ELSEVIER

Information Processing Letters 54 (1995) 275–280

Information Processing Letters

Why is the 3D Delaunay triangulation difficult to construct?

Kokichi Sugihara [a,*], Hiroshi Inagaki [b]

[a] *Department of Mathematical Engineering and Information Physics, Faculty of Engineering, University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo 113, Japan*

[b] *Department of Information and Computer Engineering, Toyota College of Technology, 2-1 Eisei-cho, Toyota, Aichi 471, Japan*

Communicated by M.J. Atallah; received 21 October 1994; revised 8 February 1995

Keywords: Delaunay triangulation; Degeneracy; Numerical robustness; Topological consistency

1. Introduction

The Delaunay triangulation is one of the most fundamental concepts in computational geometry, and has many applications in engineering such as finite element analysis, computer graphics and interpolation. From a theoretical point of view, the properties of the Delaunay triangulations have been well studied in both two and three dimensions, and many efficient algorithms exist [1,2,14,16].

From a practical point of view, however, we still have serious problems, because in actual computers arithmetic is carried out only in finite precision, and algorithms often fail due to inconsistency caused by numerical errors. Indeed, the same difficulty arises in many geometric algorithms [7].

Various approaches have been proposed to numerically robust geometric algorithms. They include the exact-arithmetic approach [11,18,19] (often together with the symbolic perturbation technique [4,5,23]), the tolerance-based approach [6,13], the axiomatic approach [10,17], and the combinatorial-abstraction approach [8,9,20,21]. Among them we have been studying the combinatorial-abstraction approach extensively, because this approach enables us to separate the inconsistency issue completely from the error analysis issue and consequently the design and analysis of algorithms is simpler than in other approaches.

Studying the Delaunay triangulation on the basis of this approach, we found a somewhat strange fact: in the two-dimensional space the construction of the Delaunay triangulation is almost equivalent to the construction of the Voronoi diagram, whereas in the three-dimensional space the Delaunay triangulation is much more difficult to construct than the Voronoi diagram. This is mainly due to the difference of the degeneracy in two and three dimensions.

This paper shows why the three-dimensional Delaunay triangulation is difficult to construct and presents a way to avoid this difficulty.

2. Delaunay triangulation

Let $P = \{p_1, p_2, \ldots, p_n\}$ be a finite set of distinct points in $\mathbb{R}^d$; in this paper we are interested in only $d = 2$ or 3. Let $\mathrm{CH}(P)$ denote the convex hull of P. If S is a maximal subset of P such that all the points in S are on a common sphere

* Corresponding author.

SSDI 0020-0190(95)00028-3

(circle for $d = 2$) and if this sphere contains no element of P in its interior, the convex hull $\mathrm{CH}(S)$ of S is called a *Delaunay polytope* (*Delaunay polyhedron* for $d = 3$ and *Delaunay polygon* for $d = 2$). The convex hull $\mathrm{CH}(P)$ is partitioned into the Delaunay polytopes and their boundaries. A Delaunay polytope is said to be *nondegenerate* if it has exactly $d + 1$ vertices, and *degenerate* otherwise. If all the Delaunay polytopes are nondegenerate, we say that P is *nondegenerate*.

First, suppose that P is nondegenerate. Then, all the Delaunay polytopes are d-simplices (i.e., tetrahedra for $d = 3$ and triangles for $d = 2$). The convex hull $\mathrm{CH}(P)$ is partitioned into d-simplices and their boundaries. This partition is called the *Delaunay triangulation* and is denoted by $\mathrm{Del}(P)$.

Next, suppose that P is degenerate. Then, some of the Delaunay polytopes are not simplices. We decompose these polytopes into d-simplices in an arbitrary manner, and thus obtain the partition of $\mathrm{CH}(P)$ into d-simplices and their boundaries. This partition is also called the Delaunay triangulation and is denoted by $\mathrm{Del}(P)$.

