ACM Symposium on Solid Modeling and Applications (2004)
G. Elber, N. Patrikalakis, P. Brunet (Editors)

Contour Interpolation with Bounded Dihedral Angles[†]

Sergey Bereg[‡] & Minghui Jiang[§] & Binhai Zhu[¶]

Abstract

In this paper, we present the first nontrivial theoretical bound on the quality of the 3D solids generated by any contour interpolation method. Given two arbitrary parallel contour slices with n vertices in 3D, let α *be the smallest angle in the constrained Delaunay triangulation of the corresponding 2D contour overlay, we present a contour interpolation method which reconstructs a 3D solid with the minimum dihedral angle of at least* $\frac{\alpha}{8}$*. Our algorithm runs in* $O(n \log n)$ *time where n is the size of the contour overlay.*

We also present a heuristic algorithm that optimizes the dihedral angles of a mesh representing a surface in 3D.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Curve, surface, solid, and object representations

1. Introduction

Polyhedral surface reconstruction from parallel slices of two-dimensional contours is an important problem in medical sciences, geographical information systems and computational biology, etc. A method called *contour interpolation* is widely used in practice, which takes parallel slices of two-dimensional contours delineating the boundary of solid and empty regions and constructs a 3D solid by adding some Steiner points properly [BBX95, BCL96, GOS96, Kep75, OPC96].

The general idea of the contour interpolation is as follows. First, the symmetric difference of contour overlay between two adjacent slices, which is a group of interior-disjoint polygons with vertices from both slices, is computed. (The polygons may have holes and be nested). For example, the symmetric difference of two polygons P_1 and P_2 from two adjacent slices in Fig. 1 consists of four polygons $S_i, i = 1, \ldots, 4$. Second, the polygons are triangulated, possibly with Steiner points (note that without using Steiner points it might not be possible to reconstruct a 3D solid from 2D slices). Finally, a height is assigned to each vertex of the triangulation: vertices and Steiner points on polygon boundaries are assigned the heights of their corresponding slices; internal Steiner points are assigned heights in between by a height interpolation scheme. After the height interpolation, the triangulation is then lifted up (each vertex is lifted to its own height), thus reconstructing a patch of the polyhedral surface in three-dimensional space.

There has been extensive research on contour interpolation using different kinds of heuristic algorithms, some of which handle the general case without limiting the number and topology of contours in the slices. In some recent works [BGLSS03, OPC96], positions and heights of Steiner points are determined by heuristics; reconstructed surfaces appear smooth in experiments, but there is no theoretical bounds on the quality of the reconstructed solid, e.g., the dihedral angles between neighbouring triangles. Our goal is to improve the contour interpolation method to reconstruct polyhedral surfaces with bounded dihedral angles. Similar to previous works, our method also consists of a symmetric difference computation step, a triangulation step and a height interpolation step. As the symmetric difference computation step is the same as [BGLSS03, OPC96], we assume that contour overlay is given as input in our problem.

1.1. Related work

Surface reconstruction studied from contours has been studied by many researchers since it finds applications in many areas. Fuchs *et al.* [FKU77] consider the problem of minimizing the total surface area occupied by the triangles. They translated the problem into a problem on a toroidal directed graph: The problem is thus reduced to the search for certain minimum cost cycles in this graph and a fast algorithm for finding such cycles is developed.

The methods in [Boi88, Gei93] are based on Delaunay triangulation constructed in every slice. Cheng and Dey [CD99] improved the post-processing step and avoided the computation of three dimensional Delaunay tetrahedra.

The quality of mesh generation based on the dihedral angles is concerned in the papers [BCER95, CDE*00, MV00]. The differ-

† This research is supported by NSF CARGO Grant DMS-0138065 and NSF EPSCOR Visiting Scholar's Program.

‡ Department of Computer Science, University of Texas at Dallas, Box 830688, Richardson, TX 75083, USA. E-mail: besp@utdallas.edu

§ Department of Computer Science, Montana State University, Bozeman, MT 59717-3880, USA. E-mail: jiang@cs.montana.edu

¶ Department of Computer Science, Montana State University, Bozeman, MT 59717-3880, USA. E-mail: bhz@cs.montana.edu

ence between their approaches and the present one is that they considered the simplicial mesh modeling a volume in the space (they also consider higher dimensions) and we focus on surface reconstruction. For example, in 3D their problem is for tetrahedra whose dihedral angles are optimized (the *sliver* is an example of a tetrahedron that can have arbitrarily small dihedral angles).

