

## An Experimental Study of Sliver Exudation

H. Edelsbrunner<sup>1</sup> and Damrong Guoy<sup>2</sup>

<sup>1</sup>Department of Computer Science, Duke University, Durham, NC, USA and Raindrop Geomagic, Research Triangle Park, NC, USA; <sup>2</sup>Center for Simulation of Advanced Rockets, Computational Science and Engineering Program, University of Illinois at Urbana-Champaign, IL, USA

**Abstract.** *We present results on a two-step improvement of mesh quality in three-dimensional Delaunay triangulations. The first step refines the triangulation by inserting sinks and eliminates tetrahedra with large circumradius over shortest edge length ratio. The second step assigns weights to the vertices to eliminate slivers. Our experimental findings provide evidence for the practical effectiveness of sliver exudation.*

**Keywords.** Dynamic triangulation; Mesh generation; Mesh quality; Slivers; Tetrahedra; Weighted Delaunay triangulations

### 1. Introduction

This paper is generally about improving the mesh quality of three-dimensional Delaunay triangulations and specifically about the practical effectiveness of sliver exudation as a method to eliminate flat tetrahedra. We implement published algorithms and study them experimentally, focusing on mesh quality.

**Meshing.** A *mesh* of a three-dimensional domain is a decomposition into simple pieces, called *elements*. We consider Delaunay triangulations, which decompose the domain into tetrahedral elements. One of the distinguishing properties of Delaunay versus other triangulations is that they are determined by the set of vertices sampled from the

domain. Another is that they have fast and reliable algorithms that make the construction of large and complicated meshes possible. There is an extensive literature studying three-dimensional Delaunay triangulations; see for example the recent text by Edelsbrunner [4].

Since the Delaunay triangulation is unique for a given set of vertices, the problem of constructing a good quality mesh reduces to choosing vertices for which the Delaunay tetrahedra have good quality. In a nutshell this means that all angles are in the intermediate range, avoiding small and large values. The Delaunay refinement algorithm pioneered by Ruppert [10] in two dimensions makes use of this reduction by adding vertices incrementally. Shewchuk [11] extends Ruppert's algorithm to three dimensions and reports good success by generally adding vertices at circumcenters of poor quality tetrahedra. The authors of this paper limit the choice of new vertices to sinks, which are special circumcenters [5]. The thus modified refinement method is used as the first step of the mesh improvement algorithm studied in this paper.

**Slivers.** The main shortcoming of the Delaunay refinement algorithm in three dimensions is its inability to remove slivers, which are rather flat tetrahedra with relatively small circumspheres. The persistent presence of slivers in large Delaunay triangulations has been observed experimentally at least as early as 1985 [1], but effective methods dealing with them have been found only recently [2,3,6,9]. This paper implements the sliver exudation algorithm of Cheng *et al.* and studies its effectiveness in practice. The question in focus is how large a minimum dihedral angle this method can achieve. The positive lower bound proved in Cheng *et al.* [2] is conservative and exceedingly small, and this paper confirms our intuition that the lower bound

---

Research by both authors is partially supported by NSF under grants CCR-97-12088 and DMS 98-73945. Research of the first author is also supported by the NSF under grants EIA-9972879 and CCR-00-86013 and by ARO under grant DAAG55-98-1-0177.

*Correspondence and offprint requests to:* H. Edelsbrunner, Department of Computer Science, Duke University, Box 90129, Durham, NC, USA

that can be achieved in practice is reasonably large, which makes the sliver exudation algorithm a viable method in practice.

Our experimental results are, however, inconclusive for tetrahedra near the mesh boundary. The reason is a fundamental weakness of the sliver exudation method. Boundary treatment methods such as the one described by Li and Teng [9] will have to be added in the future to get software that guarantees good mesh quality throughout the domain.

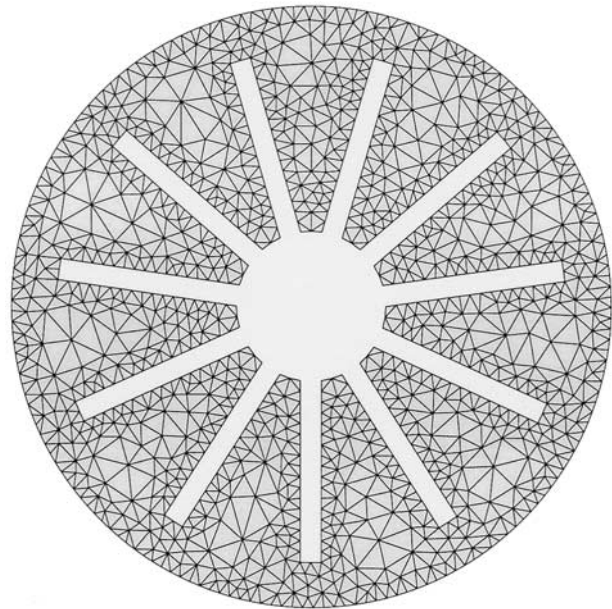
**Outline.** Section 2 reviews background material on Delaunay triangulations and mesh quality. Section 3 describes weighted Delaunay triangulations and the sliver exudation algorithm. Section 4 presents our experimental results for five three-dimensional data sets. Section 5 concludes the paper.

## 2. Delaunay Refinement

In this section, we review the material necessary to understand the first step of our mesh improvement, the Delaunay refinement through sink insertion. We refer to the experimental study in Edelsbrunner and Guoy [5] for details.

**Delaunay triangulations.** Given a finite set  $S$  of points in  $R^3$ , the *Delaunay triangulation* consists of a subset of the tetrahedra spanned by the points. We call the circumsphere of such a tetrahedron *empty* if all points other than the vertices of that tetrahedron lie outside the sphere. The Delaunay triangulation consists exactly of all tetrahedra with empty circumspheres. In the general case, in which no five points lie on a common sphere, the Delaunay triangulation is unambiguous and has the face-to-face property. In degenerate cases, we construct a triangulation that is the Delaunay triangulation of an infinitesimal perturbation of the point set [7].

The above definition does not mention any kind of boundary for the domain we mesh. We deal with this problem by constructing conforming Delaunay triangulations that contain a specified two-dimensional triangulation of the domain boundary as a subcomplex. This is achieved by making sure that no vertices lie inside the equator spheres of the boundary triangles. We then remove the tetrahedra outside that boundary and thus obtain a triangulation of the domain. The property of having empty equator spheres of boundary triangles is maintained throughout the mesh improvement process. Figure 1 illustrates the idea with an example in two dimen-



**Fig. 1.** Delaunay triangulation in two dimensions with triangles outside the two boundary curves removed.

sions, where the Delaunay triangulation consists of all triangles with empty circumcircles, and the boundary edges are protected by empty diameter circles.

