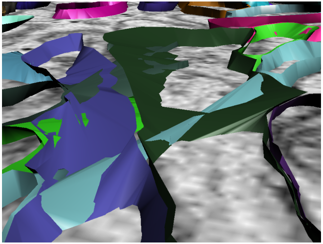
non-oriented

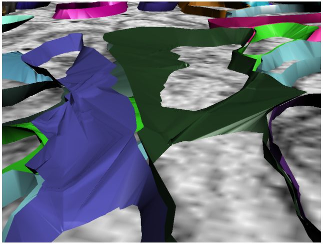


oriented

- is water-tight
- has oriented surface normals
- is non-intersecting
- has no mesh irregularities
- has manifold edges and vertices
- is composed of low aspect ratio triangles
- is topologically correct
- is close to the true surface

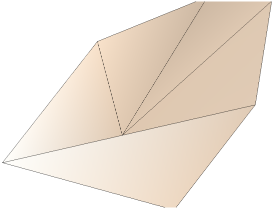
Analysis-ready requirements



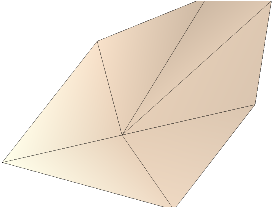


- is water-tight
- has oriented surface normals
- is non-intersecting
- has no mesh irregularities
- has manifold edges and vertices
- is composed of low aspect ratio triangles
- is topologically correct
- is close to the true surface

Analysis-ready requirements



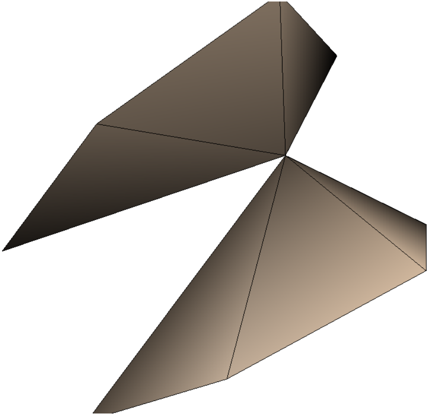
irregularity



no irregularities

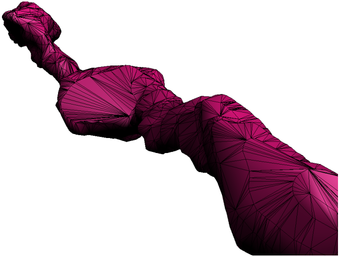
- is water-tight
- has oriented surface normals
- is non-intersecting
- has no mesh irregularities
- has manifold edges and vertices
- is composed of low aspect ratio triangles
- is topologically correct
- is close to the true surface

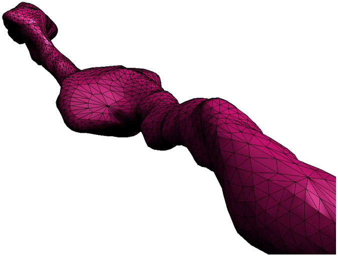
Analysis-ready requirements



non-manifold

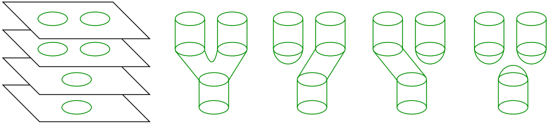
- is water-tight
- has oriented surface normals
- is non-intersecting
- has no mesh irregularities
- has manifold edges and vertices
- is composed of low aspect ratio triangles
- is topologically correct
- is close to the true surface

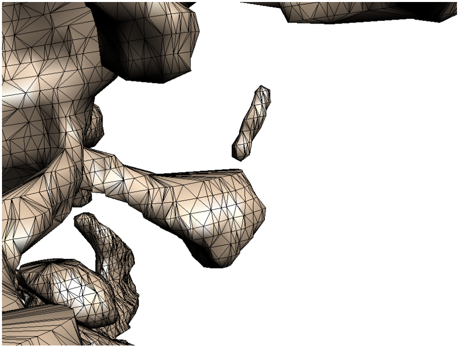




- is water-tight
- has oriented surface normals
- is non-intersecting
- has no mesh irregularities
- has manifold edges and vertices
- is composed of low aspect ratio triangles
- is topologically correct
- is close to the true surface

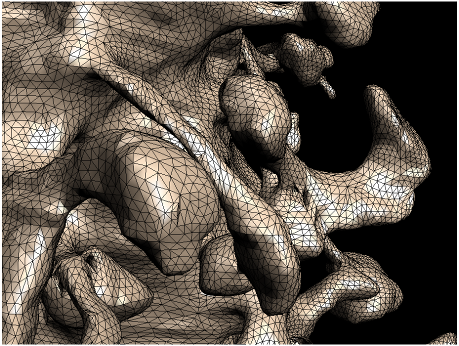
Analysis-ready requirements





- is water-tight
- has oriented surface normals
- is non-intersecting
- has no mesh irregularities
- has manifold edges and vertices
- is composed of low aspect ratio triangles
- is topologically correct
- is close to the true surface