A related research has been done for generating smooth surfaces [CMN98, HSS03, SSBT01]. Klein *et al.* [KSS00] considered the problem of reconstructing triangulated surfaces and its simplification.

The paper is organized as follows. In Section 2, we present the details of our algorithm and its analysis. In Section 3, we describe a heuristic algorithm for improving the dihedral angles of a mesh. In Section 4, we conclude the paper with some open problems.

2. Algorithm and Analysis

In this section, we present the technical details of the algorithms and the proofs. Consider two adjacent slices of a solid body. Suppose that the polygons in one slice are colored red and the polygons of the other slice are blue. Let P be a polygon of the symmetric difference, see for example Fig. 1. Not any Steiner triangulation of P can be used for contour interpolation. We consider restricted Steiner triangulations and formulate the following two-dimensional problem for the triangulation step.

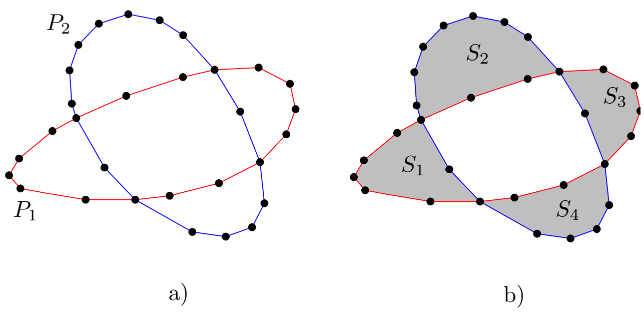

Figure 1: *a) Overlay of two polygons P_1 and P_2, b) Symmetric difference.*

STCC Problem Given P, a simple polygon possibly with holes, with each boundary edge colored either red or blue, the problem *Steiner Triangulation with Color Constraints (STCC)* is to find a Steiner triangulation for polygon P satisfying the following constraints:

1. Steiner points can be added only in the interior of the polygon and Steiner points are of green color.
2. Every internal edge (new created edge in the interior of the polygon) has at least one green vertex.
3. If two triangles share an edge with two green vertices, then their two unshared vertices cannot be both red or both blue.

We now describe an algorithm that solves the STCC problem and produces triangulations with the additional property that no angle is smaller than a constant. Our algorithm is incremental and any edge not satisfying constraint 2 or 3 is called *illegal*. The algorithm consists of two simple steps:

1. Run Chew's constrained Delaunay algorithm [Che89] on P to obtain a triangulation T_1.

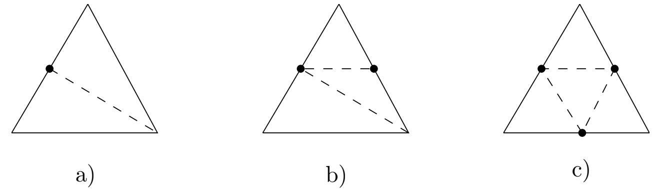

Figure 2: *Illustration for the STCC algorithm. Step 2b. The green vertices are bold and new edges are dashed.*

2. Modify T_1 to meet color constraints (see Fig. 2):
   a. For each illegal edge violating constraint 2, add a green Steiner point at the midpoint of the edge.
   b. For each triangle that has at least one illegal edge, do the following:
      i. If it has one illegal edge, add an edge to connect the Steiner point at the midpoint of the illegal edge to the vertex opposed to the illegal edge, see Fig. 2 a);
      ii. If it has two illegal edges, first add an edge to connect the two Steiner points, then triangulate the resulting trapezoid arbitrarily, see Fig. 2 b);
      iii. If it has three illegal edges, add three edges to connect the three Steiner points into a triangle, see Fig. 2 c).

