AUTOMATIC (TRIANGULAR) MESH GENERATION BASED ON LONGEST-EDGE ALGORITHMS*

Maria-Cecilia Rivara, Nancy Hitschfeld and Carlos Hasan
Department of Computer Science, University of Chile
Casilla 2777, Santiago, Chile
mcrivara@dcc.uchile.cl, nancy@dcc.uchile.cl, chasan@dcc.uchile.cl

Abstract. Several aspects of the LEPP algorithms (based on the use of the Longest-Edge Propagation Path of the triangles), for dealing both with the triangulation improvement problem and with the automatic quality triangulation problem, are reviewed and discussed. Applications producing quality nonobtuse triangulations and quality simplification for terrain modeling are also discussed.

Key Words: Mesh generation, quality triangulation, Delaunay algorithm, longest-edge algorithms, boundary nonobtuse triangulation, finite volume methods, terrain modeling

1 Introduction

In the adaptive finite element context, several mathematical algorithms for the refinement and/or derefinement of unstructured triangulations, based on the bisection of triangles by its longest-edge, have been discussed and used in the last 15 years.[1–4] These algorithms guarantee the construction of refined, nested and irregular triangulations of analogous quality as the input

*Submitted to the International journal of numerical methods, juli 1998

triangulation. However, the use of a new and related mathematical concept (the longest-edge propagation path of a triangle[5,6]), has allowed the development of new longest-edge algorithms for dealing with more general aspects of the mesh generation problem: (1) triangulation refinement problem, (2) triangulation improvement problem, (3) automatic quality triangulation problem, (4) quality nonobtuse triangulation problem, and (5) terrain modeling triangulation problem. In this paper, different aspects of these mesh generation problems and some of the algorithms proposed to deal with them, are reviewed, discussed and illustrated. In particular, the following specific applications are discussed: quality triangulations as needed for finite element methods; quality nonobtuse boundary triangulations as needed for mixed finite element and finite volume methods, and quality (surface) terrain simplification.

2 Mesh generation related problems

The polygon triangulation problem, an important issue for finite element applications, can be formulated as follows:

Definition 1 *Polygon Triangulation Problem: given N representative points of a polygonal region, join them by non intersecting straight line segments so that every region internal to the polygon is a triangle. The resulting triagulations is a conforming triangulation (the intersection of adjacent triangles is either a common vertex or a common side).*

Many criteria have been proposed as to what constitutes a "good" triangulation for numerical purposes, some of which involve maximizing the smallest angle or minimizing the total edge length. The Delaunay algorithm which constructs triangulations satisfying the first criteria has been of common use in engineering applications, followed by a postprocess step which assures the boundary respect of the polygon.

In the adaptive finite element context, the triangulation refinement problem is also critical. To state this problem, some requirements and criteria about how to define the set of triangles to be refined and how to obtain the desired resolution need to be specified. To simplify we shall introduce a subregion R to define the refinement area, and a condition over the diameter (longest-edge) of the triangles (given by a resolution parameter ε) to fix the desired resolution.

Definition 2 *Triangulation Refinement Problem: given an acceptable triangulation of a polygonal region Ω, construct a locally refined triangulation such that the diameters of the triangles that intersect the refinement area R are less than ε, and such that the smallest (or the largest) angle is bounded.*

In the case we dispose of a bad-quality triangulation of the polygonal geometry (having a non-adequate distribution of vertices) the triangulation improvement problem has to be considered. To state this problem, a triangle quality indicator function q(t), a tolerance parameter ε, and a local triangle improvement criterion need to be specified.

Definition 3 *Triangulation Improvement Problem: given a non-quality triangulation τ_0 of a polygonal region Ω (having triangles such that its quality indicator $q(t) < \varepsilon$), construct an improved triangulation τ such that each triangle t satisfies $q(t) \geq \varepsilon$.*

Note that if an initial coarse triangulation of the boundary polygonal vertices is considered, the more general (automatic) quality polygon triangulation problem can be stated.