Analysis-ready requirements

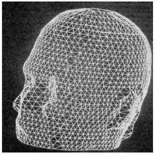


- is water-tight
- has oriented surface normals
- is non-intersecting
- has no mesh irregularities
- has manifold edges and vertices
- is composed of low aspect ratio triangles
- is topologically correct
- is close to the true surface

Outline

Reconstruction methods - geometric

Single component reconstruction



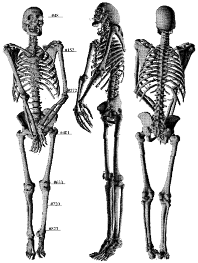
[Fuchs et al., 1977]

- One of the seminal works was Fuchs et al. [Fuchs et al., 1977] who posed the problem and presented a triangulation solution
- Spawned a number of geometric approaches [Christiansen & Sederberg, 1978, Boissonnat, 1988, Barequet & Sharir, 1994]

- is water-tight
- has oriented surface normals
- is non-intersecting
- has no mesh irregularities
- has manifold edges and vertices
- is composed of low aspect ratio triangles
- is topologically correct
- is close to the true surface

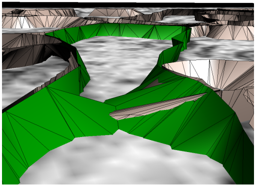
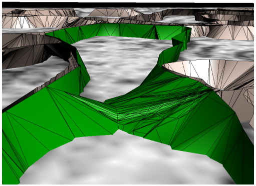
Reconstruction methods - geometric

Single component reconstruction



- is water-tight
- has oriented surface normals
- is non-intersecting
- has no mesh irregularities
- has manifold edges and vertices
- is composed of low aspect ratio triangles
- is topologically correct
- is close to the true surface

- [Bajaj et al., 1996] presented an algorithm with geometric and topological guarantees:
  - Guaranteed to be water-tight
  - Guaranteed topology with some assumptions
  - Supports arbitrary topologies such as branching

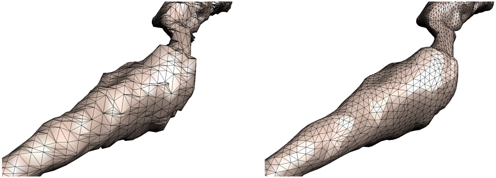



- [Edwards & Bajaj, 2010] post-processes surfaces and removes intersections
  - Maintains all guarantees of original algorithm
  - Produces many sliver triangles

- is water-tight
- has oriented surface normals
- is non-intersecting
- has no mesh irregularities
- has manifold edges and vertices
- is composed of low aspect ratio triangles
- is topologically correct
- is close to the true surface

Reconstruction methods - geometric

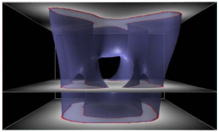
Single component meshing



- [Zhang et al., 2005] further post-processes surfaces and produces a mesh with quality triangles
  - No guarantees about maintaining intersection-free geometries

- is water-tight
- has oriented surface normals
- is non-intersecting
- has no mesh irregularities
- has manifold edges and vertices
- is composed of low aspect ratio triangles
- is topologically correct
- is close to the true surface

Reconstruction methods - implicit



- [Turk & O'Brien, 1999, Bermano et al., 2011] generates an implicit function in 3D then takes the zero-set
  - Quality of the geometry is dependent on the zero-set extraction
  - Non-intersecting by definition, although after extraction the surfaces will be touching
  - Very small geometries may get capped off and not carried through unknown region
    - Can be fixed using A-splines, but scaffolds are essentially a geometric reconstruction

- is water-tight
- has oriented surface normals
- is non-intersecting
- has no mesh irregularities
- has manifold edges and vertices
- is composed of low aspect ratio triangles
- is topologically correct
- is close to the true surface

Outline

Inter-component intersections

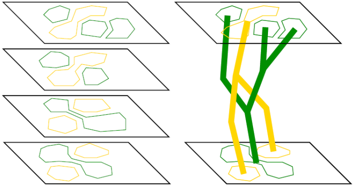
Intersections can occur between singly-reconstructed components in inter-slice regions

Such intersections occur when data is

- highly anisotropic
- tightly packed
- tortuous

Our algorithm reconstructs multiple components or a "forest" of structures:

1. Use single component reconstruction method on each component
2. Remove intersections between components



Intersection removal

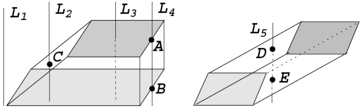
Our algorithm removes intersections by adjusting only z-values of existing tiles (triangles)

