NEW LEPP-ALGORITHMS FOR QUALITY POLYGON AND VOLUME TRIANGULATION: IMPLEMENTATION ISSUES AND PRACTICAL BEHAVIOR

María-Cecilia Rivara and Mauricio Palma
Department of Computer Science, University of Chile
Casilla 2777, Santiago, Chile
mcrivara@dcc.uchile.cl; mpalma@dcc.uchile.cl

ABSTRACT

In this paper we discuss the implementation and practical behavior of new LEPP-algorithms for the automatic construction of good-quality polygon and surface triangulations, which naturally extend to 3-dimensions. The algorithms essentially combine two basic techniques: a Backward Longest-Edge (interior) point insertion strategy and a Boundary Treatement technique, which together (in 2-dimensions) guarantee the construction of good-quality triangulations of smallest angles greater than or equal to 30º. The triangulations obtained have in practice an optimal number of points (analogously to the circumcenter point insertion algorithm of Ruppert). Different practical variants of the point insertion strategy are discussed and compared in this paper. We show that, in practice, they tend to produce the best mesh for a maximum number of allowable triangles. These techniques are in turn the basis to design automatic general algorithms to produce good-quality (adapted to the geometry) volume triangulations of general polyhedra including small details. Empirical evidence which shows the successful use of these ideas in 3-dimensions is also included.

1. INTRODUCTION

During the last 15 years the triangular mesh generation problem has evolved into an important and interdisciplinary research field. In this general context, the following three related problems (in 2 and 3-dimensions) should be consistently considered: (1) Triangular mesh refinement; (2) Triangular mesh improvement; (3) Automatic generation of good-quality surface and volume triangulation. In effect, many applications on numerical simulation, solid modeling and computer graphics require that complicated geometric objects be decomposed in simpler pieces for further processing. Furthermore, in the adaptive finite element setting, the availability of mesh refinement algorithms capable of modifying the mesh in the course of computations is a critical aspect of the entire numerical solution process. A difficult related problem (especially difficult in 3-dimensions) is the automatic construction of good-quality, adapted to the geometry, triangulations. In this case, a set of non-vertex (Steiner) points should be added to produce a quality triangulation to be later refined.

In this paper we discuss the implementation and practical behavior of new algorithms for the automatic construction of good-quality polygon and polyhedron-surface triangulations, (which naturally extend to 3-dimensions), based on the incremental improvement of a Delaunay triangulation. The algorithms make use of two basic techniques (Rivara, 1996a, 1996b): (1) a Backward Longest-Edge Point Insertion strategy, for the refinement/improvement of individual triangles (tetrahedra) over constrained Delaunay triangulations; (2) a simple Boundary Longest-Edge Treatment technique. The algorithms guarantee the construction of good-quality triangulations of smallest angles greater than or equal to 30º. We also include empirical experimentation which shows the successful extension of these ideas to 3-dimensions.

The Backward Point Insertion strategy takes advantage of the Longest-Edge Propagation Path (LEPP(t)) associated with each triangle (tetrahedron) t of the current mesh. This is defined in 2-dimensions as the finite ordered list of successive neighbor triangles having longest edge greater than or equal to the longest edge of the preceding triangle in the path. More specifically, in 2-dimensions, each new point to be added in the mesh is selected as the midpoint of the longest edge of the last (greatest) triangle of the LEPP(t), where t is the current target triangle to be refined/improved in the mesh. The essential ideas of the algorithm generalize previous work of Rivara and coauthors on pure longest-edge refinement algorithms as well as a longest-edge refinement algorithm for Delaunay triangulations (Rivara and Inostroza, 1997). These previous techniques guarantee the construction of refined triangulations that basically maintain the quality of the input coarse triangulations.

2. LONGEST-EDGE PROPAGATION PATH OF A TRIANGLE

In this section we shall consider general conforming unstructured triangulations (where the intersection of adjacent triangles is either a common vertex or a common edge).

Definition 1 For any triangle t_0 of any conforming triangulation T, the Longest-Edge Propagation Path of t_0 will be the ordered list of all the triangles t_0, t_1, t_2, ...t_{n-1}, t_n, such that t_i is the neighbor triangle of t_{i-1} by the longest edge of t_{i-1}, for i = 1,2,.., n. In addition we shall denote it as the LEPP(t_0).