Definition 4 *Quality Triangulation Problem: Given an initial (boundary) triangulation τ_0 of the boundary vertices which define the polygonal geometry, construct a geometry-adapted triangulation τ such that for each triangle t of $\tau, q(t) \geq \varepsilon$.*

For finite element/finite volume applications, the following Nonobtuse Boundary Triangulation Problem can be stated:

Definition 5 *Nonobtuse Boundary Triangulation Problem: given a quality Delaunay triangulation τ_0 of a polygonal region Ω, construct a triangulation τ such that the boundary triangles (having at least one boundary or interface edge) do not have an obtuse angle opposite to any boundary or interface edge.*

At this point the following remarks are in order:

(1) The triangulation problems stated in Definitions 2 to 5 are essentially different than the classical triangulation problem in the following sense: instead of having a fixed set of points to be triangulated, one has the freedom to choose the points to be added in order to construct a mesh either with a

desired resolution or with a given mesh-quality. The construction of the mesh is dynamically performed. Furthermore it is possible to exploit the existence of the reference triangulation (constructed for instance by means of the Delaunay algorithm) in order to reduce the computational cost to construct the output mesh.

(2) To cope with the triangulation Refinement Problem, the longest-edge refinement algorithms guarantee the construction of good quality irregular triangulations (section 4). This is due in part to their natural refinement propagation strategy farther than the (refinement) area of interest R. Furthermore, asymptotically, the number N of points inserted in R to obtain triangles of prescribed size, is optimal, and in spite of the unavoidable propagation outside the refinement region R, the time cost of the algorithm is linear in N, independent of the size of the triangulation.[7]

In the remaining of this paper, longest-edge based solutions both for the improvement and quality triangulation problems of Definition 3 and 4 will be discussed in the context of automatic quality triangulations (section 6), quality nonobtuse boundary triangulations (section 7), and triangulations for terrain modeling (section 8). Note that in all these applications, the algorithms take advantage of an LEPP point insertion technique (based on following the longest-edge propagation path of the target triangles) over Delaunay triangulations.

3 The longest-edge propagation path of a triangle

The longest-edge propagation path of a triangle concept is defined as follows:

Definition 6 *For any triangle t_0 of any conforming triangulation τ, the Longest-Edge Propagation Path of t_0 will be the ordered list of all the triangles $t_0, t_1, t_2, ...t_n$, 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 Ω, 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 7 *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 edge.*

It should be pointed out here 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 illustrate these ideas, see Figure 1(a), 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.

4 LEPP-Bisection Algorithm for the refinement of quality triangulations

By using the LEPP(t) concept, an improved Longest-Edge refinement algorithm[5,6] for non-Delaunay triangulations can be formulated, where the pure longest-edge refinement of a target triangle t_0 (see Figure 1) essentially means the repetitive longest-edge partition of pairs of terminal triangles associated with the current LEPP(t_0), until the triangle t_0 itself is partitioned.

The Figure 1 illustrates the refinement of the triangle t_0 over the initial triangulation of Figure 1(a) with associated LEPP(t_0)=$\{t_0, t_1, t_2, t_3\}$. The triangulations (b) and (c) illustrate the first 2 steps of the LEPP-Bisection procedure and their respective current LEPP(t_0), while that triangulation (d) is the final mesh obtained. Note that the new vertices have been enumerated in the order they were created.

The LEPP-Bisection procedure, schematically described in Figure 2 is a non-recursive algorithm essentially based on refining pairs of terminal triangles, where the concept of the Longest-Edge Propagation Path of the triangle t is repeatedly used (over the current triangulation) in order to find the last 2 (terminal) triangles of the path, until the initial triangle i is bisected.

Since the LEPP-Bisection algorithm is an improved version of the original longest-edge bisection algorithm, the following Theorem, based on the properties of the longest-edge bisection also holds:

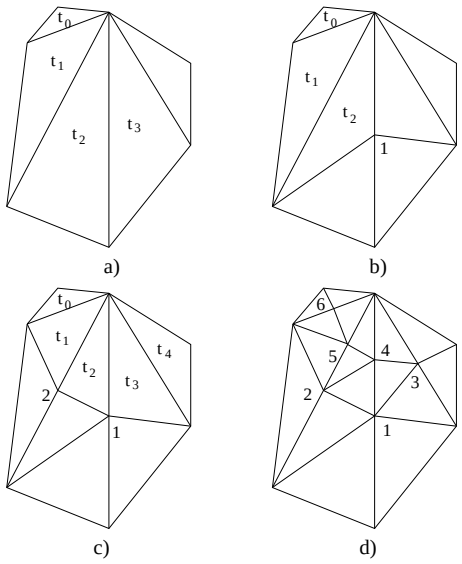

Figure 1: LEPP-Bisection of triangle t_0 (a) Initial Triangulation. (b) First step of the process. (c) Second step in the process. (d) Final triangulation