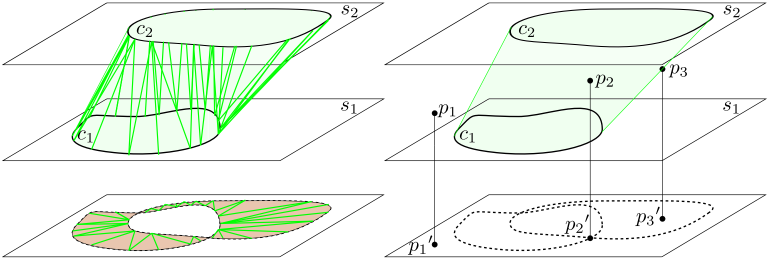
1. Removes intersections of a single component in a linear number of steps
2. Will not cause additional intersections
3. Branching treated just like any other intersection

Penumbral contours

Claim

All intersections occur in *penumbral regions*. A point's *penumbral contour* is the contour whose projection contains the projected point.



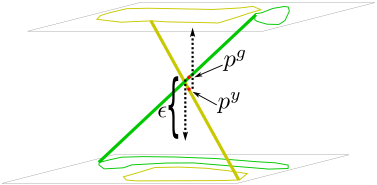


Conflict points

A *conflict point* is a point of intersection. Somewhat more formally:

Definition

Point p^g is called a *conflict point* if there is some point p^y such that the projections are equal ($p^{y\prime} = p^{g\prime}$) and p^y is closer to p^g's penumbral contour than p^g is.

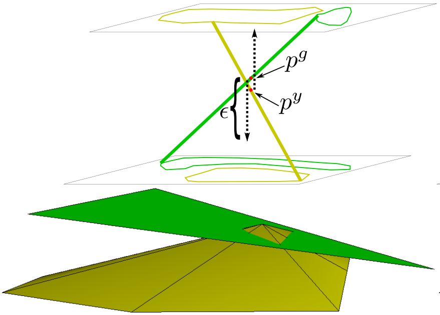


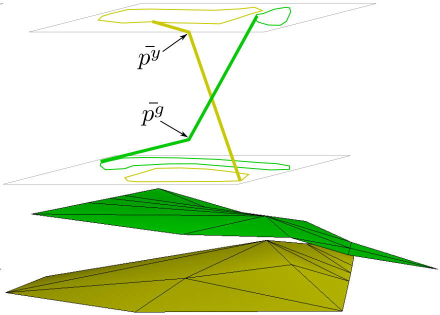
Claim

Two components C^g and C^y intersect if and only if there is at least one conflict point on the surface of either component.

Removing conflicts

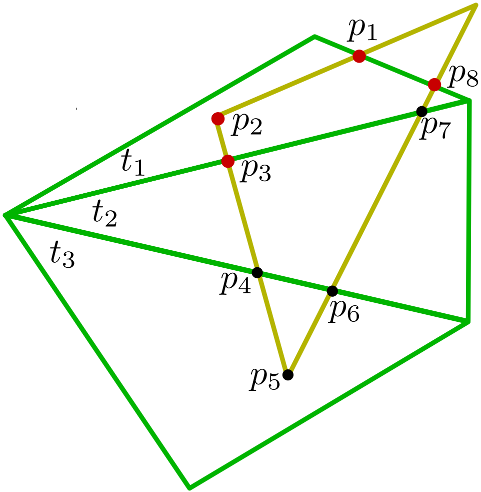
We can resolve conflict points by moving them in the directions of their penumbral contours without worrying about causing additional intersections (proof on slide A1). Once all conflict points are resolved, all intersections are removed.





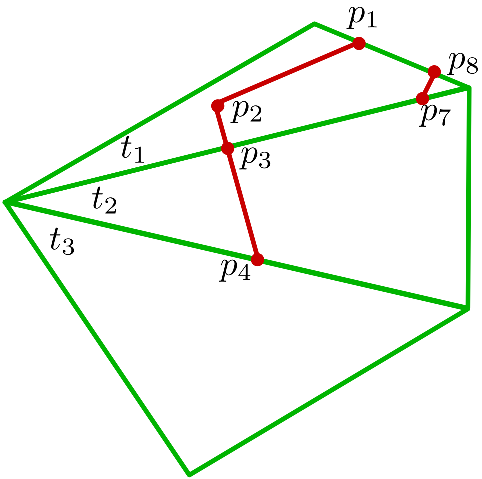
Conflict removal algorithm

1. Detect conflict points.
2. Trace paths between conflict points along edges of yellow tile. We call these *cut paths*.
3. Use original tiles and cut paths to induce new polygons.
4. Triangulate polygons and move conflict points along z-axis.