Proposition 1 For any triangle t_0 of any conforming triangulation of any bounded 2-dimensional geometry Ο, the following properties hold: (a) for any t, the LEPP(t) is always finite; (b) The triangles t_0, t_1, ...t_{n-1} have strictly increasing longest edge (if n > 1); (c) For the triangle t_n of the Longest-Edge Propagation Path of any triangle t_0, it holds that either: (i) t_n has its longest edge along the boundary, and this is greater than the longest edge of t_{n-1}, or (ii) t_n and t_{n-1} share the same common longest edge.

Definition 2 Two adjacent triangles (t, t*) will be called a pair of terminal triangles if they share their respective (common) longest edge. In addition t will be a terminal boundary triangle if its longest edge lies along a boundary side.

Note that the Longest-Edge Propagation Path of any triangle t corresponds to an associated polygon, which in certain sense measures the local quality of the current point distribution induced by t. To illustrates these ideas, see Figure 1a, where the Longest-Edge Propagation Path of t_0 corresponds to the ordered list of triangles (t_0, t_1, t_2, t_3). Moreover the pair (t_2, t_3) is a pair of terminal triangles.

The definition 1 should be slightly modified to consider the case where the longest edge is not unique. In such a case, the longest edge that produces the shortest path should be selected.

3. A BASIC LONGEST-EDGE IMPROVEMENT TECHNIQUE FOR DELAUNAY TRIANGULATIONS.

The previous longest-edge algorithms guarantee the construction of refined triangulations that basically maintain the quality of the input coarse triangulation (Rivara 1984, 1992) (Rivara and Inostroza 1995, 1997). Furthermore, for dealing with the triangulation refinement problem they are of optimal (linear) time cost (Rivara and Venere, 1996), (Rivara and Inostroza, 1997). In the present case in exchange, this Delaunay improvement technique uses the Longest-Edge Propagation Path of the target triangles (to be either refined and/or improved in the mesh) in order to decide which is the best point to be inserted, in order to produce a good-quality distribution of points. This procedure is repeatedly used until the triangle t is destroyed.

Note that we have used the word improvement instead of bisection or refinement. This is to explicitize the fact that one step of the procedure does not necessarily produce a smaller triangle. More important however, is the fact that the procedure improves the triangle in the sense of Theorem 1.

```
Basic Backward-LE-Delaunay-Improvement (t, T)
While t remains without being modified do
    Find the Longest-Edge Propagation Path of t
    Perform a Delaunay insertion of the point p
       (midpoint of the longest edge of the last triangle
         in the LEPP(t))
```

For an illustration of the algorithm see Figure 1 where the triangulation (a) is this initial Delaunay triangulation with LEPP(t_0) = {t_0,t_1,t_2,t_3}, and the triangulation (b), (c) and (d) illustrate the complete sequence of point insertions needed to improve t_0. Note that in this example, the improvement (modification) of t_0 implies the automatic Delaunay insertion of 3 additional Steiner points. Each one of these points is the midpoint of the last triangle of the current LEPP(t_0). It should be pointed out here that each Delaunay point insertion essentially improves the local point distribution in the current LEPP(t_0), and in this sense this algorithm improves the triangulations obtained with the pure Backward Longest-Edge Refinement procedure.

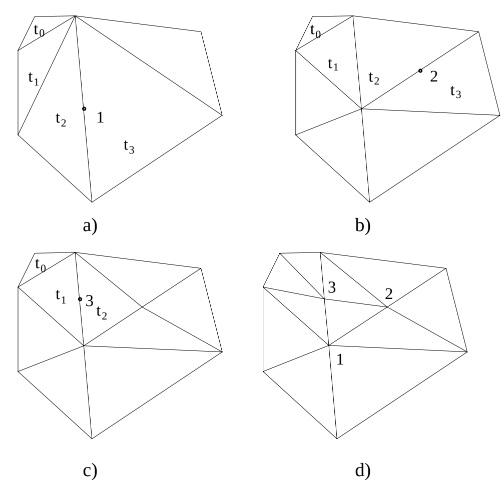


Figure 1. Backward Longest-Edge Delaunay improvement of triangle t_0

If we assume that the input triangulation has a boundary point distribution that represents well the local feature size of the geometry boundary (the assumption allows to avoid boundary troubles which will be surmounted in practice by using the boundary treatment technique of section 4), the following theorem holds (Rivara 1996b):

Theorem 1. For any Delaunay triangulation T, the repetitive use of the Backward-LE-Delaunay-Improvement technique over the worst triangles of the mesh (with smallest angle $\alpha < 30^o$) produces a quality triangulation of smallest angles greater than or equal to 30^o.