Theorem 1 *(a) The repetitive use of the pure longest-edge bisection algorithms, in order to refine t_0 and its descendants (triangles nested in t_0), tends to produce local quasi-equilateral triangulations. (b) The smallest angle α_t of any triangle t obtained throughout this process, satisfies that $\alpha_t \geq \alpha_0/2$, where α_0 is the smallest angle of t_0. (c) For any conforming triangulation τ, the global iterative application of the algorithm covers, in a monotonically increasing form, the area of t with quasi-equilateral triangles.*

Theorem 1 guarantees the construction of good-quality irregular and nested triangulations. Theorem 2 assures in exchange that the LEPP-Bisection algorithm solves the triangulation refinement problem with linear time complexity, provided that an initial good quality triangulation is used. To this end, 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) (see Figure 1), the current LEPP(t) should be updated, rather than computed from scratch in order to get the linear running time. Furthermore, the new LEPP Refinement Algorithm

```
LEPP-Bisection (t,T)
while t remains without being bisected do
  Find the LEPP(t)
  if  t*, the last triangle of the LEPP(t), is a terminal boundary triangle
  then
    bisect t*
  else
    bisect the (last) pair of terminal triangles of the LEPP(t)
  end if
end while
```

Figure 2: LEPP-Bisection procedure

produces the same triangulation as the previous recursive algorithm, in a simpler, cleaner, easy-to-implement and more direct way.

Theorem 2 *Let τ be any conforming triangulation of any bounded polygonal region Ω. Then, for any circular refinement subregion C of radious r, the use of the LEPP-Bisection algorithm to produce triangles of size ε inside C, asymptotically introduces N_i points inside C and N_o points outside C, where*

$$N_i = O(n^2),\ N_o = O(n \log n),\ \textit{and}\ n = \frac{2r}{\varepsilon}$$

5 LEPP-Delaunay algorithm for the improvement of triangulations

The LEPP Delaunay improvement algorithm uses the Longest Edge Propagation Path of the target triangles (to be improved in the mesh) over the Delaunay triangulation of the vertices, in order to decide which is the best point to be inserted to produce a good quality distribution of points.[5,6] This basic algorithm generalizes the ideas introduced in Rivara et al.[8,9]

LEPP-Delaunay-Improvement (t, T){
Input: T Delaunay triangulation
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))
end while}

Figure 3: LEPP-Delaunay improvement procedure

For an illustration of the algorithm see Figure 4, where the triangulation (a) is the initial Delaunay triangulation with LEPP(t_0) = $\{t_0, t_1, t_2, t_3\}$, and the triangulations (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 three additional Steiner points. Each one of these points is the midpoint of the longest-edge of the last triangle of the current LEPP(t_0). It should be pointed out here that each Delaunay point insertion locally improves the triangulation in the current LEPP(t_0), and in this sense this algorithm improves the triangulations obtained with the pure LEPP-Bisection procedure of section 4.

Note that we have used the word improvement instead of bisection or refinement. This is to make explicit 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 t in the sense of Theorem 3 of section 6.

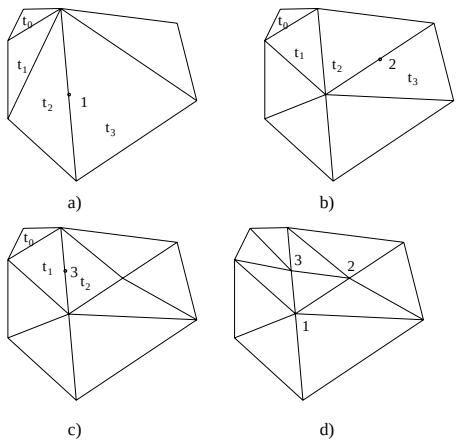

Figure 4: LEPP-Delaunay improvement of triangle t_0

6 Automatic algorithm for producing quality triangulations

By combining the basic LEPP-procedure over constrained Delaunay triangulations with adequate boundary considerations, a simple 2-dimensional automatic quality-triangulation algorithm can be formulated (see Figure 5), where δ is a threshold parameter less than or equal to 30° that can be easily adjusted. Furthermore, the following theorem holds:

Theorem 3 *For any Delaunay triangulation T, the repetitive use of the LEPP-Delaunay-Improvement algorithm over the worst triangles of the mesh (with smallest angle $\alpha < 30^\circ$) produces a quality triangulation of smallest angles greater than or equal to 30°.*

At this point the following remarks are in order:

(1) Even when Theorem 3 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. In fact, they are of analogous quality as those obtained with the circumcenter point insertion strategy.[10]

```
Quality-Polygon-Triangulation ( P, δ ) {
Input: A general polygon P (defined by a set of vertices and edges); and
a tolerance parameter (δ < 30°)
Construct T, a constrained (boundary) Delaunay triangulation of P.
Find S, the set of the worst triangles t of T (of smallest angle ¡ δ)
for  each t in S  do
   Backward-LE-Delaunay-Improvement (T, t)
   Update the set S (by adding the new small-angled triangles and elimi-
   nating those destroyed throughout the process)
end for}
LEPP-Delaunay-Improvement (T, t) {
while  t remains without being modified do
   if t* has a boundary edge l, and l is not the smallest edge of t then
      select p, the midpoint of l
   else
      Find the Lepp(t), and t* the last triangle in the Lepp(t)
      select p midpoint of the longest edge of t*
   end if
   Perform the Delaunay insertion of p
end while}
```

Figure 5: LEPP-Delaunay procedure with boundary considerations

(2) The triangulation of Figure 6 illustrates the practical behavior of the algorithm. Note that the input data was the polygon with the minimum number of vertices to describe the geometry.

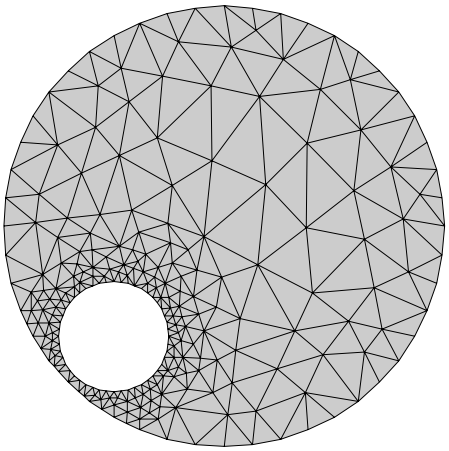

Figure 6: Automatic triangulation obtained (smallest angles greater than 30°)

7 Nonobtuse boundary triangulations

This section discusses an algorithm to solve the nonobtuse boundary problem of Definition 5, such as needed for mixed control volume discretization and finite element method.[11] This extends the LEPP-Delaunay algorithm of section 5. Nonobtuse boundary triangulations are also very useful when problems are solved combining both methods. This requires the combination of good quality meshes and well shaped Voronoi boxes, which implies both that the minimum angle should be bounded and that boundary triangles should not have obtuse angles opposite to any boundary edge or interface edge.

The generation of quality nonobtuse boundary Delaunay triangulations consists of two steps: (1) The construction of a good quality (constrained) Delaunay triangulation (CDT) of the polygon having interior angles comprised between 30° and 120°, (2) A postprocess step which eliminates boundary obtuse triangles by combining a longest-edge procedure for selecting points, the Delaunay algorithm for inserting the points and a finite number of some specific Delaunay point insertions for boundary obtuse triangles with 2-edge boundary constrained acute angles (angles defined by two boundary edges).

The construction of the good quality (constrained) Delaunay triangulation consists of: (a) The generation of an initial constrained Delaunay triangulation (essentially using the polygon vertices), and b) the use of the LEPP-Delaunay algorithm described in Figure 5 (section 6), with $\varepsilon = 30^\circ$, which produces a mesh with angles between 30° and 120°.