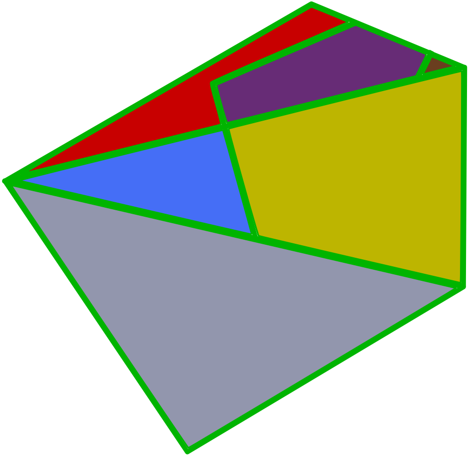
Conflict removal algorithm

1. Detect conflict points.
2. Trace paths between conflict points along edges of yellow tile. We call these *cut paths*.
3. Use original tiles and cut paths to induce new polygons.
4. Triangulate polygons and move conflict points along z-axis.

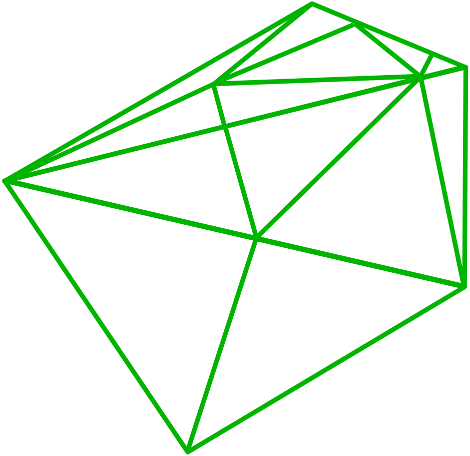


Conflict removal algorithm

1. Detect conflict points.
2. Trace paths between conflict points along edges of yellow tile. We call these *cut paths*.
3. Use original tiles and cut paths to induce new polygons.
4. Triangulate polygons and move conflict points along z-axis.



1. Detect conflict points.
2. Trace paths between conflict points along edges of yellow tile. We call these *cut paths*.
3. Use original tiles and cut paths to induce new polygons.
4. Triangulate polygons and move conflict points along z-axis.

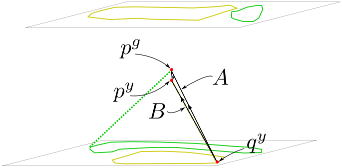


Separating by a given delta

$d = \frac{|\overline{\mathbf{A}} \times \overline{\mathbf{B}}|}{|\overline{\mathbf{B}}|}$

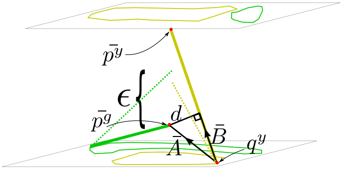
Substituting for $\overline{\mathbf{A}}$ and $\overline{\mathbf{B}}$:

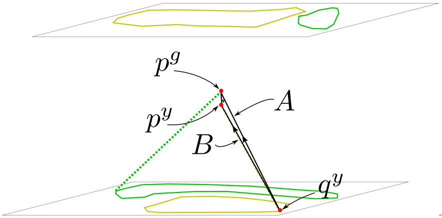
$$\begin{aligned} d^2 = & \quad ((A_y(B_z - \epsilon) - (A_z + \epsilon)B_y)^2 \\ & + \quad ((A_z + \epsilon)B_x - A_x(B_z - \epsilon))^2 \\ & + \quad (A_x B_y - A_y B_x)^2)/(B_x^2 + B_y^2 + (B_z + \epsilon)^2) \end{aligned}$$



After collecting ϵ:

$$\begin{aligned} 0 = & \quad \epsilon^2((A_y + B_y)^2 + (A_x + B_x)^2 - d^2) \\ & + \quad \epsilon(2)((A_x + B_x)(A_z B_x - A_x B_z) \\ & \quad\quad -(A_y + B_y)(A_y B_z - A_z B_y) - d^2 A_z) \\ & + \quad (A_y B_z - A_z B_y)^2 + (A_z B_x - A_x B_z)^2 \\ & \quad\quad +(A_x B_y - A_y B_x)^2 - d^2(B_x^2 + B_y^2 + B_z^2) \end{aligned}$$