**Mesh quality.** We use the classification of tetrahedra introduced in [2]. It distinguishes poor quality tetrahedra with small Hausdorff distance to a line segment from those with small Hausdorff distance to a planar figure. Among the latter, we distinguish tetrahedra with small Hausdorff distance to a triangle from *slivers*, which have small Hausdorff distance to a quadrangle. A refinement of this classification into nine smaller classes is shown in Fig. 2. We keep in mind that the classification is fuzzy and depends for example on what exactly we mean by *small* Hausdorff distance.

We use two measures to quantify what we mean by the quality of a tetrahedron. The first is the *ratio* of the circumradius over the shortest edge length,  $r/\ell$ . The regular tetrahedron has the smallest possible ratio of  $r/\ell = \sqrt{6}/4 = 0.612$ . ... The first eight types of tetrahedra in Fig. 2 have large ratio, but slivers may have ratios as small as  $\sqrt{2}/2 = 0.707$  ... or slightly smaller.

The second measure is the minimum dihedral angle,  $\zeta$ . The regular tetrahedron maximizes the minimum dihedral angle at  $\zeta = \arccos \frac{1}{3} = 70.528\dots^\circ$ . Spires, spears, and spindles may have reasonably large value of  $\zeta$ , but the remaining six types necessarily have small dihedral angles.

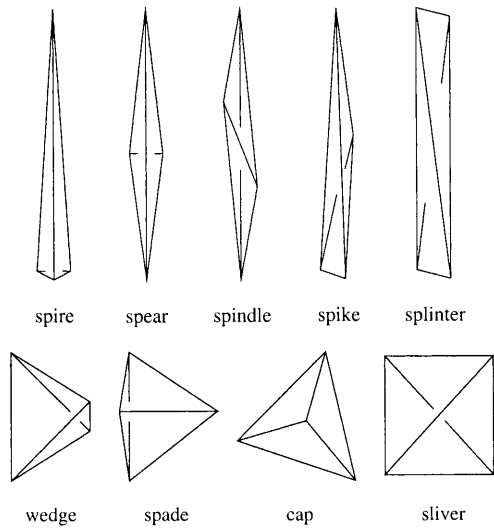


Fig. 2. A classification of poor quality tetrahedra into nine classes.

Figure 3 illustrates the regions of the various types of tetrahedra in the ratio-angle plane. The region near the upper left corner contains what we call good quality tetrahedra. Tetrahedra that are not in this region are identified by large ratio, small angle, or both.

**Sink insertion.** The basic idea of the Delaunay refinement algorithm is to identify a tetrahedron with large ratio and add its circumcenter to the vertex set. The empty sphere criterion implies that this tetrahedron is removed as part of the vertex insertion. As suggested in Edelsbrunner and Guoy [5], we modify the general strategy slightly and limit the added vertices to *sinks*, which are circumcenters that are contained inside their own Delaunay tetrahedra.

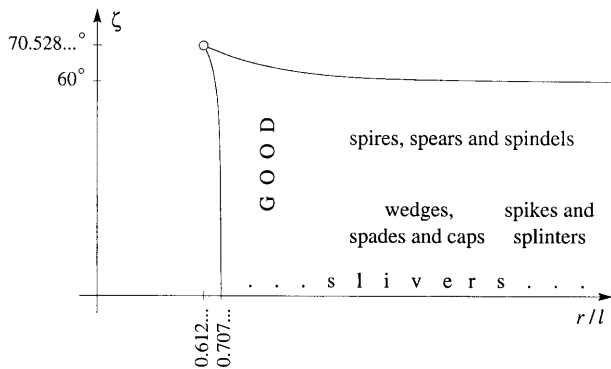


Fig. 3. Each tetrahedron is plotted as a point in the ratio-angle plane. In contrast to other low quality tetrahedra, slivers reach the left boundary of the region.

Figure 4 illustrates the definition by showing the sinks in a two-dimensional Delaunay triangulation. The effect of Delaunay refinement by sink insertion can be seen by comparing Fig. 4 before with Fig. 1 after refinement through iterative sink insertion. Near the boundary, the refinement strategy has to be modified to preserve the protecting equator spheres of boundary triangles. There are different options and we decide to keep the boundary untouched by prohibiting new vertices inside the equator spheres.

### 3. Sliver Exudation

The Delaunay refinement algorithm removes all poor quality tetrahedra, except slivers. The second step of our mesh improvement method removes slivers by sliver exudation as described in Cheng *et al.* [2]. This section provides the necessary background, most importantly the generalization of Delaunay to weighted Delaunay triangulations.

**Weighted Delaunay triangulations.** The generalization replaces the Euclidean distance by more general distance functions. We assign to each vertex  $u$  a weight  $U^2 \in \mathbb{R}$  and define the *weighted square distance* between  $(u, U)$  and  $(z, Z)$  as  $\|u - z\|^2 - (U^2 + Z^2)$ . For zero weights, the weighted square distance is the square of the Euclidean distance. Note that the weighted square distance does not change if we

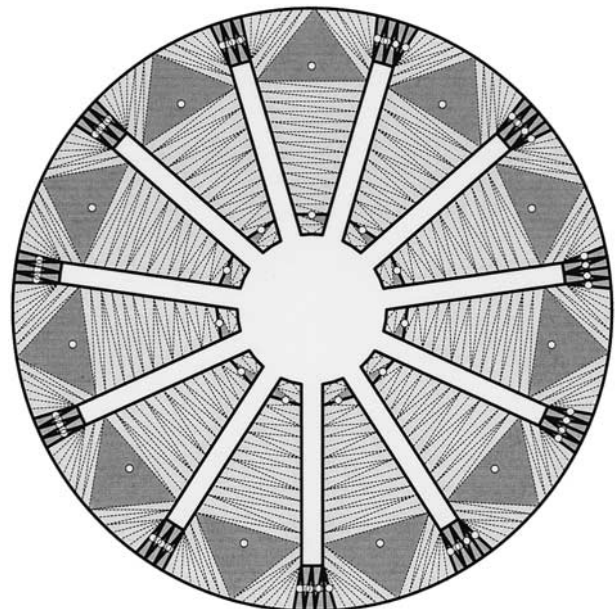


Fig. 4. In two dimensions, the (white) sinks are circumcenters of (dark) non-obtuse Delaunay triangles.

increase the weight of one vertex and decrease the weight of the other by the same amount.