Proof. The proof is based on the properties of both the longest-edge refinement algorithms and the Delaunay triangulation. In effect, the pure backward longest-edge bisection algorithm (Rivara 1996a; 1996b) essentially adds to the current set of vertices, the midpoint of the longest-edge of the last greatest triangle of the LEPP(t), which in turn is inserted by longest-edge partition of the associated pair of terminal triangles of the current LEPP(t). This work produces nested triangles, and an adequate point distribution which guarantees that the percentage of good-quality triangles (and the area covered by these triangles) increases throughout the process (Rivara 1992). However, some bad triangles still remain in the mesh due to the fact that the longest-edge refinement algorithms produce stable molecules around the vertices (after a small number of triangle partitions, the angles that share a vertex are fixed and not refined anymore (Rivara and Venere 1996)).

When the Basic Backward-LE-Delaunay-Improvement procedure is used in exchange, the midpoint of the longest-edge of the last greatest triangle of the LEPP(t) is also added, which improves the point distribution in the sense of the pure longest-edge algorithms. However, since this point is Delaunay inserted in the current triangulation, this local procedure improves the current triangulation in the following two senses: (1) the most equilateral mesh for the set of vertices is obtained; (2) the worst angles of the (non-fixed) molecules are eliminated (by edge swapping). If the triangle t is not destroyed throughout the process, the new LEPP(t) is found over a locally improved triangulation; and as a consequence, the addition of the midpoint of the longest-edge of the last greatest triangle of the new LEPP(t), improves even more the current point distribution; while that the Delaunay insertion of this point again improves the triangulation in the two senses stated before; and so on. This process guarantees that the Basic Backward-LE-Delaunay algorithm produces good-quality triangulations with smallest angles greater than or equal to 30°. Note that, smallest angles of more than 30º cannot be assured, since the longest-edge Delaunay partition of equilateral triangles can produce angles of 30º ¦

It should be pointed out here that, even when Theorem 1 guarantees the construction of quality triangulations, it says nothing about the size of these triangulations. More mathematical results in this sense are certainly needed. However, in practice, the 2-dimensional triangulations obtained are size-optimal (they are of analogous quality as those obtained with the circumcenter point insertion strategy).

4. A LONGEST-EDGE BOUNDARY TREATMENT STRATEGY

The basic LEPP Point Insertion algorithm of the preceding section performs well in practice whenever the input geometry has an adequate initial distribution of boundary points. This is intuitively due to the fact that, in this case, for each t, the pair of terminal triangles associated with the LEPP(t) are interior triangles (the edges of last triangle of the path are not along the geometry boundary) and consequently interior points (far enough from the boundary) are inserted.

Enough care should taken in exchange when the LEPP(t) finishes in the geometry boundary. To illustrate this idea consider the simple example of Figure 2a. In this case the naive use of the LEPP point insertion algorithm would produce undesirable interior points (as in Figure 2b). To avoid this effect the following boundary treatment technique has been introduced (Rivara, 1996b), where t is the last greatest triangle of the current LEPP:

Boundary-Treatment-Procedure (T, t, P)
If t has a boundary edge *l*, and *l* is not the
  smallest edge of t, then select P, the midpoint of *l*
Else select P, the midpoint of the longest-edge of t

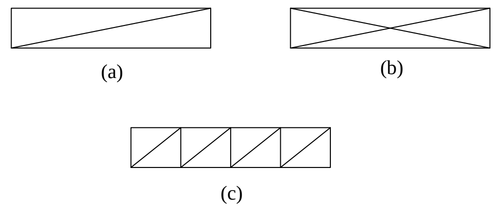


Figure 2

Note that the The Boundary Treatment procedure introduces in practice a boundary point distribution which adapts naturally and automatically to the local feature size of the geometry boundary (as in Figure 2c).

5. A (NON-ORDERED TRIANGLES) POLYGON TRIANGULATION ALGORITHM

By combining the techniques of sections 3 and 4, a simple 2-dimensional quality-triangulation algorithm can be formulated:

Non-Ordered-Triangles-Quality-Polygon-
  Triangulation ($\wp$, e)
Input: A general polygon $\wp$ (defined by a set of vertices
  and edges); and a tolerance parameter e(e<30º)
Construct T, a constrained (boundary) Delaunay
  triangulation of $\wp$.
Find S, the set of the worst triangles t of T (of smallest
  angle a_t<e)
For each t in S do
  Backward-LE-Delaunay-Improvement (T, t)
  Actualize the set S (by adding the new small-angled
  triangles and eliminating those destroyed through
  the process)
End for