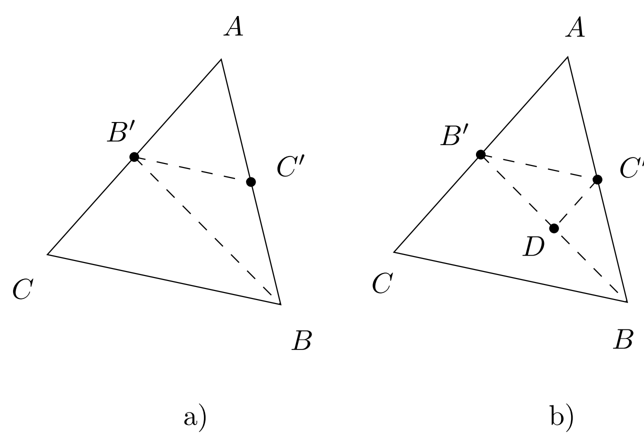

Figure 3: *Illustration for the STCC algorithm. Step 2c. A and B are the blue vertices and C is the red vertex. AB and AC are internal edges and BC is the boundary edge. The green vertices are bold and new edges are dashed.*

   c. For each illegal green edge violating the constraint 3, pick either triangle sharing the illegal edge (say t), pick either edge of the triangle t except the illegal green edge, and add a green Steiner point at the midpoint of the edge (say e). Connect the Steiner point to the opposite vertices in triangles sharing e. Fig. 3 illustrates an example of this fixing: the triangle ABC is divided into three triangles in Fig. 3 a), and the green edge $B'C'$ is fixed by introducing the green vertex D.

   Output T_2, the triangulation after these modifications.

We first prove the following lemma related to our algorithm.

Lemma 1 Given a triangle, if we connect the midpoint of one edge to the vertex opposed to the edge, then the minimum angle of the two sub-triangles is at least $\frac{1}{4}$ of the minimum angle of the original triangle.

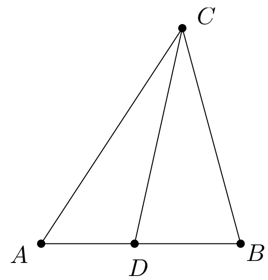

Figure 4: *Illustration for the proof of Lemma 1.*

Proof Given $\mathcal{S}$, a set of triangles, we use $\min_{\measuredangle} \mathcal{S}$ to denote the minimal angle of the triangles in $\mathcal{S}$. For brevity, we use AB to denote the length of segment $\overline{AB}$.
In $\triangle ABC$ (see Fig. 4), point D is the midpoint of segment $\overline{AB}$. Without loss of generality, we assume that $AC \geq BC$. We separate our analysis into two cases: (1) $AB \geq BC$; (2) $AB < BC$.
In case (1), $AB \geq BC$. $\angle BAC$ is the minimum angle of $\triangle ABC$; either $\angle BAC$ or $\angle ACD$ is the minimum angle of $\triangle ACD$ and $\triangle BCD$. If $\angle BAC \leq \angle ACD$, then

$$\frac{\min_{\measuredangle}\{\triangle ACD, \triangle BCD\}}{\min_{\measuredangle}\{\triangle ABC\}} = \frac{\angle BAC}{\angle BAC} = 1 > \frac{1}{4}.$$

Otherwise, we have

$$\frac{\min_{\measuredangle}\{\triangle ACD, \triangle BCD\}}{\min_{\measuredangle}\{\triangle ABC\}} = \frac{\angle ACD}{\angle BAC}.$$

Let $\alpha = \angle ACD$ and $\beta = \angle BAC$. In $\triangle ACD$, we have

$$\frac{\sin\alpha}{\sin\beta} = \frac{AD}{CD} \geq \frac{AD}{BD+BC} \geq \frac{AD}{BD+AB} = \frac{1}{3}.$$

Since $\beta \leq 60^\circ$, we have

$$\begin{aligned} \frac{\sin\beta}{3} &= \frac{4\cos\frac{\beta}{4}(2\cos^2\frac{\beta}{4}-1)}{3}\sin\frac{\beta}{4} \\ &\geq \frac{4\cos 15^\circ(2\cos^2 15^\circ - 1)}{3}\sin\frac{\beta}{4} \geq \sin\frac{\beta}{4}. \end{aligned}$$

Therefore, $\sin\alpha \geq \frac{\sin\beta}{3} \geq \sin\frac{\beta}{4}$, this implies that $\alpha \geq \frac{\beta}{4}$, which further implies

$$\frac{\min_{\measuredangle}\{\triangle ACD, \triangle BCD\}}{\min_{\measuredangle}\{\triangle ABC\}} = \frac{\alpha}{\beta} \geq \frac{1}{4}.$$