A crucial idea is the interpretation of  $(u, U)$  as the sphere with center  $u$  and radius  $U$ . Two spheres are *orthogonal* if their weighted square distance is zero. Observe that two orthogonal spheres intersect in a circle and have perpendicular tangent planes along this circle. If one of the spheres is only a point then it lies on the other sphere. It follows that the circumsphere of a tetrahedron is the unique sphere orthogonal to the four vertices. If we assign weights to the vertices then we still have a unique orthogonal sphere, known as the *orthosphere* of the tetrahedron. Figure 5 illustrates this idea in two dimensions. When all points have zero weight, their orthosphere is their circumsphere. After some points gain weights, the orthosphere changes in the manner to preserve orthogonality.

Given a set of spheres, we call the orthosphere of four *empty* if all other spheres have positive weighted square distance. The *weighted Delaunay triangulation* of the set consists of all tetrahedra spanned by the centers that have empty orthospheres. For the special case of zero weights, this is the same as the unweighted Delaunay triangulation. Similar to the unweighted case, the weighted Delaunay triangulation is unambiguous if the spheres are in general position, which includes that no five spheres have a common orthosphere. Figure 6 shows two weighted Delaunay triangulations of eight points on a cube. Assigning different weights can cause different triangulations.

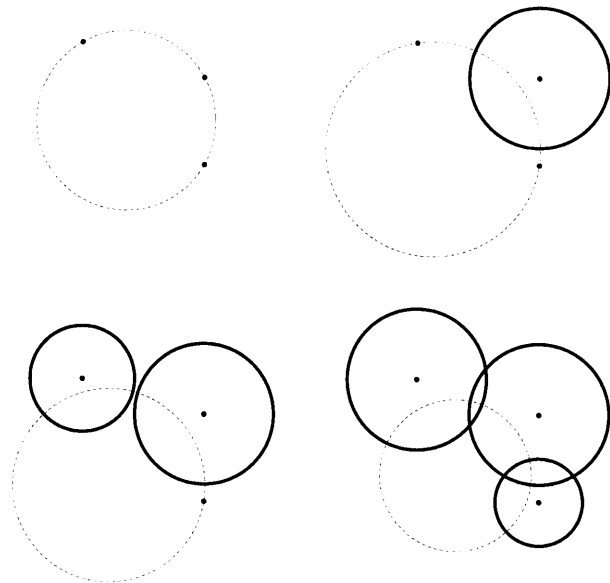


Fig. 5. Orthospheres are generalization of circumspheres.

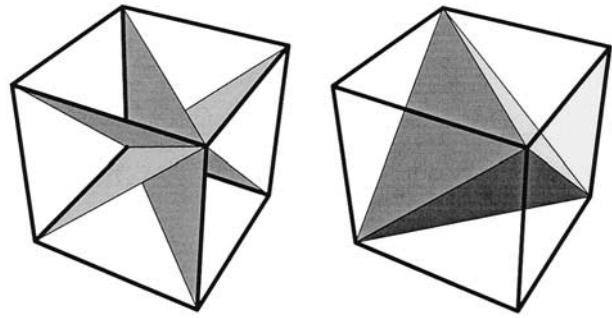


Fig. 6. Two weighted Delaunay triangulations of a cube.

**Weight pumping.** Consider a vertex  $u$  in a weighted Delaunay triangulation. Its *star* consists of all tetrahedra that contain  $u$  as a vertex. By construction, these tetrahedra all have empty orthospheres. If we continuously increase the weight of  $u$ , as shown in Fig. 7, these orthospheres are pushed away from  $u$  invading the space outside the original orthospheres.

At discrete moments in time, the weighted square distance between an invading orthosphere and a sphere of a vertex outside the star vanishes. We say the weight of  $u$  at that time is *critical*. To prevent the weighted square distance to become negative, we locally change the triangulation by a *flip*. A flip operation in 3D replaces two tetrahedra by three tetrahedra that occupy the same space, or vice versa. We refer to Edelsbrunner and Shah [8] for details on maintaining a weighted Delaunay triangulation by flipping. Figure 8 illustrates the two basic flips.

We may compute the critical weights of  $u$  in increasing order by breadth-first search using a priority queue. At any moment during the process, the *prestar* of  $u$  consists of all tetrahedra in the initial weighted Delaunay triangulation whose orthospheres have negative weighted square distances to  $u$ . Between critical weights, the new triangulation is

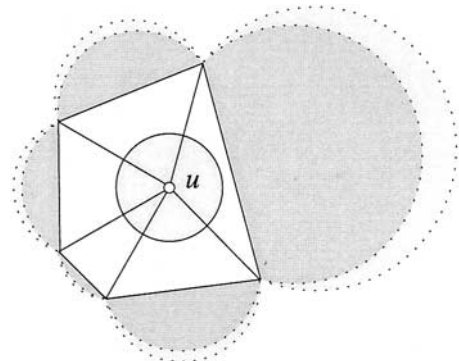
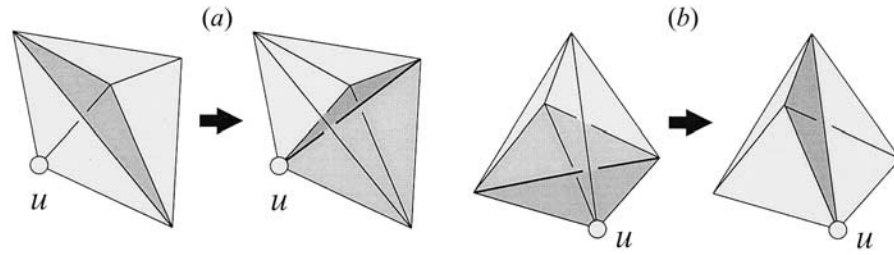


Fig. 7. The dotted orthocircles of the triangles in the star of  $u$  are pushed away from  $u$  as its weight increases.



**Fig. 8.** Two kinds of flips. (a) The two-to-three flip replaces the two tetrahedra sharing the link triangle of  $u$  by the three tetrahedra sharing a star edge of  $u$ . (b) The three-to-two flip replaces the three tetrahedra, one of which is a sliver, around the link edge of  $u$  by the two tetrahedra sharing a star triangle of  $u$ .

obtained by substituting the current star for the prestar of  $u$ . We call the above process *pumping*  $p$ . Figure 9 demonstrates this idea schematically.