Backward-LE-Delaunay-Improvement (T, t)
While t remains without being modified do
  Find the LEPP(t), and t* the last triangle in
    the LEPP(t)
  If t* has a boundary edge *l*, and *l* is not the smallest
    edge of t, then select P, the midpoint of *l*
  Else select P the midpoint of the longest edge of t*
  Perform the Delaunay insertion of P

Note that: (1) e is a threshold parameter less than or equal to 30º that can be easily adjusted; (2) in practice we have worked with a constrained Delaunay triangulation of the 2-dimensional geometry (Chew, 1989); (3) we have called Basic Non-Ordered algorithm to the quality-triangulation algorithm of this section to emphasize the fact that the triangles of the set S are maintained and processed in any order.

6. AN (ORDERED TRIANGLES) ALGORITHM FOR THE POLYGON TRIANGULATION PROBLEM.

In this version of the algorithm, an order is previously introduced and maintained over the set of the worst triangles S: the triangles are sorted according to their smallest angle and processed in this order.

At this point the following remarks are in order:

1. No relevant differences have been obtained in practice in the size and quality of the output triangulation for both Basic algorithms (Ordered versus Non-Ordered versions) and the same value of the ε parameter (see section 8).
2. The ordered algorithm can be easily parameterized to produce the best mesh for a maximum number of allowable triangles or vertices. This is a desirable and important feature in complex applications.
3. In both cases (Ordered an Non-Ordered algorithms), a suitable data structure that explicitly manage the neighbor-triangle relation should be used. In addition, since at each iteration within the while loop the LEPP(t) may or not be shortened, and may include new triangles not previously included in the LEPP(t), the current LEPP(t) should be updated, rather that computed from scratch whenever the triangle t still exists in the current mesh.
4. Both algorithms have a kind of self corrective property, in the sense that for the first triangles processed, the Delaunay insertion of the corresponding points, destroys (and improves) most of the worst triangles of S. This property has suggested the algorithm of the next section.

7. AN ALL TRIANGLES ALGORITHM

The following algorithm processes all the current bad triangles together as follows : (1) the LEPP(t) and the midpoint of the longest edge of this path is first found for all the triangles of S; (2) the Delaunay insertion of the set of the last points is then performed; (3) the set S of the worst triangles is actualized.

Quality-Polygon-Triangulation (℘, e)
Input: A general polygon ℘ (defined by a set of vertices and edges); and a tolerance parameter e(e<30º)
Construct T, a constrained (boundary) Delaunay triangulation of ℘.
Find S, the set of the worst triangles t of T (of smallest angle a_t<e)
While S ≠ ϕ do
  For each t in S do
    Backward-LE-Point-Insertion (T, t)
    Actualize the set S (by adding the new small-angled triangles and eliminating those destroyed through the process)
  End for
End while

Backward-LE-Point-Insertion (T, t)
Find the LEPP(t), and t* the last triangle in the LEPP(t)
If t* has a boundary edge l, and l is not the smallest edge of t, then select P, the midpoint of l
Else select P the midpoint of the longest edge of t*
Perform the Delaunay insertion of P

In spite of the arbitrary order used in this variant of the algorithm, it also tends to produce, in practice, the best mesh for a maximum number of allowable triangles or vertices (self corrective property).

8. 2-DIMENSIONAL EXPERIMENTATION

Empirical experimentation with the 2-dimensional algorithms (Palma, 1996), has shown they produce triangulations of analogous size and quality as the circumcenter algorithm (Ruppert, 1995) in the sense that optimal size meshes are obtained with smallest angle greater than 30º. Note however that the LEPP-algorithms have the following practical advantages: (1) the LEPP(t) is always interior to the polygon geometry; (2) the points inserted in the LEPP(t) are always midpoints of the longest edge of one or two known triangles of the current mesh. This knowledge can be used to improve the efficiency and robustness of the Delaunay routine in the following sense: the search of the triangle that contains the point to be inserted is avoided, and the initial (locally) non-Delaunay triangulation can be easily obtained by longest-edge partition of the involved triangles.

The triangulations of Figures 3, 4, and 5 illustrate the practical behavior of the LEPP-algorithms. For these three examples the input was the polygon with the minimum number of vertices required to describe the geometry. Thus, the triangulation of Figure 3 was obtained automatically from an 18-vertices input polygon; the algorithm inserted the remaining vertices on the boundary. For the example of Figure 4, most of the vertices inserted are interior vertices. Figure 5 illustrates the case where an interior edge has to be respected.