In case (2), $AB < BC$. $\angle ACB$ is the minimum angle of $\triangle ABC$; $\angle ACD$ is the minimum angle of $\triangle ACD$ and $\triangle BCD$. Since $\triangle ACD$ and $\triangle BCD$ have the same area, we have

$$\frac{1}{2}AC \cdot CD \cdot \sin\angle ACD = \frac{1}{2}BC \cdot CD \cdot \sin\angle BCD.$$

This implies

$$\frac{\sin\angle ACD}{\sin\angle BCD} = \frac{BC}{AC}.$$

Let $\alpha = \angle ACD$ and $\beta = \angle ACB$. We have,

$$\frac{\sin\alpha}{\sin(\beta-\alpha)} = \frac{\sin\angle ACD}{\sin\angle BCD} = \frac{BC}{AC} \geq \frac{BC}{AB+BC} \geq \frac{BC}{BC+BC} = \frac{1}{2}.$$

This implies that

$$\frac{\sin\alpha}{\sin\beta} \geq \frac{\sin\alpha}{\sin(\beta-\alpha)+\sin\alpha} = \frac{1}{\frac{\sin(\beta-\alpha)}{\sin\alpha}+1} \geq \frac{1}{3}.$$

Following similar argument as in case (1), we have

$$\frac{\min_{\measuredangle}\{\triangle ACD, \triangle BCD\}}{\min_{\measuredangle}\{\triangle ABC\}} \geq \frac{1}{4}.$$

□

The first good news is that our algorithm adds $O(n)$ Steiner points. We have the following theorem about our algorithm.

Theorem 2 Let α be the smallest angle in the constrained Delaunay triangulation T_1 of P. Our algorithm produces a STCC with no angle smaller than $\frac{\alpha}{16}$.

Proof By adding a green Steiner point at the midpoint of each illegal edge violating constraint 2, the illegal edge is split into two legal sub-edges; any new edge introduced by a green Steiner point has at least one green vertex so it clearly satisfies constraint 2. With the addition of green Steiner points in step 2(c) of the algorithm, each illegal edge violating constraint 3 becomes legal because one of the two triangles sharing the edge now has three green vertices. Any new edge introduced by this Steiner point satisfies constraint 3: if a new edge has two green vertices, then one of the two triangles sharing the edge must have three green vertices. Therefore, our algorithm solves the STCC problem.
In the first step of our algorithm, we use the constrained Delaunay triangulation T_1, which maximizes the smallest angle among all triangulations of P. Let α be the minimum angle of the triangulation T_1. From Lemma 1, it is easy to see that the final triangulation T_2 produced by our algorithm has no angle smaller than a constant $\frac{\alpha}{16}$: after step (2)(b), the smallest angle is at least $\frac{\alpha}{4}$ and after step (2)(c), the smallest angle is at least $\frac{\alpha}{16}$ in the worst case. □

We now describe our improved contour interpolation algorithm for surface reconstruction. The only difference between our algorithm and other contour interpolation algorithms are the selection and height assignment of Steiner points.

1. For each polygon compute the symmetric difference of a contour overlay, formulate a corresponding STCC problem: edges from upper layer are colored red; edges from lower layer are colored blue.
2. Solve the STCC problems, and obtain Steiner triangulation T_2 with guaranteed lower bound on minimal angle.
3. For each vertex in the Steiner triangulation T_2, assign height 1 if it is red, height 0 if blue, and height 0.5 if green. Lift the triangulation up and obtain a three-dimensional triangulated surface T_3.

We have the following theorem about our algorithm.

Theorem 3 Let α be the smallest angle in the constrained Delaunay triangulation T_1 of P. The triangulated three-dimensional surface T_3 generated by our contour interpolation algorithm has a minimum dihedral angle of $\frac{\alpha}{8}$.

Proof Let $\triangle ABC$ and $\triangle ABD$ be a pair of neighbouring triangles sharing edge AB. We show that the dihedral angle around edge AB is bounded by a constant. Because of constraint 2, either vertex A or vertex B must be green. Without loss of generality, assume vertex A is green. We now consider two cases.