Let  $d_u$  be the minimum Euclidean distance between  $u$  and any other vertex in the triangulation. The sliver exudation algorithm pumps  $p$  to a weight that maximizes the minimum dihedral angle  $\zeta$  of any tetrahedron in the star, but it does not expand the radius beyond  $0.45 \cdot d_u$ . Here 0.45 is an arbitrary positive constant less than one half. The reason for that restriction is that overlapping spheres may cause some vertices to be deleted from the weighted Delaunay triangulation.

With the mentioned stopping criterion, all spheres are disjoint and all points are vertices of the weighted Delaunay triangulation. The main result of Cheng *et al.* [2] is a proof that if we pump all vertices as described then all dihedral angles are larger than some constant  $\epsilon > 0$  that is independent of the set of input spheres. In the proof, that constant is positive but miserably small. As our experiments

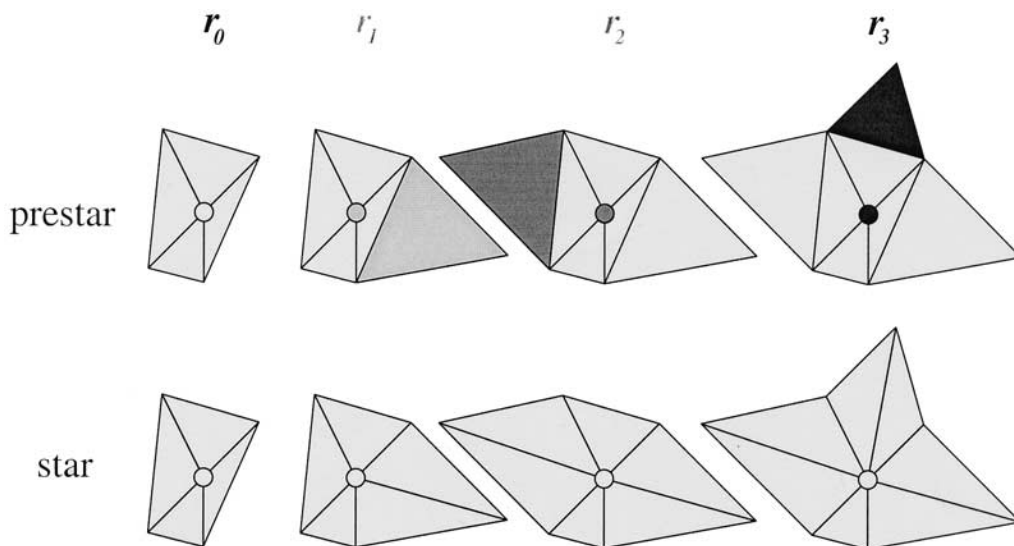
show, the constant that can be achieved in practice is much larger and possibly around  $5^\circ$ .

**Exudation algorithm.** The algorithm pumps every vertex in the triangulation. There is no restriction on the scheduling sequence, and pumping each vertex once is enough. Let  $S$  be the vertex set of the initial (unweighted) Delaunay triangulation.

```

void SLIVER EXUDATION ( $S$ )
foreach vertex  $u \in S$  do
     $U^2 = \text{PUMP}(u)$ ;
    substitute ( $u, U$ ) for  $u$  and star for prestar
endfor.
    
```

The optimal weight  $U^2$  for a vertex  $u$  is computed as explained above. Keep in mind that  $U^2$  is optimal only under fairly limiting conditions, namely that all other vertices have the fixed weight they happen to have at the moment and  $U$  is less than  $0.45 \cdot d_u$ . To formally describe the pumping process, we write



**Fig. 9.** Between critical weights, the new triangulation is obtained by substituting the current star for the prestar.

$U_0^2 = 0$  for the initial weight and  $U_1^2 < U_2^2 < \dots$  for the critical weights of  $u$ . We let  $\zeta_i$  be the minimum dihedral angle of all tetrahedra in the star of  $u$  at the weight  $U_i^2$ . The optimal weight  $U_j^2$  is the one that maximizes the angle  $\zeta_i$ .

```

float PUMP ( $u$ )
   $i = 0$ ;  $U_o^2 = 0$ ;  $j = 0$  compute  $\zeta_o$ ;
  loop
     $i = i + 1$ ;
    compute next critical weight  $U_i^2$ ;
    if  $U_i > 0.45 \cdot d_u$  then exit endif;
    expand the star of  $u$  and compute  $\zeta_i$ ;
    if  $\zeta_i > \zeta_j$  then  $j = i$  endif
  forever;
  return  $U_j^2$ 

```

We note that for efficiency reasons, the star in Function PUMP is computed incrementally. The pre-star is explicitly constructed and the Delaunay triangulation is updated only once per vertex in Function SLIVEREXUDATION. In order to maintain the data structure dynamically, we use the prestar to delete and the star to create records for tetrahedra.

The sliver exudation algorithm is modified for vertices near the domain boundary. Specifically, we limit the weight of every interior vertex so it cannot assume a negative weighted square distance to the equator sphere protecting any boundary triangle. Similarly, we limit the weight of a boundary vertex so it cannot assume a negative weighted square distance to the equator sphere protecting any non-incident boundary triangle. We check this condition in the prestar of the vertex when it is pumped.

Let  $n = \text{card } S$  be the number of vertices. The number of iterations in Function SLIVEREXUDATION is  $n$ . Assuming a constant upper bound on the ratios  $r/l$ , Cheng *et al.* [2] prove that the star of every vertex has constant size. Function PUMP thus takes only constant time per vertex, which adds up to a total time of  $O(n)$  for sliver exudation.

## 4. Experimental Results

This section presents experimental results for five three-dimensional data sets. All experiments are done on a Pentium II 450 MHz CPU with 128 MB of memory. For each data set, we evaluate the mesh quality of the Delaunay triangulation initially (I), after refinement (R), and after sliver exudation (X).

Each data set starts out with a boundary triangulation (B), so the initial Delaunay triangulation has no interior vertices. Our algorithm behaves differ-

ently in the interior and near the boundary. To differentiate, we call a tetrahedron *next* to the boundary if at least one of its vertices lies on the boundary, and *interior*, otherwise. To simplify the discussion, we fix a threshold and call a tetrahedron a *sliver* if its minimum dihedral angle is  $\zeta < 5^\circ$ .