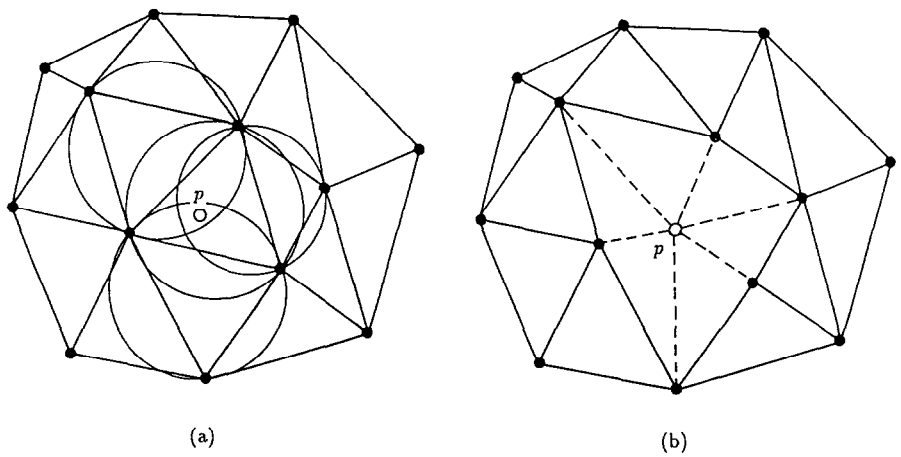
Note that $\mathrm{Del}(P)$ is unique if and only if P is nondegenerate. We call the d-simplices in $\mathrm{Del}(P)$ the *Delaunay simplices*, and the $(d-1)$-dimensional faces (i.e., edges for $d = 2$ and triangles for $d = 3$) of the Delaunay simplices the *Delaunay facets*. We say that two Delaunay simplices are *adjacent* to each other if their boundary share a common Delaunay facet.

3. Incremental construction

Let us fix finite set P of points in $\mathbb{R}^d$. A sphere (or a circle for $d = 2$) is called an *empty sphere* if it contains no element of P in the interior. A d-simplex with the vertices in P can be a Delaunay simplex if and only if the circumscribing sphere is empty. From this property, we can consider an incremental method for constructing the diagram.

In the incremental method, we start with the Delaunay triangulation for a few generators, and modify it by adding new generators one by one. As shown in Fig. 1(a), suppose that we have already constructed $\mathrm{Del}(P)$ and that we want to add new generator p ($\notin P$). Delaunay simplex T is said to be *inconsistent* with p if the sphere circumscribing T contains p in the interior, and *consistent* otherwise. For simplicity let us assume that $p \in \mathrm{CH}(P)$, as shown in Fig. 1(a). Then, we can construct $\mathrm{Del}(P \cup \{p\})$ in the following way [22].

First, we find all Delaunay simplices inconsistent with p, and delete them from $\mathrm{Del}(P)$. For the example in Fig. 1(a), we have four inconsistent Delaunay simplices, and removing them, we



(a) (b)

Fig. 1. Incremental construction: (a) Delaunay triangulation and a new generator; (b) retriangulation.

have a large connected region. Next, we decompose this region into simplices with the apex at p, as shown in (b); thus we get $\mathrm{Del}(P \cup \{p\})$.

This procedure can be represented naively in the following way.

Algorithm 1 (naive method)
1. Find all Delaunay simplices inconsistent with p, and name their union the region R.
2. Remove all the Delaunay simplices in R.
3. Decompose R into simplices with the apex at p and with the base facets on the boundary of R.

This algorithm is valid if computation is done precisely and if $P \cup \{p\}$ is non-degenerate, but is not necessarily otherwise. To see this, let us place the next assumption on the behavior of numerical errors.

Assumption 1. If p is not on the circumsphere of Delaunay simplex T, consistency of T with p is judged correctly. If p is exactly on the circumsphere of T, however, the judgement of the consistency is done at random.

This assumption simulates the situation where numerical error takes place but the amount of numerical error is small.

Suppose that we have constructed the two-dimensional Delaunay triangulation for seven generators $p_1, p_2, \ldots, p_7$, as shown in Fig. 2(a), where six of them are on common circle C, and also suppose that we want to add new generator p that is also on C. In Assumption 1, it can happen that the four Delaunay triangles represented by the shaded areas in Fig. 2(a) are judged inconsistent with p while the other triangles are judged consistent. Then, Algorithm 1 removes them and generates new simplices as shown by broken lines in Fig. 2(b). The result is inconsistent because simplices overlap each other. Thus, Algorithm 1 fails easily.

Fortunately, however, we usually do not use Algorithm 1 because it is time consuming. In Step 1 of Algorithm 1 we have to check all the Delaunay simplices. On the other hand, we know that all the inconsistent Delaunay simplices form a connected region containing p. Hence, in order to avoid time consuming check, we usually follow the next algorithm, which is different from Algorithm 1 in Steps 1 and 2 [12].