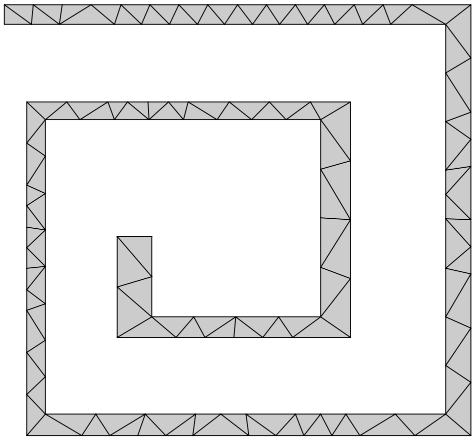


Figure 3

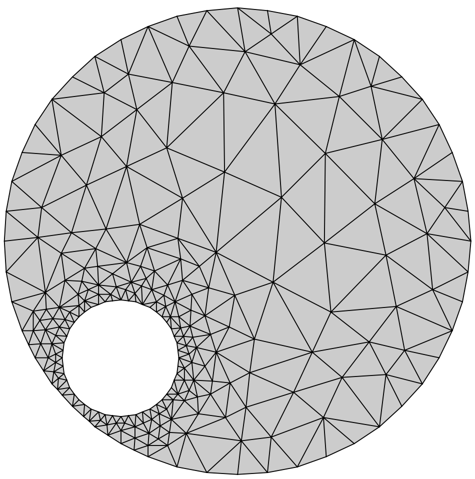
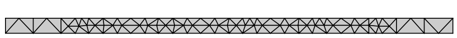


Figure 4

a)



b)

Figure 5. (a) input polygon (b) final triangulation

For the triangulation of Figure 6 in exchange, an initial uniform distribution of 400 boundary vertices was considered and, the aspect ratio (the ratio between the longest edge and the minimum height) criterion was used to control the current mesh quality.

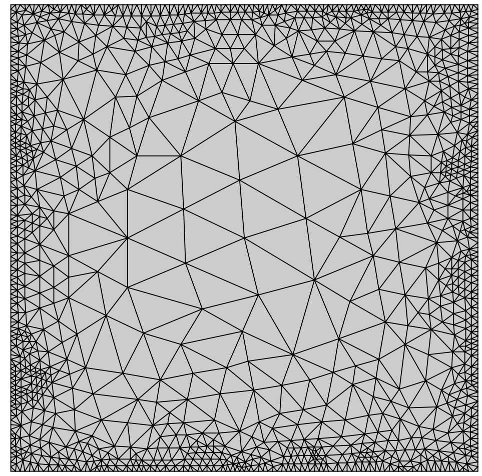


Figure 6

We have also used the square example to study the number of interior vertices inserted as a function of different (uniform) distributions of boundary vertices. In this sense the behavior of the algorithm was optimal: the number of interior vertices inserted is approximately a linear function of the number of input boundary vertices.

The square geometry with an input boundary distribution of 400 vertices was also used to perform the numerical experiments reported in Figures 7 and 8. In particular, Figure 7 summarizes the minimum angle behavior throughout the process for the Ordered algorithm of section 5. An essentially increasing function was obtained until an angle of approximately 28º was reached. A slightly oscillatory behavior is observed between 28º and 30º. This result compares advantageously with the example reported in (Ruppert, 1995) which shows an highly oscillatory behavior. Figure 8 reports the number of bad triangles obtained throughout the process for three different intervals of smallest angles ([0º, 10º), [10º, 20º) and [20º, 30º)). Clearly, the algorithm tends to eliminate, in increasing order, the smallest-angled triangles of the mesh. The same kind of behavior holds for the Non-Ordered and the All Triangles algorithms.