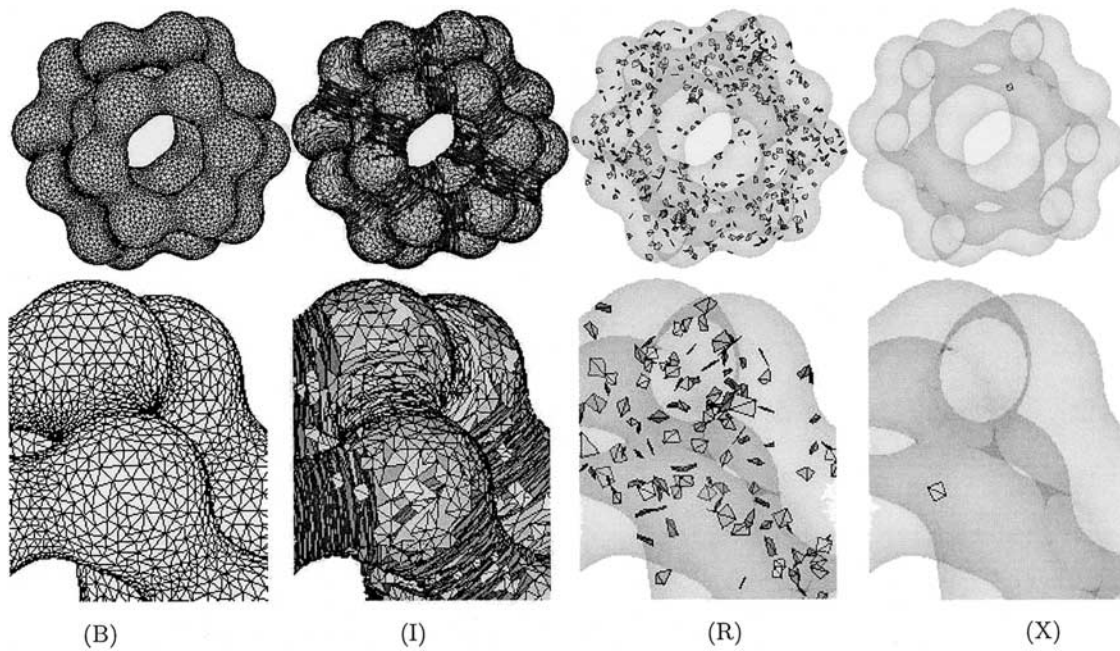
**A smooth surface of Permutahedron.** Our first example is a triangulated skin surface obtained from 24 spheres, which are connected by blending hyperboloids and inverse sphere patches. The sphere centers are the vertices of a convex polytope and represent the permutations of four objects, hence the name. The first row in Fig. 10 shows the entire mesh and the second an enlarged portion. The first column shows the boundary mesh while the other three columns show aspects of the volume mesh at different stages of the algorithm. The corresponding statistics is presented in Table 1.

The initial Delaunay triangulation consists of more than 76,000 tetrahedra, of which about 38% are slivers and are therefore visible in the second column of Fig. 10. In the sphere regions the slivers tend to be small and parallel to the boundary, while in hyperboloid regions they form stacks of large slivers roughly normal to the axis.

The refinement takes about 53 minutes and decreases the ratio below 1.0 except for 384 tetrahedra in the interior and more than 5,000 next to the boundary. The total number of vertices and tetrahedra goes up by more than a factor of three, which is reasonable because the initial mesh has no vertex in the interior. By construction, all new vertices are in the interior. At the same time, the refinement step eliminates the majority of the slivers, namely the ones with large circumspheres, but 516 slivers with small circumspheres remain.

Sliver exudation takes about 9 minutes and decreases the number of tetrahedra by 2.7% as it replaces flat tetrahedra. The ratio distribution worsens only slightly. The average minimum dihedral angle improves slightly, but most significantly, all dihedral angles move above the five degree threshold, except for one. That one sliver has  $\zeta > 3^\circ$  and lies next to the boundary. We guess that it survives the exudation process because of the limited freedom in assigning weights near the domain boundary.

**A surface with sharp corners of Wheel.** The second example is one twelfth of a wheel. It has mechanical shape with sharp corners and sharp edges in the boundary. Figure 11 shows the entire shape in the first row and an enlarged portion



**Fig. 10.** Permutahedron data. From left to right, surface mesh (B), initial Delaunay mesh (I), Delaunay mesh after refinement (R), and weighted Delaunay mesh after exudation (X). Volume meshes are displayed by showing transparent boundary together with opaque slivers. No other tetrahedra are shown.

**Table 1.** Permutahedron data. Evaluation of boundary surface mesh (B), initial Delaunay mesh (I), the mesh after refinement by sink insertion (R), and the mesh after sliver exudation (X). Distribution tables distinguish tetrahedra in the interior (int) from the ones next to the boundary (bd)

	#vert	#tri	#tet	run time	$r/\ell$			$\zeta$		
					min	avg	max	min	avg	max
B	14,287	28,622	–	–	0.58	0.72	2.02	–	–	–
I	14,287	–	76,329	–	0.71	4.65	15.74	0.002	9.68	58.73
R	43,569	–	230,119	53 min	0.62	0.83	2.14	0.282	45.29	69.90
X	43,569	–	223,936	9 min	0.62	0.83	2.14	3.376	46.49	69.90

	$r/\ell$ distribution					$\zeta$ distribution				
	0–1	1–2	2–3	3–10	$\geq 10$	0–5	5–10	10–20	20–40	40–71
I, int	0	0	0	0	0	0	0	0	0	0
I, bd	838	7,570	8,869	58,299	753	28,726	20,009	17,238	9,704	652
R, int	143,984	384	0	0	0	268	827	3,775	31,892	107,606
R, bd	80,372	5,374	5	0	0	248	768	2,230	20,795	61,710
X, int	139,415	887	0	0	0	0	5	934	29,896	109,467
X, bd	78,162	5,466	6	0	0	1	16	710	19,992	62,915

in the second row. The corresponding statistics is presented in Table 2.

The initial mesh contains more than 34,000 tetrahedra with 2.4% slivers, which can be seen in the second column of Fig. 11. The refinement step leaves a few tetrahedra with ratios that exceed the

threshold of 1.0. In about 16 minutes, it almost doubles the number of vertices and almost triples the number of tetrahedra. The size inflation is less dramatic for the *wheel* than for the *Permutahedron* data because the domain is fairly thin and requires only a small number of interior vertices.