The step designed to eliminate boundary obtuse triangles of polygonal regions considers two cases: (a) triangles with only one boundary edge opposite to the obtuse angle (1-edge boundary obtuse triangles), and (b) triangles with two boundary edges and one of them opposite to the obtuse angle (2-edge boundary obtuse triangles)

A detailed description of the algorithm and the proof of the theorems of this section can be found in Hitschfeld et al.[12]

7.1 Elimination of 1-edge boundary obtuse triangles

Each 1-edge boundary obtuse triangle is simply eliminated by Delaunay insertion of the midpoint of the boundary edge. Since the obtuse angle is smaller than or equal to 120°, the insertion of only one point is required (see Theorem 4) even when some some diagonal swappings might be necessary.

Theorem 4 *Let τ be any improved Delaunay triangulation of any polygonal geometry (with smallest angle greater than or equal to $30°$). Let consider either an 1-edge boundary obtuse triangle, or two 1-edge interface triangles being at least one of them an obtuse triangle as shown in Figures 7(b) and 7(c) (one or two 1-edge interface obtuse triangles); In any of both cases, let consider e the unique boundary or interface edge involved. Then (a) the obtuse triangle is eliminated by Delaunay insertion of the midpoint of e; and (b) the new generated boundary triangles are nonobtuse triangles.*

The three possible cases considered by Theorem 4 are illustrated by Figure 7. Figure 7(a) shows a 1-edge boundary obtuse triangle, Figure 7(b) two 1-edge interface triangles where one of the angles opposite to the interface edge is obtuse and Figure 7(c) shows two 1-edge interface triangles where both angles opposite to the interface edge are obtuse. The Delaunay insertion of the interface edge midpoint in case (c) destroys the two obtuse angles not generating new obtuse angles opposite to interface edges.

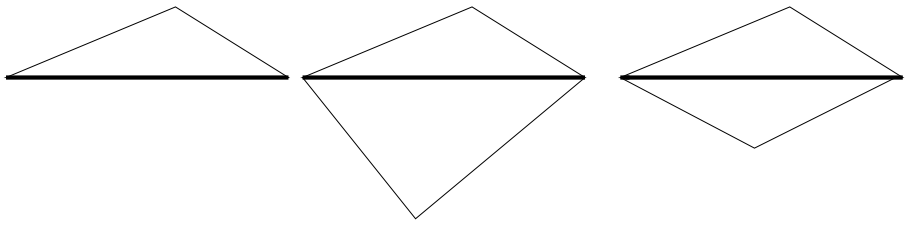

Figure 7: Cases of 1-edge boundary and interface obtuse triangles

Corollary 1 *For any improved Delaunay triangulation of any polygonal geometry the number of point insertions (V_1) to eliminate N_{1b} 1-edge boundary obtuse triangle and N_{1i} interface obtuse triangles is $N_{1b}+\frac{N_{1i}}{2} \leq V_1 \leq N_{1b}+N_{1i}$.*

The insertion of $N_{1b} + \frac{N_{1i}}{2}$ is possible when the improved triangulation has an odd number of interface obtuse triangles and all of them correspond to the case (c) of Figure 7.

7.2 Elimination of 2-edge boundary obtuse triangles

The elimination of 2-edge boundary obtuse triangles requires a special treatment when the smallest edge is an interior edge. Note that in such a case, the boundary constrained angle β is the smallest angle of the triangle (β is the angle defined by the two boundary edges). The strategy used for the 1-edge boundary obtuse triangles can not be applied in this particular case since after two applications of the strategy, a new triangle similar to the original one will be obtained, (t_o is similar to t_4 in Figure 8). One additional problem is that due to the boundary restrictions the minimum angle of this triangle can be less than 30° and consequently, the obtuse angle can be greater than 120°.

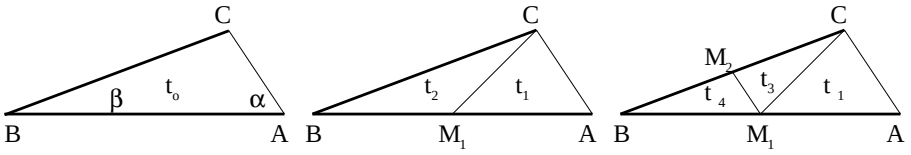

Figure 8: t_o is similar to t_4

The essential ideas of the algorithm to handle case 2 illustrated in Figure 9 are the followings: An 2-edge boundary isosceles triangle of largest edges equal to half the smallest boundary edge of the target triangle is constructed (triangle BMN in Figure 9(b)). Since the points M and N are inserted using the Delaunay criterion, this construction maintains the Delaunay property of the mesh but can produce an 1-edge boundary obtuse triangle t_1, which is in turn eliminated by the Delaunay insertion of the midpoint of the largest edge of t_1 (Figure 9(c)). This procedure can again produce a new boundary obtuse triangle t_1, with largest angle smaller than the previous one and so on. The boundary obtuse triangles are eliminated after the Delaunay insertion of a finite number of points.