Case 1: vertex B is also green. Because of constraint 3, vertices C and D cannot be both higher or both lower than edge AB; it is easy to see that the dihedral angle around edge AB is at least 90°.
Case 2: vertex B is red. (The other case, where vertex B is blue, is symmetric.) Because of constraint 2, neither C nor D can be blue. We consider the following subcases (symmetric subcases are omitted).
Case 2a: A is green; B is red; C is green; D is green. The extreme case happens when edge AB is perpendicular to the plane determined by $\triangle CAD$; the dihedral angle around edge AB is at least $\angle CAD$, which is at least twice the minimum angle in T_2, which is $2 \times \frac{\alpha}{16} = \frac{\alpha}{8}$.
Case 2b: A is green; B is red; C is red; D is red. This case is similar to case 2a; the dihedral angle is at least $\angle CBD$, which is at least $\frac{\alpha}{8}$.
Case 2c: A is green; B is red; C is green; D is red. If we make D green instead, the dihedral angle becomes smaller and this case changes to case 2a; therefore, the dihedral angle here is also at least $\frac{\alpha}{8}$.
In summary, the minimum dihedral angle between neighbouring triangles in triangulated surface T_3 is at least a constant, $\min\{90^\circ, \frac{\alpha}{8}\} = \frac{\alpha}{8}$. □

Assume that P is already given and P has n vertices, it is easy to see that our algorithm runs in $O(n \log n)$ time, which is dominated by the step of computing the constrained Delaunay triangulation T_1. Notice that when the size of the input is in terms of that of the 2D slices, then the overall complexity of the algorithm is dominated by the computation of the contour overlay, which is quadratic in the worst case. In real applications, such a quadratic bound is very unlikely.

3. Heuristic Algorithm

The algorithm suggested in the previous section generates a mesh with a guaranteed quality. In this Section we describe a heuristic algorithm that is more practical although it does not guarantees a bound on dihedral angles of the produced mesh. The heuristic algorithm is more general and can be used for improving any mesh (not necessarily generated by contour interpolation).

The idea of the algorithm is to attempt to flatten dihedral angles small enough by inserting Steiner points. An example illustrated in Fig. 5 demonstrates the procedure. Suppose that the edge $p_i p_j$ defines a small dihedral angle α, the angle between two facets $p_i p_j p_a$ and $p_i p_j p_b$. We rotate the mesh so that the edge $p_i p_j$ looks as a ridge on the mesh assuming that we view it from $z = +\infty$ ($p_i p_j$ is depicted as bold in Fig. 5 a)). Suppose that the edges $p_i p_a, p_i p_b, p_j p_a$ and $p_j p_b$ look as valley edges. Clearly, the insertion a Steiner point p_k slightly below the segment $p_i p_j$ will generate four new edges $p_k p_i, p_k p_j, p_k p_a$ and $p_k p_b$. The edges $p_k p_i$ and $p_k p_j$ are the ridges and the edges $p_k p_a$ and $p_k p_b$ are the valleys. The dihedral angle between the facets $p_i p_k p_a$ and $p_i p_k p_b$ is smaller than the angle α. Similarly the dihedral angle between the facets $p_j p_k p_a$ and $p_j p_k p_b$ is smaller than α. This is true for any choice of the point p_k in the tetrahedron $p_i p_j p_a p_b$. The two new dihedral angles defined by the edges $p_k p_a$ and $p_k p_b$ can be arbitrarily flat (close to 180°) if p_k is chosen close to $p_i p_j$.

The algorithm selects a set of edges to remove. Note that, if two edges have a common vertex, then the removal of one edge changes the dihedral angle defined by the other edge, see Fig. 5. Therefore we select the edges so that they do not have a common vertex. The algorithm has the following step.

1. For all edges e of the mesh, compute the dihedral angles $dih\_angle(e)$.
2. Sort the edges $e_1, e_2, \ldots, e_m$ in decreasing order of the dihedral angles

$$dih\_angle(e_1) \le dih\_angle(e_2) \le \cdots \le dih\_angle(e_m).$$

3. $E_{sel} = \emptyset$. Initialize the set of selected edges.
   Label each facet f as *free*, i.e. $label(f) = FREE$.
4. For $i = 1$ to m do the following.
   a. Let f_1 and f_2 be two facets incident to e_i.
   b. If $label(f_1) = FREE$ and $label(f_2) = FREE$, then add e_i to E_{sel}.
5. For each $e \in E_{sel}$, find a Steiner point, remove e_i and update the mesh (add one vertex, four edges and four facets).