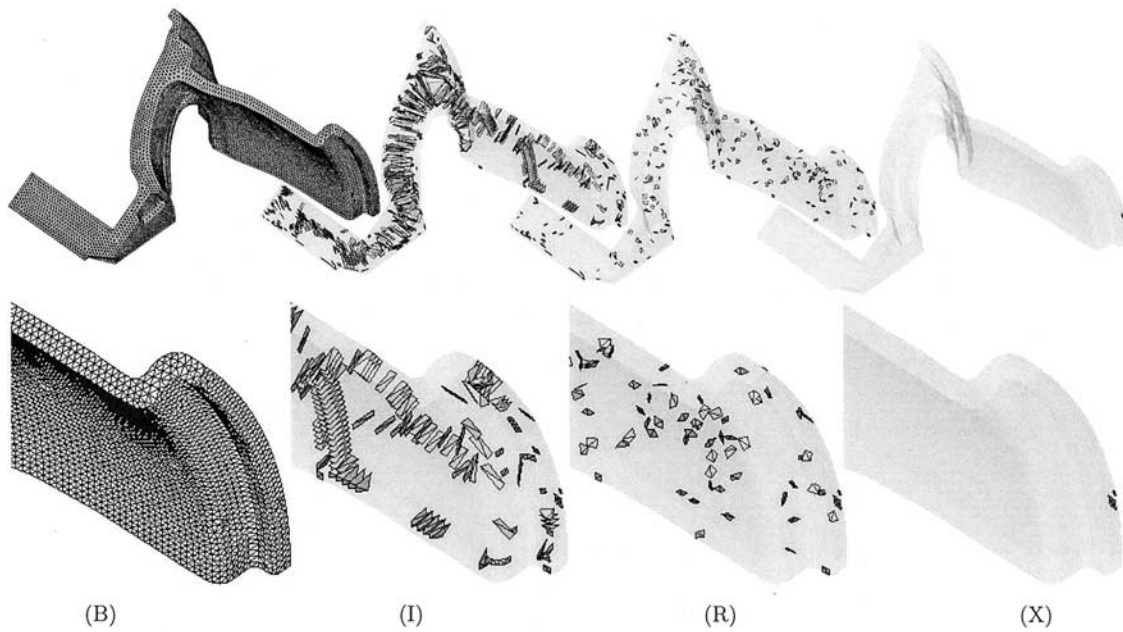


Fig. 11. Wheel data. The display conventions are the same as in Figure 10.

Table 2. Evaluation of the Wheel data

	#vert	#tri	#tet	run time	$r/\ell$			$\zeta$		
					min	avg	max	min	avg	max
B	11,525	23,046	–	–	0.58	0.67	1.78	–	–	–
I	11,525	–	34,892	–	0.65	1.94	6.57	0.000	34.16	63.92
R	20,688	–	93,186	16 min	0.61	0.82	2.34	0.000	46.02	70.28
X	20,688	–	90,969	2 min	0.61	0.82	2.34	0.336	47.10	70.28

	$r/\ell$ distribution					$\zeta$ distribution				
	0–1	1–2	2–3	3–10	$\geq 10$	0–5	5–10	10–20	20–40	40–71
I, int	0	0	0	0	0	0	0	0	0	0
I, bd	2,402	21,255	7,922	3,313	0	846	2,462	8,078	5,681	17,825
R, int	32,479	24	0	0	0	65	182	876	7,009	24,371
R, bd	59,709	841	133	0	0	188	327	1,308	11,984	46,876
X, int	31,384	109	0	0	0	0	0	227	6,540	24,726
X, bd	58,345	999	132	0	0	3	7	423	11,463	47,850

The refinement step also reduces the number of slivers but not by the impressive rate we have observed earlier. The exudation step takes only two minutes but leaves three slivers next to the boundary. As in the first example, it reduces the number of tetrahedra by 2.4%.

**Two boundary triangulations:** `ToothA` and `ToothB`. In the third example, we study the effect

of the boundary triangulation on the mesh improvement algorithm. The domain is a human tooth for which we compute the mesh starting with two different boundary triangulations. The two models are shown in Fig. 12, and the statistics is presented in Table 3.

The boundary triangulation is better for `ToothB` than for `ToothA`. That difference has apparently no influence on the running time of the algorithm, but



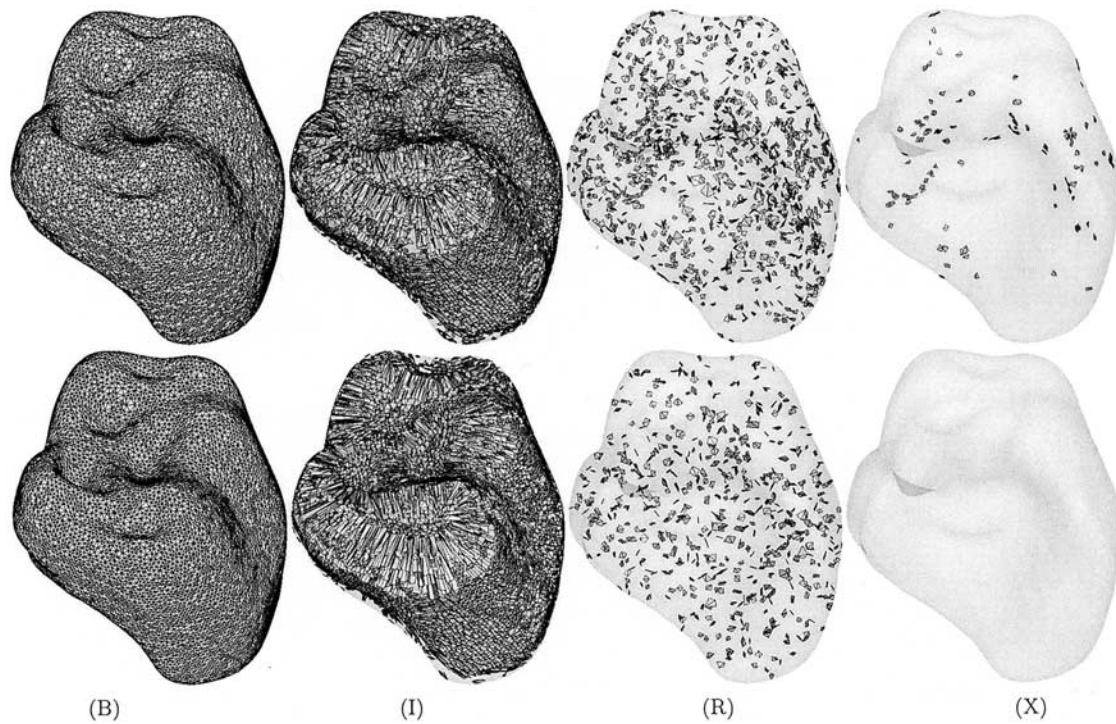


Fig. 12. The ToothA model in the upper and the ToothB model in the lower row.