Algorithm 2 (textbook method)
1. Find the Delaunay simplex including p, and name it the region R.
2. Until R cannot be augmented any more, do: if there is Delaunay simplex T adjacent to R

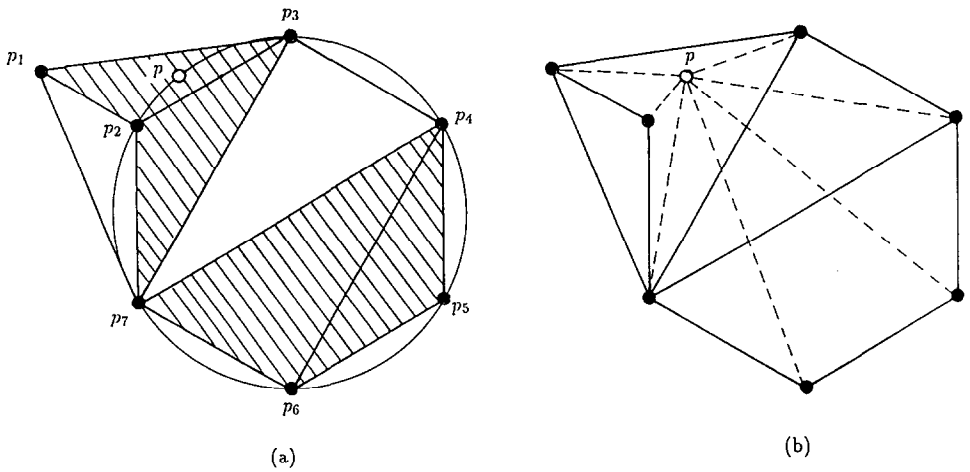


Fig. 2. Inconsistency in the two-dimensional Delaunay triangulation: (a) degenerate Delaunay triangulation and a new generator; (b) overlapping triangles obtained by Algorithm 1.

and inconsistent with p, remove the Delaunay facet separating T and R, and thus augment the region R.

3. Decompose R into simplices with the apex at p and with the base facets on the boundary of R.

In Step 2 of this algorithm we check only those Delaunay simplices that are adjacent to R. This, on the one hand, guarantees efficiency in computational time, and, on the other, guarantees consistency in the topological structure. Indeed, for the Delaunay triangulation in Fig. 2(a), the triangle $p_1p_2p_3$ is initially set as R, and then the triangle $p_2p_3p_7$ is merged to R, but the triangle $p_4p_6p_7$ or $p_4p_5p_6$ is not merged to R because they are not adjacent to R.

4. Difficulty in three dimensions

Algorithm 2 works well in two dimensions, but does not in three dimensions. The following is an example in which Algorithm 2 generates overlapping tetrahedra.

Suppose that eight generators p_n, p_s, p_1, $p_2,\ldots,p_6$ are on the boundary of an empty sphere; let p_n be at the north pole, p_s be at the south pole, and $p_1,\ldots,p_6$ be circularly placed on the equator in equal space. This situation is depicted in Fig. 3(a), in which the sphere is seen in the direction parallel to the line passing through the north and south poles, and hence the circle in this figure corresponds to the equator and the center corresponds to the north and south poles. Assume that other generators exist outside the sphere, though they are omitted in this figure.

Suppose that Del(P) contains tetrahedra $p_np_sp_ip_{i+1}$ $(i=1,2,\ldots,6)$ where p_7 is read as p_1. Now assume that new generator p is given on the shorter arc of the equator connecting p_1 and p_2. Then, p is exactly on the sphere that circumscribes the tetrahedra $p_np_sp_ip_{i+1}$ $(i=1,2,\ldots,6)$. Hence, in Assumption 1, whether or not the Delaunay tetrahedron $p_np_sp_ip_{i+1}$ is consistent with p is judged at random. Therefore, it may happen that the tetrahedron $p_np_sp_6p_1$ is judged consistent, and all the other five tetrahedra are judged inconsistent.