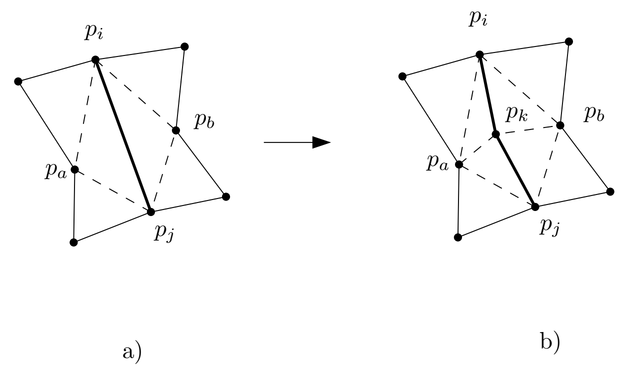

Figure 5: *Insertion of a new vertex p_k. The ridge edges are bold and valley edges are dashed.*

There can be various methods of choosing the Steiner point in the step 5. One can use the following general approach. Let γ be a parameter such that the new dihedral angles must be at least γ. Let $e_i = (p_i p_j)$ and f_1, f_2, f_3 and f_4 be the faces incident to the edges $p_a p_i, p_i p_b, p_b p_j$ and $p_j p_a$, respectively. The constraint $dih\_angle(p_a p_i) \le \gamma$ means that p_k must be in between two planes, see Fig. 6. We also can express the constraint $dih\_angle(p_a p_k) \le \gamma$ algebraically and z-coordinate of $p_k = (x, y, z)$ (assuming that the plane $p_a p_i p_k$ is horizontal) must be sandwiched between two functions $F'_a(x,y) \le z \le F''_a(x,y)$. Similarly we can express the constraints $dih\_angle(p_l p_k) \le \gamma, l \in \{i, b, j\}$ using functions $F'_i, F''_i, F'_b, F''_b, F'_j, F''_j$. Solving these equations we can find a point p_k in the feasible domain if it is not empty.

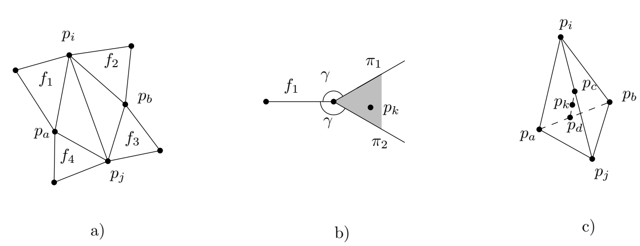

Figure 6: *a) Faces f_1, f_2, f_3 and f_4. b) Planes π_1 and π_2, c) Steiner point p_k.*

We implemented a rather simpler algorithm for choosing the Steiner point p_k. Let p_c be the mid-point of the segment $p_i p_j$ and p_d be the mid-point of the segment $p_a p_b$, see Fig. 6 c). We place the Steiner point p_k on the segment $p_c p_d$. The first try is the mid-point of $p_c p_d$ and, if some dihedral angles of the edges $p_a p_i, p_i p_b, p_b p_j$ and $p_j p_a$ are decreased, we check the points $t * p_c + (1-t) * p_d$ where $t = i/10, i = 6,7,8,9$. We tested the program on neuron data together with contour interpolation, see Fig. 7 and Fig. 8. The Figure 9 illustrates the selection of the edges E_{sel} in the step 4. The heuristic algorithm increases the dihedral angles locally. The result is that the average angle of 3095 edges is increased by 6.791177°.