Theorem 5 *Let t be a 2-edge boundary obtuse triangle with interior smallest edge. Then: (1) If the constrained angle is greater than β_0 (with $\beta_0 > 32.53°$), the obtuse angle is eliminated by Delaunay insertion of exactly two points obtained by creating a 2-edge boundary isosceles triangle; (2) If the constrained angle is less than or equal to β_0, a new 1-edge boundary obtuse triangle can be generated, which is eliminated by Delaunay insertion of a finite number of i points bounded by $i \leq 2\frac{\overline{C-M_0}}{\overline{M_0-N}}$ (see Figure 9).*

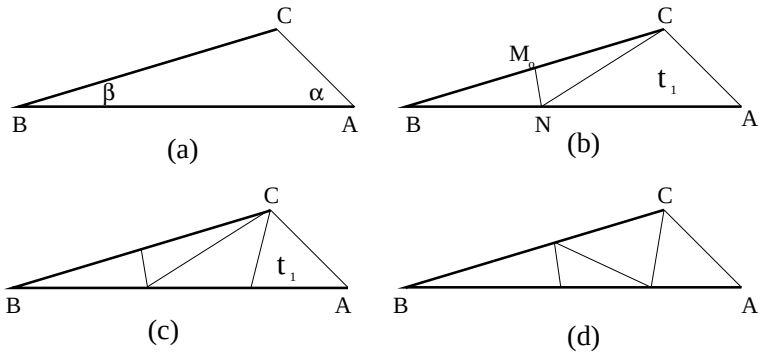

Figure 9: Elimination of 2-edge boundary obtuse triangles

Note that, the elimination of 2-edge boundary obtuse triangles with smallest interior edge might introduce triangles with angles less than 30°. The number of involved triangles depends on the number of point insertions and on the number of diagonal swapping made to eliminate the 2-edge boundary obtuse triangles.

Figure 10 illustrates the more complex case arising when interfaces with several interface edges converge to a common vertex A. In this case we eliminate the obtuse angles by Delaunay insertion of the midpoint M of the smallest interface edge and a point N_j on each edge j so that the distance between N_j and A is equal to the distance between M and A. Thus, isosceles triangles are generated around A.

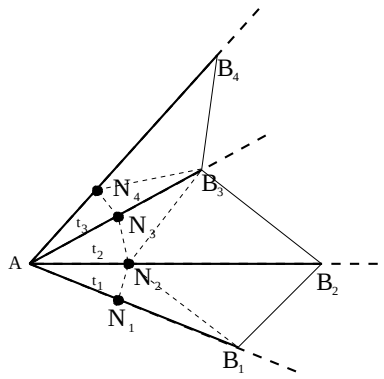

Figure 10: Adjacent boundary obtuse triangles

Proposition 2 *The number of point insertions to eliminate N boundary obtuse angles is $O(N)$.*

Corollary 2 *The final triangulation obtained having nonobtuse boundary and interfaces triangles is a Delaunay triangulation.*

7.2.1 Examples

Figure 11(a) shows a strip geometry with an "interior" interface edge. As expected, the number of inserted points to destroy the boundary and interface obtuse angles is less than or equal to the number of boundary obtuse triangles as shown in Table 1.