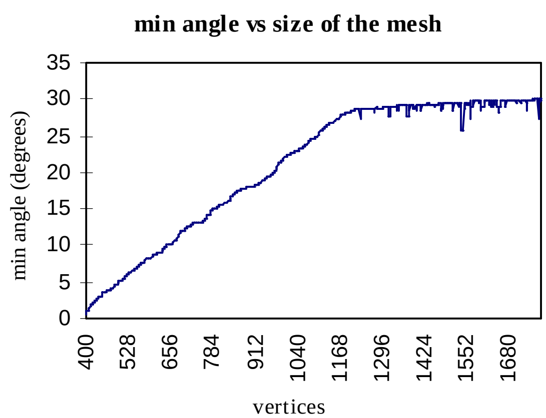


Figure 7

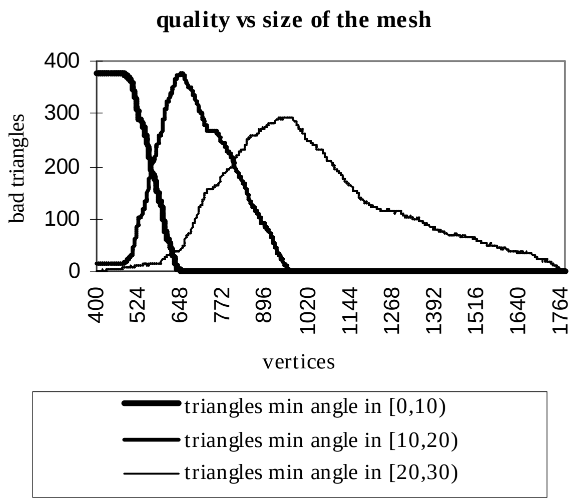


Figure 8

Finally, the 2-dimensional LEPP-algorithms have been successfully used to obtain quality surface triangulations of different polyhedra. To this end, the Non-Ordered algorithm of section 5 was used to triangulate the faces, combined with an adequate (iterative) boundary communication procedure between pairs of adjacent faces (both face triangulations must share the same vertices in the final surface mesh). Figures 9 and 10(a) illustrate the use of the surface LEPP Delaunay algorithm.

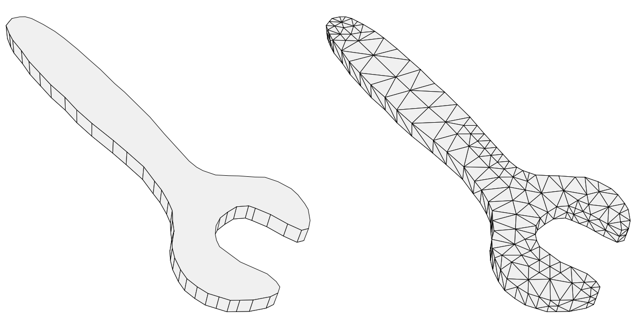


Figure 9

9. 3-DIMENSIONAL ALGORITHMS

Since in 3-dimensions the refinement propagates in several directions the following definitions are in order (Rivara and Levin, 1992).

Definition 3 The 3D-Longest-Edge Propagation Path of any tetrahedron t, is the set of all the neighbor tetrahedra (by the longest edge) having respective longest edge greater than or equal to the longest edge of the preceding tetrahedra in the path.

Definition 4 A terminal-tetrahedra-set is the set of all the tetrahedra of the mesh that share their common longest edge.

Proposition 2 The 3D-Longest-Edge Propagation Path of any tetrahedron has, in the general case, a set of N terminal-tetrahedra-sets, with N>1.

By using these concepts the basic backward algorithm of section 4 can be generalized to 3-dimensions as follows:

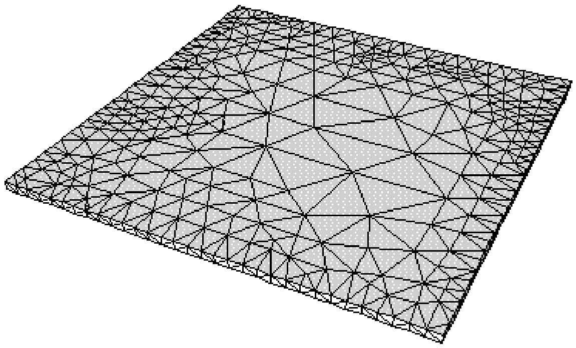
3D-Backward-LE-Delaunay-Improvement (T, t)
While t remains without being modified do
  Find the set of N terminal-tetrahedra-sets associated with the 3D-Longest-Edge Propagation Path of t
  Perform the Delaunay insertion of the midpoint of the longest edge of the terminal-tetrahedra-set having the greatest longest-edge

Note that in the preceding algorithm, the midpoint of the largest terminal edge is repeatedly inserted until the tetrahedron t is destroyed. This is different from the algorithm proposed in (Rivara 1996a, 1996b), where the midpoints of all the terminal-tetrahedra-set are simultaneously inserted. It should be also noted that a quite rough boundary treatment strategy (a direct extension of the 2D case) has been considered in this paper.