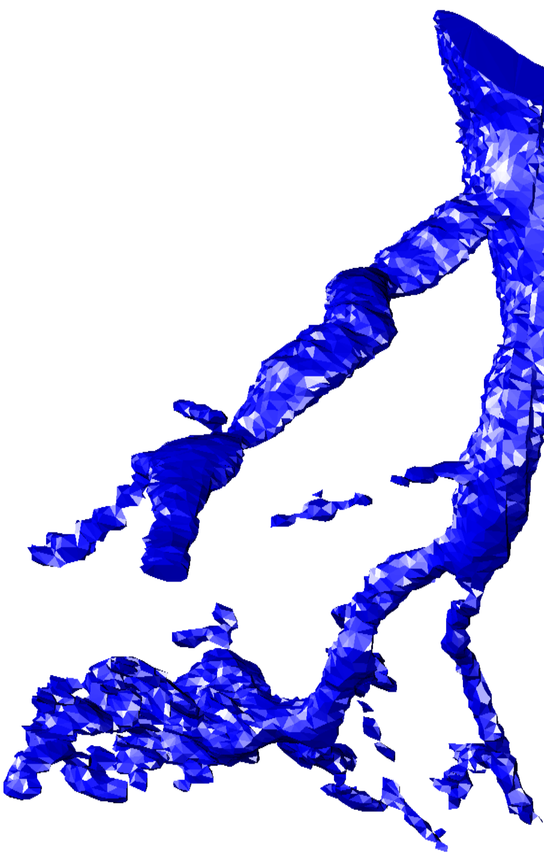

Figure 7: *Contour interpolation for neuron.*

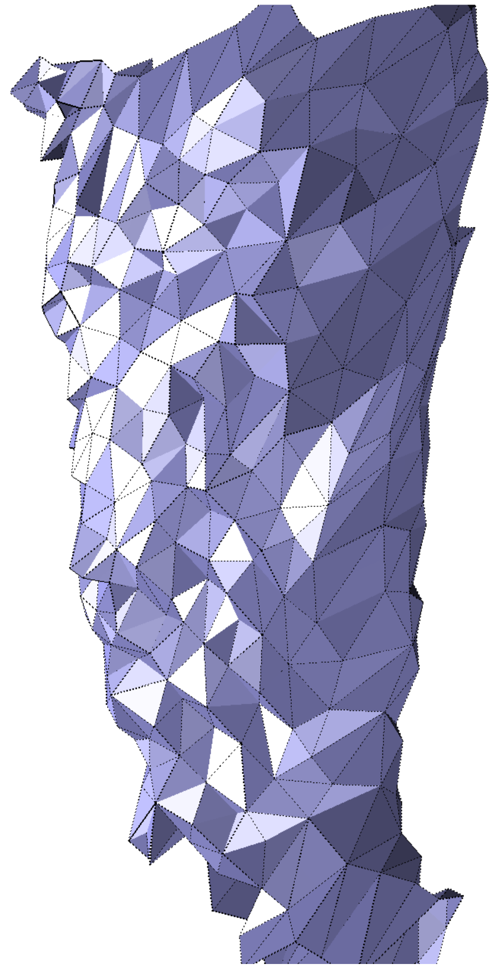

Figure 8: *Closer look at the neuron surface.*

4. Concluding Remarks

In this paper, we present the first non-trivial theoretical bound on the quality of the 3D solids reconstructed by any contour interpolation method. Our bound is still weak and does not seem to have any practical implications. (When α is small, the fact that T_3 has the minimum dihedral angle of $\frac{\alpha}{8}$ does not imply anything meaningful in practice.) However, it is a good starting point for further research in this direction. Several questions remain to be answered: (1) Is it possible to prove any topological result regarding the reconstructed solid using any contour interpolation method? (2) Is it possible to prove any practical bound on certain geometric parameters (e.g., minimum dihedral angle, minimum surface triangle area, etc) of the reconstructed solid using any contour interpolation method?

References

[BBX95] BAJAJ C., BERNARDINI F., XU G.: Automatic reconstruction of surface and scalar fields from 3D scans. In *Proc. 22nd ACM Conference on Computer Graphics and Interactive Techniques* (1995), pp. 109–118. 1

[BCER95] BERN M., CHEW P., EPPSTEIN D., RUPPERT J.: Dihedral bounds for mesh generation in high dimensions. In *Proc. 6th ACM-SIAM Sympos. Discrete Algorithms* (1995), pp. 189–196. 1

[BCL96] BAJAJ C., COYLE E., LIN K.: Arbitrary topology shape reconstruction from planar cross sections. *Graphical Models and Image Processing 58*, 6 (1996), 524–543. 1

[BGLSS03] BAREQUET G., GOODRICH M. T., LEVI-STEINER A., STEINER D.: Straight-skeleton based contour interpolation. In *Proc. 14th ACM-SIAM Sympos. Discrete Algorithms* (2003), pp. 119–127. 1