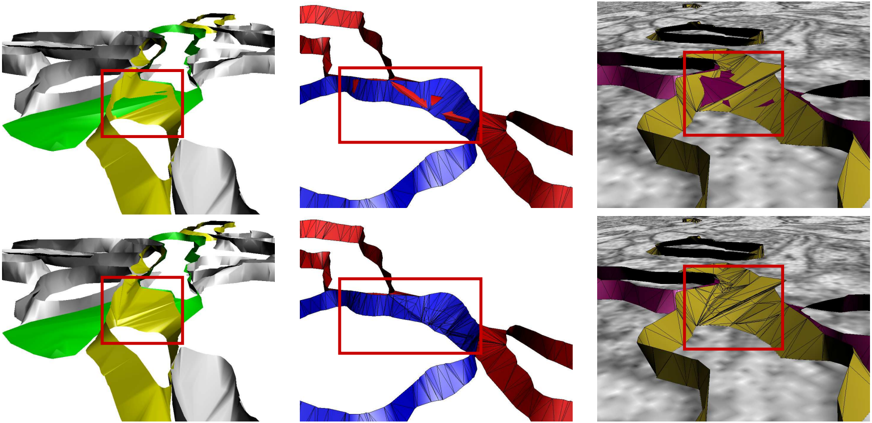


Theorem

$\epsilon < |p^g - \mathscr{Z}(p^g)|$ and $\epsilon < |p^y - \mathscr{Z}(p^y)|$

Idea of proof: as points approach original contours, which are separated by d, the chords will be separated by at least d in the limit.

Results



Conclusions and notes

- Algorithm is $O(n^2)$ where n is the number of tiles.
  - Average case is closer to $n \log n$ complexity of sweep line algorithm as large majority of 2D intersections are not conflict points.
- Original contours remain unchanged – only makes changes in interpolated data between slices
- Topologically correct and water tight
- Generates large number of extra triangles in intersecting regions

Outline

Error bounds - motivation

When performing FEM or BEM simulations, two error terms contribute:

- surface approximation
- numerical solution of PDE

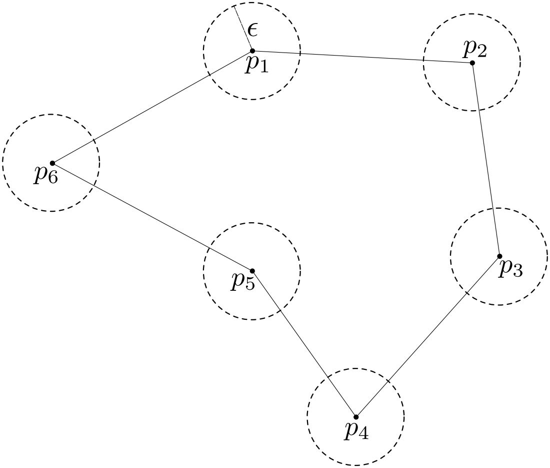
If the surface approximation error dominates, then there is no benefit to discretizing more finely for an improved solution.

Why is there so little discussion of error bounds in the literature? Some ideas:

- Bounding image segmentation error is difficult
- It isn't considered important for visualization
- Optimizing certain criteria is sufficient. Consider the following statements:
  - "[Our algorithm has] a variety of possible options for choosing optimizing criteria" [Fuchs et al., 1977].
  - "Various conditions may be imposed: maximize the volume, minimize the surface, minimize the edge length or angles..." [Boissonnat & Geiger, 1992].
  - "...the minimum surface optimizing algorithm..." [Bajaj et al., 1996].
  - "Optimality: creating the best surface, in terms that are subjective, but well-defined for each solution" [Barequet & Vaxman, 2007].
  - "[Our algorithm] tends in practice to minimize the surface area of the reconstruction..." [Barequet & Vaxman, 2009].
- It's too hard

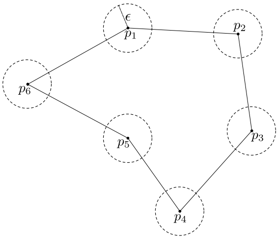
Error bounds – approach

Fortunately we have some points that are known to be on the surface (or at least close)



Volume error bound

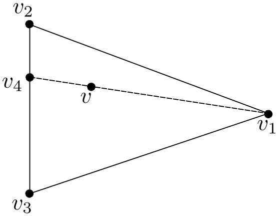
- Goal #1: bound the error of the surface area and volume
- Approach: start by bounding perimeter and area of original contours
  - Perimeter error bounded by $2n\epsilon$ (derivation in slide A2)
  - Area error bounded by $2\epsilon\Gamma(P_{rep})$ where $\Gamma(P_{rep})$ is the perimeter (derivation in slide A3)
- We hope to bound the entire surface area and volume using these bounds and knowledge of our reconstruction properties
  - A promising approach: area and volume derivatives [Bryant et al., 2004]
- These bounds are useful for "cable equation" simulations



Surface error bound

- Goal #2: bound the deviation of the reconstructed surface from the true surface
- Approach: start by bounding deviation of a single triangle
  - Many triangles have all 3 vertices on contours
  - Assuming vertices are exact, error at a point given by its barycentric coordinates $(\lambda_1, \lambda_2, \lambda_3)$ is

$$\frac{\lambda_1 \lambda_2 d_{12}^2 + \lambda_1 \lambda_3 d_{13}^2 + \lambda_2 \lambda_3 d_{23}^2}{2} f''(\xi)$$