Table 3. Evaluation of the ToothA data in the upper and of the ToothB data in the lower table

	#vert	#tri	#tet	run time	$r/\ell$			$\zeta$		
					min	avg	max	min	avg	max
B	13,453	26,092	–	–	0.58	0.82	3.40	–	–	–
I	13,453	–	47,286	–	0.74	7.88	37.02	0.007	20.19	60.12
R	52,325	–	291,122	1 h	0.61	0.84	3.60	0.047	44.40	70.48
X	52,325	–	282,633	15 min	0.61	0.84	3.60	0.170	45.69	70.48

	#vert	#tri	#tet	run time	$r/\ell$			$\zeta$		
					min	avg	max	min	avg	max
B	13,453	26,092	–	–	0.58	0.73	1.55	–	–	–
I	13,453	–	44,438	–	0.79	7.84	27.83	0.021	22.97	59.99
R	51,274	–	281,978	1 h	0.62	0.82	1.89	0.159	45.49	69.99
X	51,274	–	274,332	15 min	0.62	0.82	1.89	8.924	46.71	69.99

it has a significant influence on the mesh quality refinement and exudation achieve.

The difference in mesh quality is most striking after the exudation step: we observe 142 slivers with  $\zeta = 0.17^\circ$  in ToothA compared to no sliver and  $\zeta = 8.924^\circ$  in ToothB. It is telling that all 142 slivers lie next to the boundary, which is strong evidence

that our insistence on maintaining the boundary is the main reason for the slivers in the final mesh.

**More examples:** Head and Hog. We present experimental results for two additional data sets. The results are similar to what we have seen above, so we can be brief.

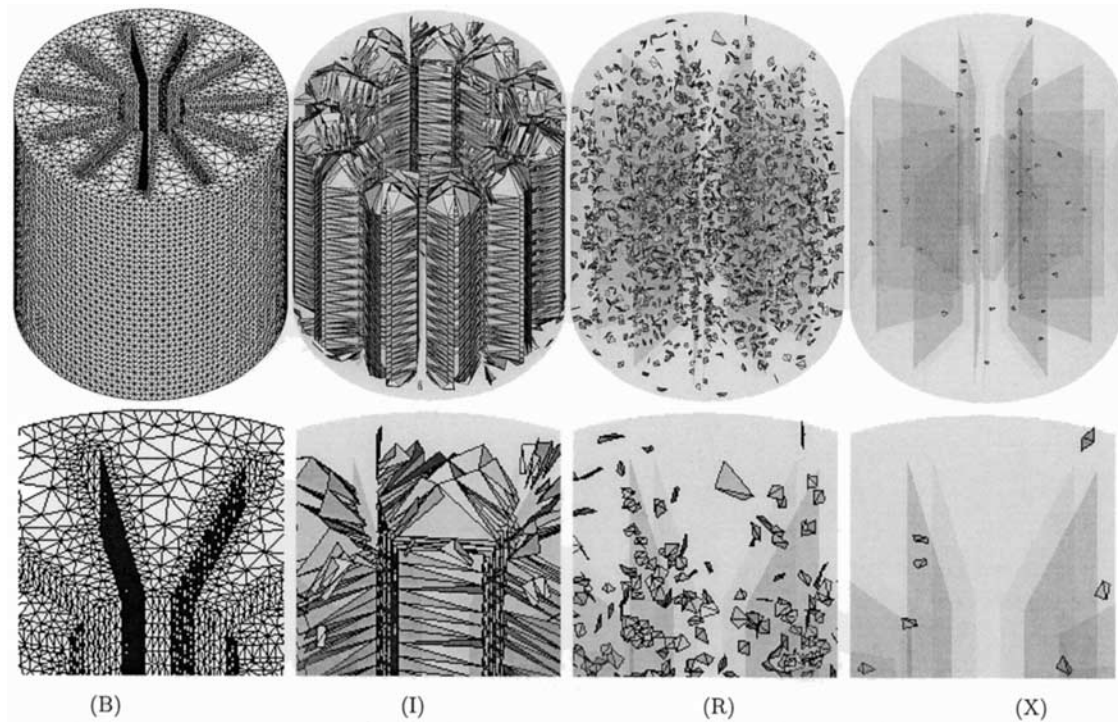


Fig. 13. The Head data models the solid propellant inside the rocket booster of Space Shuttle.

The Head data is displayed in Fig. 13 and the statistics is provided in Table 4. Both the input boundary triangulation and the output weighted Delaunay triangulation are the largest of all our examples, which explains the rather long running time.

The Hog data is displayed in Fig. 14 and the statistics is provided in Table 5. The large volume of the animal requires a large number of interior vertices, and we observe ratios of final over initial size that exceed the ratios in the other data sets.

Table 4. Evaluation of the Head data

	#vert	#tri	#tet	run time	$r/\ell$			$\zeta$		
					min	avg	max	min	avg	max
B	33,970	67,940	–	–	0.58	1.21	1.60	–	–	–
I	33,970	–	100,452	–	0.67	4.10	8.67	0.170	20.54	61.21
R	110,919	–	610,463	5 h	0.62	0.93	2.39	0.034	41.33	69.88
X	110,919	–	593,098	1.5 h	0.62	0.93	2.39	3.014	42.48	69.88

	$r/\ell$ distribution					$\zeta$ distribution				
	0–1	1–2	2–3	3–10	$\geq 10$	0–5	5–10	10–20	20–40	40–71
I, int	0	0	0	0	0	0	0	0	0	0
I, bd	482	20,880	19,382	59,708	0	10,119	13,378	20,230	50,535	6,190
R, int	370,547	8,008	7	0	0	750	2,450	11,243	89,281	274,838
R, bd	100,766	131,118	17	0	0	1,500	4,345	21,055	118,286	86,715
X, int	358,948	9,300	5	0	0	1	81	4,095	84,396	279,680
X, bd	98,389	126,434	22	0	0	38	1,182	16,519	119,244	87,862