[Boi88] BOISSONNAT J.-D.: Shape reconstruction from pla-

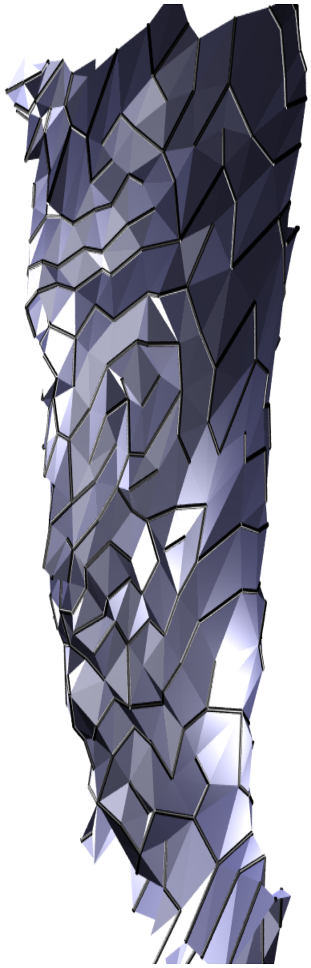

Figure 9: *The selected edges are depicted as bold.*

nar cross-sections. *Comput. Vision Graph. Image Process. 44*, 1 (Oct. 1988), 1–29. 1

[CD99] CHENG S. W., DEY T. K.: Improved construction of delaunay based contour surfaces. In *Proc. ACM Sympos. Solid Modeling and Applications* (1999), pp. 322–323. 1

[CDE*00] CHENG S.-W., DEY T. K., EDELSBRUNNER H., FACELLO M. A., TENG S.-H.: Sliver exudation. *J. ACM 47*, 5 (2000), 883–904. 1

[Che89] CHEW L. P.: Constrained Delaunay triangulations. *Algorithmica 4* (1989), 97–108. 2

[CMN98] CHAI J., MIYOSHI T., NAKAMAE E.: Contour interpolation and surface reconstruction of smooth terrain models. In *Proceedings of the conference on Visualization '98* (1998), IEEE Computer Society Press, pp. 27–33. 2

[FKU77] FUCHS H., KEDEM Z. M., USELTON S. P.: Optimal surface reconstruction from planar contours. *Commun. ACM 20* (1977), 693–702. 1

[Gei93] GEIGER B.: *Three-dimensional modeling of human organs and its application to diagnosis and surgical planning*. Report 2105, INRIA Sophia-Antipolis, Valbonne, France, 1993. 1

[GOS96] GITLIN C., O'ROURKE J., SUBRAMANIAN V.: On reconstructing polyhedra from parallel slices. *Internat. J. Comput. Geom. Appl. 6*, 1 (1996), 103–122. 1

[HSS03] HORMANN K., SPINELLO S., SCHRÖDER P.: C^1-continuous terrain reconstruction from sparse contours. In *Proceedings of Vision, Modeling, and Visualization 2003* (München, Germany, Nov. 2003), Ertl T., Girod B., Greiner G., Niemann H., Seidel H.-P., Steinbach E.,, Westermann R., (Eds.), infix, pp. 289–297. 2

[Kep75] KEPPEL E.: Approximating complex surfaces by triangulation of contour lines. *IBM J. Res. Develop. 19* (1975), 2–11. 1

[KSS00] KLEIN R., SCHILLING A. G., STRASSER W.: Reconstruction and simplification of surfaces from contours. *Graphical Models 62*, 6 (2000), 429–443. 2

[MV00] MITCHELL S. A., VAVASIS S. A.: Quality mesh generation in higher dimensions. *SIAM J. Comput. 29* (2000), 1334–1370. 1

[OPC96] OLIVA J. M., PERRIN M., COQUILLART S.: 3D reconstruction of complex polyhedral shapes from contours using a simplified generalized Voronoi diagram. *Comput. Graph. Forum 15*, 3 (1996), 397–408. 1

[SSBT01] SURAZHSKY T., SURAZHSKY V., BAREQUET G., TAL A.: Blending polygonal shapes with different topologies. *Computers & Graphics 25*, 1 (2001), 29–39. 2