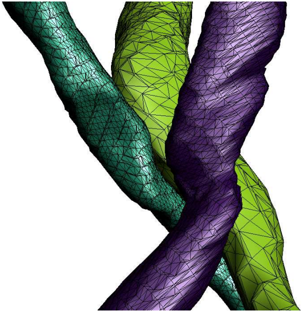
Surface error bound

- Single triangle: $\frac{\lambda_1\lambda_2 d_{12}^2+\lambda_1\lambda_3 d_{13}^2+\lambda_2\lambda_3 d_{23}^2}{2} f''(\xi)$
- Problem A: how do we estimate the second derivative?
  - We have a lot of 2D information (in the form of contours), but we don't know the normals at those points
  - This is biological data, so we'll have to determine an acceptable feature size
- Problem B: what about triangles with only two or even one vertex on the surface?
  - Use Taylor's theorem? Bound could then be unusably loose.

Outline

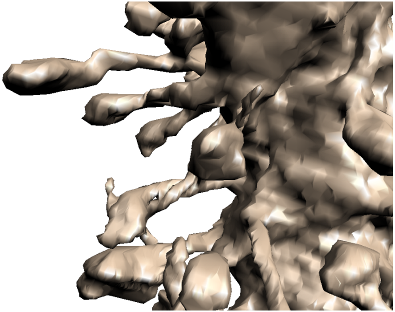
Mesh improvement

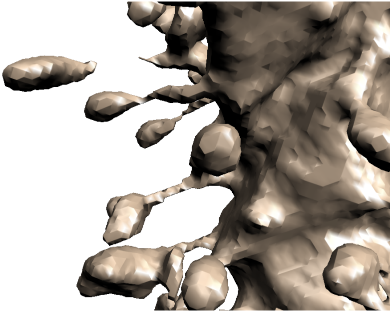
Current mesh improvement algorithm has no intersection guarantees



In practice, this is not a problem because . . .

Current mesh improvement algorithm causes erosion





This will damage whatever error bounds our reconstruction algorithm gives.

Mesh improvement

Desired: mesh improvement algorithm that respects constraints, which could be

- spatial scaffolding
- volume/surface area bounds

Outline

The problem

Problem statement

Given planar contours in parallel slices, build analysis-ready 3D surfaces

An analysis-ready surface

- is water-tight
- has oriented surface normals
- is non-intersecting
- has manifold edges and vertices
- is composed of low aspect ratio triangles
- is topologically correct
- is close to the true surface

List of TODOs

TODO:

- Determine error bounds for current reconstruction algorithm
  - These include volume, surface area, and deviation from true surface
  - Hopefully these will be general enough to apply to other algorithms
- Modify mesh improvement algorithm
  - Must respect error bounds through some kind of constraint set
  - Must be adaptive for quality simulations
- Look at better guarantees of topological correctness

Thanks!

References

[Alberts et al., 2009] Alberts, J., Edwards, J., Johnston, J. & Ferrin, J. (2009).
In Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series vol. 7332, p. 43,.

[Bajaj et al., 1996] Bajaj, C., Coyle, E. & Lin, K. (1996).
Graph. Models Image Process. 58, 524–543.

[Bajaj & Goswami, 2008] Bajaj, C. & Goswami, S. (2008).
In Proceedings of the 2008 ACM symposium on Solid and physical modeling pp. 193–202, ACM.

[Barequet & Sharir, 1994] Barequet, G. & Sharir, M. (1994).
In Computer Vision and Image Understanding pp. 93–102,.

[Barequet & Vaxman, 2007] Barequet, G. & Vaxman, A. (2007).
In SPM '07: Proceedings of the 2007 ACM symposium on Solid and physical modeling pp. 97–107, ACM, New York, NY, USA.

[Barequet & Vaxman, 2009] Barequet, G. & Vaxman, A. (2009).
In SGP '09: Proceedings of the Symposium on Geometry Processing pp. 1327–1337, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland.

[Bermano et al., 2011] Bermano, A., Vaxman, A. & Gotsman, C. (2011).
ACM Transactions on Graphics 30.
Accepted for Publication.

[Boissonnat, 1988] Boissonnat, J. (1988).
Computer Vision, Graphics, and Image Processing 44, 1–29.

[Boissonnat & Geiger, 1992] Boissonnat, J. & Geiger, B. (1992).
In Proceedings of SPIE vol. 964,.

References]References

[Bryant et al., 2004] Bryant, R., Edelsbrunner, H., Koehl, P. & Levitt, M. (2004).
Discrete and Computational Geometry 32, 293–308.

[Christiansen & Sederberg, 1978] Christiansen, H. & Sederberg, T. (1978).
SIGGRAPH Comput. Graph. 12, 187–192.