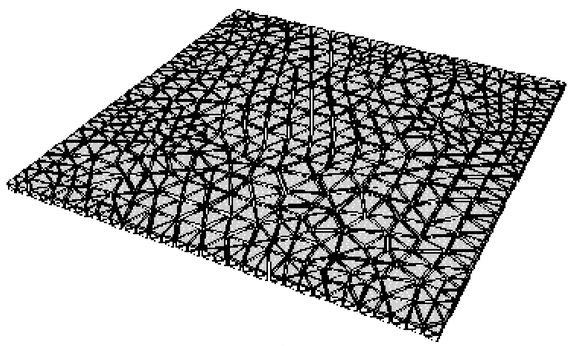
It is well known that the ability of the Delaunay technique to produce quality triangulations in 3-D is a strong function of the point placement algorithm. Since the pure 3D longest-edge refinement algorithm improves in practice the point distribution over the 3D geometry (Rivara and Levin 1992), we postulate that the combination of the 3D backward longest-edge Delaunay technique and an appropriate boundary treatment technique should constitute an effective triangulation improvement tool. Empirical experimentation performed has shown that, in practice, the 3-dimensional LEPP point insertion strategy behaves as expected both for the elimination of undesirable slivers, and for the improvement of the mesh. The examples reported in Figures 10 to 13 illustrate this behavior.

To study the mesh quality we have used a tetrahedron the quality indicator computed as the nomalized ratio between the tetrahedron volume and the cube of its longest-edge, in such a way that, for the equilateral tetrahedron, the indicator is equal to 1. Note that for sliver tetrahedra the quality indicator is near to 0.

In order to obtain the volume triangulation of the plate example of Figure 10, a good-quality surface triangulation was first obtained (see Figure 10a) by using the LEPP surface algorithm directly applied to the 8-vertices polyhedron geometry. Then by using the surface vertices as input data, the initial 3D Delaunay triangulation was constructed (700 vertices, 1710 tetrahedra and 0.056 minimum quality indicator), which was in turn improved by using the 3D LEPP-Delaunay improvement algorithm.



(a)



(b)

Figure 10

Figure 10b illustrates the volume mesh obtained after adding 230 new vertices (930 vertices, 2431 tetrahedra and 0.125 minimum quality indicator). Note that, as expected for this example, an almost uniform mesh was obtained. Figure 11 summarizes the behavior of the minimum quality indicator for the plate example throughout for the 3D improvement algorithm. Analogously to the 2-dimensions, the algorithm clearly tends to eliminate the worst tetrahedra of the mesh.

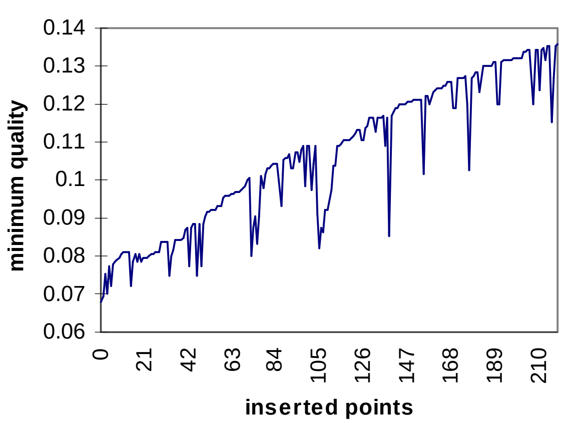
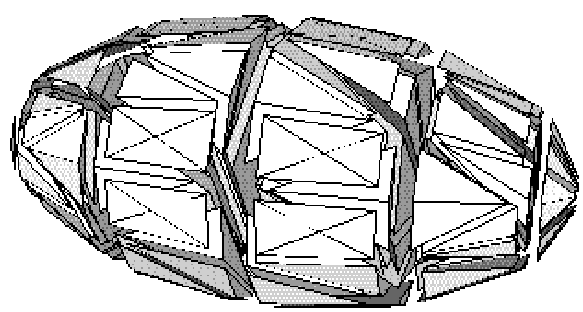


Figure 11

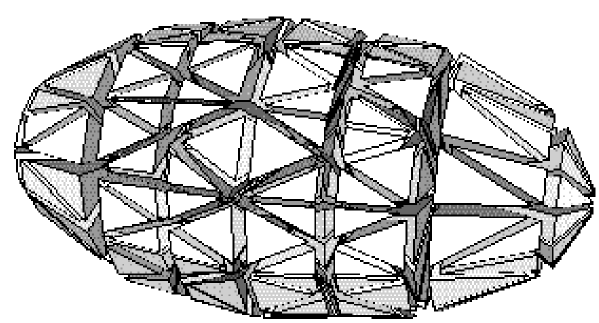
The second test problem consists on a spheroid geometry. Figure 12a shows the initial Delaunay triangulation used, which has 16 boundary slivers (62 vertices, 168 tetrahedra and 0.004 minimum quality indicator). Figure 12b illustrates the improved volume mesh obtained after adding 59 new vertices to the mesh (130 vertices, 392 tetrahedra and 0.098 minimum quality indicator). Figure 13 reports the number of tetrahedra obtained throughout the process for three different intervals of quality indicator. Note that, as in the 2-dimensional case, the LEPP-Delaunay improvement algorithm quickly eliminated the worst tetrahedra of the mesh (by adding the first 10 vertices).