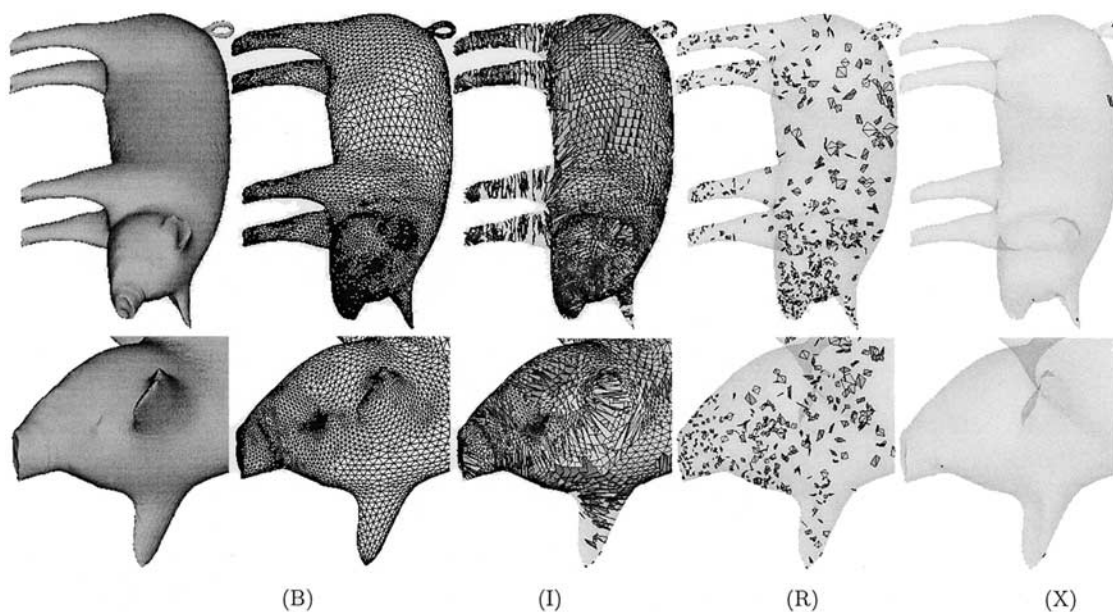


Fig. 14. The first two pictures show the surface of the Hog data smoothly rendered and triangulated.

Table 5. Evaluation of the Hog data

	#vert	#tri	#tet	run time	$r/\ell$			$\zeta$		
					min	avg	max	min	avg	max
B	13,474	26,948	–	–	0.58	0.76	2.89	–	–	–
I	13,474	–	46,491	–	0.71	7.84	297.15	0.000	18.75	59.88
R	56,982	–	317,565	2 h	0.62	0.84	3.50	0.322	44.98	70.20
X	56,982	–	308,801	20 min	0.62	0.84	3.50	2.529	46.22	70.20

	$r/\ell$ distribution					$\zeta$ distribution				
	0–1	1–2	2–3	3–10	$\geq 10$	0–5	5–10	10–20	20–40	40–71
I, int	0	0	0	0	0	0	0	0	0	0
I, bd	433	6,030	5,860	26,455	7,713	10,076	9,391	8,928	11,888	6,208
R, int	231,103	1,783	4	0	0	406	1,347	6,225	51,811	173,101
R, bd	71,376	13,247	50	2	0	327	740	2,867	24,392	56,349
X, int	223,660	2,614	4	0	0	0	10	1,578	48,247	176,443
X, bd	69,493	12,980	48	2	0	6	73	1,226	23,754	57,464

### 5. Discussion

The computational experiments presented in this paper provide evidence for the practical viability of sliver exudation as a method to remove slivers from three-dimensional Delaunay triangulations. Generally, the method succeeds in increasing all dihedral angles above 5°.

Our results are not as crisp as one would hope, and the main and perhaps only reason for the short-

coming is the lack of an effective method for improving the boundary mesh. This is not a weakness of our experimental set-up but rather a fundamental limitation of the sliver exudation algorithm as described in Cheng *et al.* [2]. Our positive results in the interior warrant additional efforts to rethink the way we deal with domain boundaries. Ideally, we would like to integrate the improvement of the boundary triangulation into the mesh improvement algorithm.

We note that the measure of ‘sliverness’ used in this paper is different from that in Cheng *et al.* [2]. We use the minimum dihedral angle  $\zeta$ , while the original exudation paper uses the ratio  $v/\ell^3$  of volume over the cube of the shortest edge length. Both measures approach zero as the tetrahedron gets flat, but they are quite different at the other extreme. The angle  $\zeta$  assumes its maximum  $70.528^\circ$  for the regular tetrahedron while the ratio  $v/\ell^3$  goes to infinity for skinny tetrahedra with short edges, like spires, spears, and splinters. The biggest advantage of the minimum dihedral angle  $\zeta$  is that it is intuitive and makes our statistical results easier to comprehend.

## Acknowledgement

The second author thanks Timothy J. Baker for the Hog data, Raindrop Geomagic for the Tooth data, and Holun Cheng for his skin software that generates surface mesh of Permutahedron.

## References

1. Cavendish, J.C., Field, D.A., Frey, W.H. (1985) An approach to automatic three-dimensional finite element mesh generation. *Internat. J. Numer. Methods Engrg.* 21, 329–347
2. Cheng, S.-W., Dey, T.K., Edelsbrunner, H., Facello, M. A., Teng, S.-H. (2000) Sliver exudation. *J. ACM*, 47, 883–904
3. Chew, L.P. (1997) Guaranteed-quality Delaunay meshing in 3D. *Proceedings 13th Annual Symposium on Computational Geometry*, 391–393.
4. Edelsbrunner, H. (2001) *Geometry and Topology for Mesh Generation*. Cambridge University Press
5. Edelsbrunner, H., Guoy, D. (2002) Sink insertion for mesh improvement. *Internat. J. Found. Comput. Sci.* 13, 223–242
6. Edelsbrunner, H., Li, X.-Y., Miller, G.L., Stathopoulos, A., Talmor, D., Teng, S.-H., Üngör, A., Walkington, N. (2000) Smoothing and cleaning up slivers. *Proceedings 32nd Annual ACM Symposium on the Theory of Computing*, 273–277
7. Edelsbrunner, H., Mücke, E.P. (1990) Simulation of Simplicity: a technique to cope with degenerate cases in geometric algorithms. *ACM Trans. Graphics*, 9, 66–104
8. Edelsbrunner, H., Shah, N.R. (1996) Incremental topological flipping works for regular triangulations. *Algorithmica*, 15, 223–241
9. Li, X.-Y., Teng, S.-H. (2001) Generating well-shaped Delaunay meshes in 3D. *Proceedings 12th Annual ACM-SIAM Symposium on Discrete Algorithms*, 28–37
10. Ruppert, J. (1995) A Delaunay refinement algorithm for quality 2-dimensional mesh generation. *J. Algorithms*, 18, 548–585
11. Shewchuk, J. (1998) Tetrahedral mesh generation by Delaunay refinement. *Proceedings 14th Annual Symposium on Computational Geometry*, 86–95