[Cline & Edwards, 2011] Cline, A. & Edwards, J. (2011).
Bounding the error of interpolation on a triangle.
private communication.

[Edwards & Bajaj, 2010] Edwards, J. & Bajaj, C. (2010).
In Proceedings of the 14th ACM Symposium on Solid and Physical Modeling pp. 51–60, ACM ACM.

[Fiala et al., 2002] Fiala, J., Spacek, J. & Harris, K. (2002).
Brain research reviews 39, 29–54.

[Fuchs et al., 1977] Fuchs, H., Kedem, Z. & Uselton, S. (1977).
Commun. ACM 20, 693–702.

[Turk & O'Brien, 1999] Turk, G. & O'Brien, J. (1999).
In SIGGRAPH'99 pp. 335–342,.

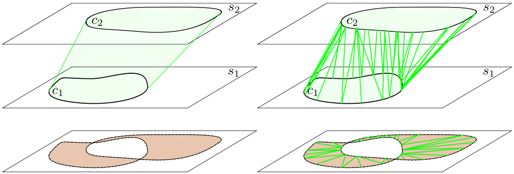
[Zhang et al., 2005] Zhang, Y., Bajaj, C. & Sohn, B. (2005).
Computer methods in applied mechanics and engineering 194, 5083–5106.

A1 - Moving conflict points

Our algorithm will remove intersections without causing other intersections.

Theorem

Moving any conflict point p^g in the direction of its penumbral contour will not generate any additional conflict points among any pair of components.



Idea of proof: as a point moves toward its penumbral contour it won't enter any component because only two components can intersect in a given penumbra.

Perimeter (rough bounds suggested by Alex Rand): All indices are modulo n. Let the perimeter be

$$f(p_1, ..., p_n) = f(P) = \sum_{i=1}^{n} \|p_{i+1} - p_i\| = \sum_{i=1}^{n} \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2}$$

such that $\forall p_i, \|p_i - v_i\| \leq \epsilon$.

$$\begin{aligned}\nabla f(P) &= \sum_{i=1}^{n} \frac{d}{dx_i}\|p_{i+1} - p_i\|\hat{k}_{2i-1} + \frac{d}{dy_i}\|p_{i+1} - p_i\|\hat{k}_{2i} + \frac{d}{dx_{i+1}}\|p_{i+1} - p_i\|\hat{k}_{2i+1} + \frac{d}{dy_{i+1}}\|p_{i+1} - p_i\|\hat{k}_{2i+2} \\ &= \sum_{i=1}^{n} \frac{-1}{\|p_{i+1} - p_i\|}\langle p_{i+1} - p_i\rangle + \frac{1}{\|p_{i+1} - p_i\|}\langle p_{i+1} - p_i\rangle\end{aligned}$$

where $(\hat{k}_1, \ldots, \hat{k}_{2n})$ is the standard basis for $\mathbb{R}^{2n}$. This results in $2n$ unit vectors (in n different planes in $\mathbb{R}^{2n}$). By the triangle inequality,

$$\|\nabla f(P)\| \leq 2n$$

Let $V = (v_1, v_2, \ldots, v_n)$ be the vertices of P_{rep}. By Taylor's theorem,

$$f(V) = f(P) + (f(V) - f(P))\|\nabla f(\Xi)\|$$

As stated above, $\|\nabla f(\Xi)\| \leq 2n$ and $\|f(V) - f(P)\| \leq \epsilon$. Thus the perimeter error is bounded by $2n\epsilon$.

Tighter bounds could be derived using, for example, Lagrange multipliers, but this is more difficult and messy.

Area: All indices are modulo n. Using the cross-product, the area of a triangle $\Delta p_1 p_2 p_3$ is given as

$$A = \frac{1}{2}|(p_2 - p_1) \times (p_3 - p_1)| = \frac{1}{2}\left[(x_2 - x_1)(y_3 - y_1) - (y_2 - y_1)(x_3 - x_1)\right]$$

The area of a polygon with n vertices is given as the sum of the areas of the triangles of the form $\Delta p_1 p_i p_{i+1}$, $i \neq 1$ and $i \neq n$. Thus the area of the polygon is

$$\begin{aligned} f(p_1, \ldots, p_n) = f(P) = & \frac{1}{2}[(x_2 - x_1)(y_3 - y_1) - (y_2 - y_1)(x_3 - x_1) \\ & +(x_3 - x_1)(y_4 - y_1) - (y_3 - y_1)(x_4 - x_1) \\ & + \ldots \\ & +(x_{n-1} - x_1)(y_n - y_1) - (y_{n-1} - y_1)(x_n - x_1)] \\ = & \quad \frac{1}{2}\sum_{i=2}^{n-1}(x_i - x_1)(y_{i+1} - y_1) - (y_i - y_1)(x_{i+1} - x_1) \end{aligned}$$