At this point the following concluding remarks are in order :

1. The 3D algorithm considered in this paper does not make use of the surface triangulation information. Research in progress takes advantages of this information both for the design of an intelligent 3D boundary treatment technique and for constructing an initial constrained Delaunay triangulation of non-convex objects.
2. Theoretical research concerning the mathematical properties of the 2D LEPP-Delaunay algorithms is also in progress, and will be published else where.



(a)



(b)

Figure 12

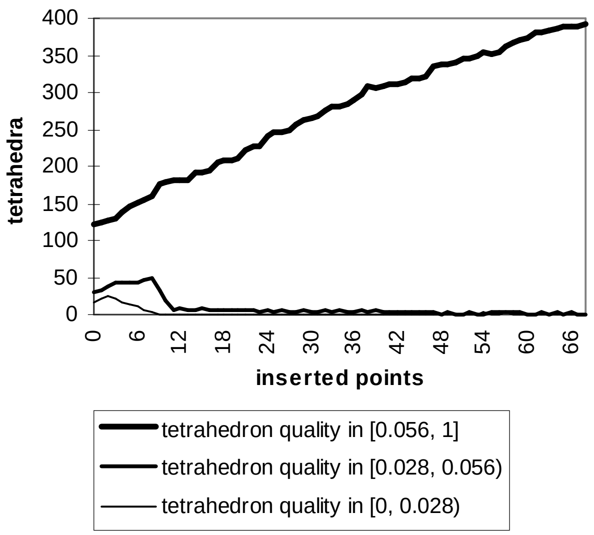


Figure 13

Acknowledgments:

This work has been partially supported by Proyecto Fondecyt 1940323.

References

Chew, L. P. Constrained Delaunay triangulation, Algorithmica 4(1989), 97-108.

Chew, L. P. Guaranteed-quality triangular meshes, Technical Report TR-89-983, Cornell University, 1989.

Liu, A. and Joe, B. On the shape of tetrahedra from bisection, Mathematics of Computations, 63(1994), 141-154.

Palma, M. Triangulación automática de polígonos y superficies de poliedros, Departamento de Ciencias de la Computación, U. de Chile, Octubre 1996.

Rivara, M. C. Algorithms for refining triangular grid suitable for adaptive and multigrid techniques, International Journal for Numerical Methods in Engineering, 29(1984), 745-756.

Rivara, M. C. Selective refinement/derefinement algorithms for sequences of nested triangulations, International Journal for Numerical Methods in Engineering, 28(1992), 281-290.

Rivara, M. C. and Levin, C. A 3D refinement algorithm suitable for adaptive and multigrid techniques, Communications in Applied Numerical Methods, 8(1992), 281-290.

Rivara, M. C. and Inostroza, P. A discussion on mixed (longest side midpoint insertion) Delaunay techniques for the triangulation refinement problem, Proceedings 4th International Meshing Roundatable, Sandia National Labs, New Mexico, 1995, pp. 335-346.

Rivara, M. C. and Iribarren, G. The 4-triangles longest-side partition of triangles and linear refinement algorithms, Mathematics of Computation, 65(1996), 1485-1502.

Rivara, M. C. and Venere, M. Cost analysis of the longest-side (triangle bisection) refinement algorithm for triangulations, Engineering with Computers, 12(1996), 224-234.

Rivara, M. C. New mathematical tools and techniques for the refinement and/or improvement of unstructured triangulations, Proceedings 5th International Meshing Roundtable 96, Pittsburgh, 1996a, pp. 77-86.

Rivara, M. C. New longest-edge algorithms for the refinement and/or improvement of unstructured triangulations, International Journal for Numerical Methods in Engineering, 1996b, To appear.

Rivara, M. C. and Inostroza, P. Using longest-side bisection techniques for the automatic refinement of Delaunay triangulations, International Journal for Numerical Methods in Engineering, 40(1997), 581-597.

Ruppert, J. A Delaunay refinement algorithm for quality 2-dimensional mesh generation, Journal of Algorithms, 18(1995), 548-585.