Then Step 2 of Algorithm 2 generates the region R as shown by the shaded area in Fig. 3(a), and Step 3 of Algorithm 2 generates overlapping tetrahedra as shown in Fig. 3(b); for example, tetrahedra $p_np_sp_1p_6$ and $pp_np_5p_6$

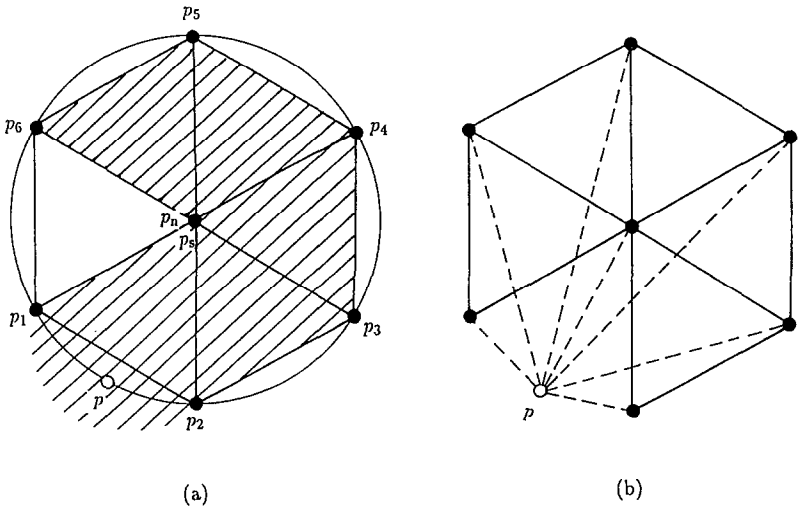


Fig. 3. Inconsistency in the three-dimensional Delaunay triangulation: (a) degenerate Delaunay triangulation and a new generator; (b) overlapping tetrahedra obtained by Algorithm 2.

overlap. Thus, Algorithm 2 is not stable in the three-dimensional space.

In order to avoid overlapping tetrahedra, the region R generated in Step 2 of Algorithm 2 should be star-shaped. Indeed, if P is not degenerate, the region R is star-shaped [2,3,14]. Hence, we should modify Algorithm 2 in such a way that the region R is guaranteed star-shaped even if P is degenerate.

Now that we understand how the difficulties arise, it is not hard to avoid it. To this end, we make the inconsistency test in the order that the Delaunay tetrahedra closer to p are checked earlier, and once we judge a tetrahedron consistent, we also judge that all the tetrahedra behind it are necessarily consistent. By this trick we can avoid overlapping tetrahedra. Thus we get the next algorithm, which differs from Algorithm 2 only in Step 2.

Algorithm 3 (stable method)

1. Find the Delaunay simplex including p, and name it the region R.
2. Until R cannot be augmented any more, do:
   if there is Delaunay simplex T such that
   (i) the line segments connecting p to the vertices of T are contained in $R \cup T$, and such that
   (ii) T is inconsistent with p,
   then, remove the Delaunay facet separating T and R, to augment the region R.
3. Decompose R into simplices with the apex at p and with the base facets on the boundary of R.

In Assumption 1, this algorithm never generates overlapping tetrahedra. This is because the condition (i) in Step 2 guarantees that the region R is always star-shaped with respect to p, i.e., all the facets on the boundary of R are visible from p when we consider the simplices not belonging to R are opaque.

5. Concluding remarks

In actual environment of computation, numerical errors and degeneracy are inevitable. Consequently, how to translate algorithms in textbooks into numerically robust computer programs is an important and nontrivial problem. This paper has concentrated on the incremental construction of the Delaunay triangulation, and filled the gap between the textbook algorithm and the practical computer program.

The difficulty we discussed in this paper usually does not appear when we construct the three-dimensional Voronoi diagram, i.e., the dual of the Delaunay diagram. Consider again the collection of overlapping tetrahedra shown in Fig. 3(b). If we take the dual of this structure, we obtain the collection of polyhedra, i.e., Voronoi polyhedra, whose facets are on the planes perpendicularly bisecting two points in P. Note that the points in P are on a common sphere and consequently all the bisecting planes go through the center of the sphere. Actually, the Delaunay edges penetrating other tetrahedra in Fig. 3(b) correspond to very tiny Voronoi facets at the center of the sphere, and the resulting collection of the Voronoi polyhedra form an almost correct partition of the space. In this sense, the construction of the Delaunay triangulation is more difficult than the construction of the Voronoi diagram in three dimensions.

The Delaunay triangulation for n points in three dimensions can have as much as $O(n^2)$ Delaunay tetrahedra [14], and hence the worst case time complexity is not smaller than $O(n^2)$. However, we empirically know that in many cases, including the case where the points are generated randomly in a cube, the number of Delaunay tetrahedra is of $O(n)$. In these cases, we also know from experience that if the bucketing technique is used for finding the Delaunay simplex that contains p, Algorithm 3 runs in $O(1)$ time on the average, and consequently the Delaunay triangulation can be constructed in $O(n)$ time on the average. See [15] for the details of employing the bucketing technique.