Let $f_i(P) = (x_i - x_1)(y_{i+1} - y_1) - (y_i - y_1)(x_{i+1} - x_1)$.

$$\nabla f(P) = \sum_{i=1}^{n} \frac{d}{dx_i} f_i \hat{k}_{2i-1} + \frac{d}{dy_i} f_i \hat{k}_{2i}$$

where $(\hat{k}_1, \ldots, \hat{k}_{2n})$ is the standard basis for $\mathbb{R}^{2n}$. ...

$$\frac{d}{dx_1} f_i(P) = (y_1 - y_{i+1}) + (y_i - y_1) = (y_i - y_{i+1})$$

$$\frac{d}{dx_1} f(P) = \sum_{i=2}^{n-1} (y_i - y_{i+1}) = y_2 - y_n \tag{1}$$

Similarly,

$$\frac{d}{dy_1} f(P) = \sum_{i=2}^{n-1} (x_{i+1} - x_i) = x_n - x_2 \tag{2}$$

At other vertices,

$$\begin{aligned} \frac{d}{dx_i} f(P) &= \begin{cases} \frac{d}{dx_i} f_j(P) & i-1 \leq j \leq i \\ 0 & \text{elsewhere} \end{cases} \\ &= \frac{d}{dx_i} f_{i-1}(P) + \frac{d}{dx_i} f_i(P) \\ &= (y_{i+1} - y_1) - (y_{i-1} - y_1) \\ &= y_{i+1} - y_{i-1} \end{aligned}$$

Similarly,

$$\frac{d}{dy_i} f(P) = x_{i-1} - x_{i+1}$$

Combining with equations (1) and (2),

$$\nabla f(P) = \sum_{i=1}^{n} (y_{i+1} - y_{i-1}) \hat{k}_{2n-1} + (x_{i-1} - x_{i+1}) \hat{k}_{2n} \tag{3}$$

A3 - Area error bound derivation (cont.)

We can bound the length of the gradient:

$$\|\nabla f(P)\| = \left(\sum_{i=1}^{n}(y_{i+1} - y_{i-1})^2 + (x_{i-1} - x_{i+1})^2\right)^{1/2} \leq \sum_{i=1}^{n} \|p_{i+1} - p_{i-1}\| \leq 2\Gamma(P_{rep})$$

where $\Gamma(P_{rep})$ is the perimeter of polygon P_{rep}. The last inequality is due to the triangle inequality. Thus we can bound the error in the area of P_{rep} to be $\leq 2\epsilon\Gamma(P_{rep})$.

See left figure. We're looking for the error at v. Let the barycentric coordinates of v be $(\lambda_1, \lambda_2, \lambda_3)$. Preliminaries:

$$\begin{aligned} d_{14}^2 &= d_{13}^2 + d_{34}^2 - 2(v_1 - v_3)\cdot(v_4 - v_3) && \text{(law of cosines)} \\ &= d_{13}^2 + \frac{\lambda_2^2 d_{23}^2}{(\lambda_2+\lambda_3)^2} - \frac{2\lambda_2}{\lambda_2+\lambda_3}\left(\frac{d_{13}^2 + d_{23}^2 - d_{12}^2}{2}\right) && \text{(law of cosines again)} \\ &= \frac{\lambda_2+\lambda_3}{(\lambda_2+\lambda_3)^2}\left(\lambda_2 d_{12}^2 + \lambda_3 d_{13}^2 - \frac{\lambda_2\lambda_3 d_{23}^2}{(\lambda_2+\lambda_3)}\right) \end{aligned}$$

Then the error is (see right figure):

$$\begin{aligned} \epsilon(v) &= \lambda_4 \epsilon(v_4) + \epsilon'(v) \\ &= \lambda_4 \frac{\lambda_2\lambda_3 d_{23}^2}{2(\lambda_2+\lambda_3)^2} f_{23}''(\xi_{23}) + \frac{\lambda_1(\lambda_2+\lambda_3)^2}{2(\lambda_2+\lambda_3)^2}\left(\lambda_2 d_{12}^2 + \lambda_3 d_{13}^2 - \frac{\lambda_2\lambda_3 d_{23}^2}{(\lambda_2+\lambda_3)}\right) f_{14}''(\xi_{14}) \\ &= \frac{\lambda_1\lambda_2 d_{12}^2 + \lambda_1\lambda_3 d_{13}^2 + \lambda_2\lambda_3 d_{23}^2}{2} f''(\xi) \end{aligned}$$

where $f_{14}''(\xi_{14})$ is the second derivative in the direction of the vector $v_4 - v_1$ at an unknown point ξ_{14}. $f_{23}''(\xi_{23})$ is defined similarly. $f''(\xi)$ is the second directional derivative at an unknown point ξ in an unknown direction.