References

[1] F. Aurenhammer, Voronoi diagrams – A survey of a fundamental geometric data structure, *ACM Comput. Surveys* **23** (1991) 345–405.

[2] H. Edelsbrunner, *Algorithms in Combinatorial Geometry* (Springer, Berlin, 1987).

[3] H. Edelsbrunner, An acyclicity theorem for cell complexes in d dimensions, *Combinatorica* **10** (1990) 251–260.

[4] H. Edelsbrunner and E.P. Mücke, Simulation of simplicity – A technique to cope with degenerate cases in geometric algorithms, in: *Proc. 4th ACM Ann. Symp. on Computational Geometry*, Urbana-Champaign (1988) 118–133.

[5] I. Emiris and J. Canny, An efficient approach to removing geometric degeneracies, in: *Proc. 8th ACM Ann. Symp. on Computational Geometry*, Berlin (1992) 74–82.

[6] S. Fortune, Stable maintenance of point set triangulations in two dimensions, in: *Proc. 30th IEEE Ann. Symp. on Foundation of Computer Science*, Research Triangle Park (1989) 494–499.

[7] C. Hoffmann, The problems of accuracy and robustness in geometric computation, *Computer* **22**(3) (1989) 31–41.

[8] H. Inagaki and K. Sugihara, Numerically robust algorithm for constructing constrained Delaunay triangulation, in: *Proc. 6th Canadian Conf. on Computational Geometry*, Saskatoon (1994).

[9] H. Inagaki, K. Sugihara and N. Sugie, Numerically robust incremental algorithm for constructing three-dimensional Voronoi diagrams, in: *Proc. 4th Canadian Conf. on Computational Geometry*, St. John's (1992) 334–339.

[10] D.E. Knuth, *Axioms and Hulls*, Lecture Notes in Computer Science **606** (Springer, Berlin, 1992).

[11] M. Karasick, D. Lieber and L.R. Nackman, Efficient Delaunay triangulation using rational arithmetic, *ACM Trans. Graphics* **10** (1991) 71–91.

[12] D.-T. Lee and B.J. Schachter, Two algorithms for constructing the Delaunay triangulations, *Internat. J. Comput. Inform. Sci.* **9** (1980) 219–242.

[13] V. Milenkovic, Verifiable implementations of geometric algorithms using finite precision arithmetic, *Artificial Intelligence* **37** (1988) 377–401.

[14] A. Okabe, B. Boots and K. Sugihara, *Spatial Tessellations – Concepts and Applications of Voronoi Diagrams* (Wiley, London, 1992).

[15] T. Ohya, M. Iri and K. Murota, Improvements of the incremental method for the Voronoi diagram with computational comparison of various algorithms, *J. Oper. Res. Soc. Japan* **27** (1984) 306–336.

[16] F.P. Preparata and M.I. Shamos, *Computational Geometry – An Introduction* (Springer, New York, 1985).

[17] P. Schorn, Robust algorithms in a program library for geometric computation, Dissertation submitted to the Swiss Federal Institute of Technology (ETH) Zürich for the degree of Doctor of Technical Sciences, Diss ETH Nr. 9519 (1991).

[18] K. Sugihara, A simple method for avoiding numerical errors and degeneracy in Voronoi diagram construction, *IEICE Trans. Found. Electr., Commun. Comput. Sci.* **E75-A** (1992) 468–477.

[19] K. Sugihara and M. Iri, A solid modelling system free from topological inconsistency, *J. Inform. Process.* **12** (1989) 380–393.

[20] K. Sugihara and M. Iri, Construction of the Voronoi diagram for "one million" generators in single-precision arithmetic, *Proc. IEEE* **80** (1992) 1471–1484.

[21] K. Sugihara and M. Iri, A robust topology-oriented incremental algorithm for Voronoi diagrams, *Internat. J. Comput. Geom. Appl.* **4** (1994), in press.

[22] D.F. Watson, Computing the n-dimensional Delaunay tessellation with application to Voronoi polytopes, *Comput. J.* **24** (1981) 167–172.

[23] C.-K. Yap, A geometric consistency theorem for a symbolic perturbation scheme, in: *Proc. 4th ACM Ann. Symp. on Computational Geometry*, Urbana-Champaign (1988) 134–142.