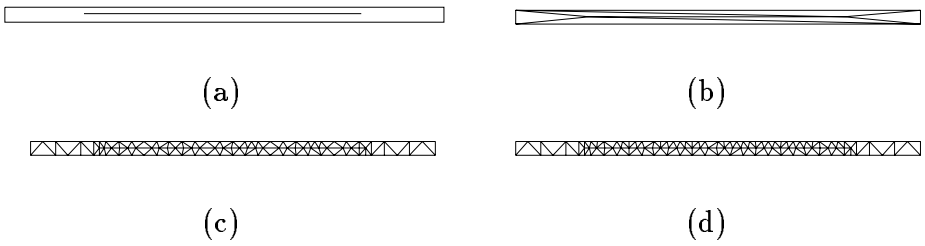

Figure 11: Example 1

	Example 1		
	Delaunay	LEPP-Del.	Final mesh
vertices	6	99	116
triangles	6	128	149
min. angle	1.00	30.77	30.77
aver. min. angle	4.10	43.53	44.72
max. angle	175.52	108.16	106.60
aver. max. angle	144.80	83.65	81.68
b-obtuse triangles	2	21	0

Table 1: Statistical information for the example 1 (Figure 11)

Figure 12(a) shows a two circles polygon with interior interface edges. The number of inserted points to eliminate boundary and interface obtuse

triangles (shown in Table 2) is in complete agreement with the theory. After the elimination of 2-edge boundary obtuse triangles, a small number of interior triangles with minimum angle less than 30° might appear. This is the case of this example where 16 triangles with minimum angle less than 30° still remain in the final mesh (obtained after the elimination of the boundary or interface obtuse triangles). These bad quality triangles could be easily eliminated by using the LEPP-Delaunay algorithm once more. It should be pointed out here however that some small geometry constrained angles, that can not be improved due to the geometry constraints of the problem, also appear in the final mesh.

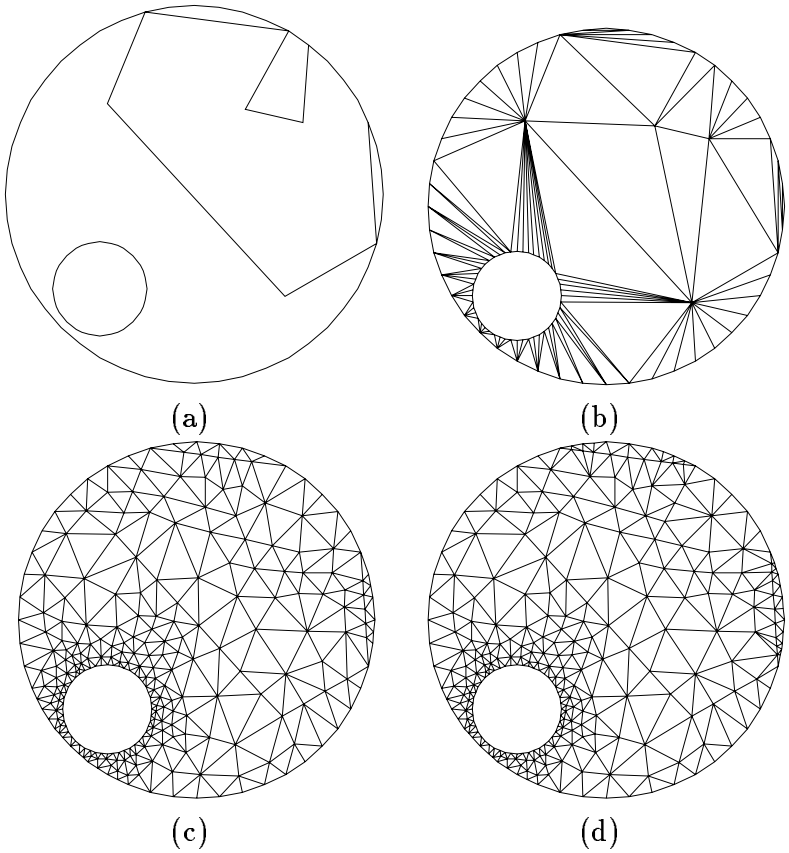

Figure 12: Example 2

	Example 2		
	Delaunay	LEPP-Del	Final mesh
vertices	100	272	291
triangles	104	434	463
min. angle	0.84	30.06	12.40
aver. min. angle	15.73	43.15	42.39
max. angle	172.49	115.17	126.820
aver. max. angle	111.87	79.80	80.49
b-obtuse triangles	9	8	0

Table 2: Statistical information for the example 2 (Figure 12)

8 Terrain modeling application

In this section, we consider the application of a (3D) surface LEPP-Delaunay algorithm (for improving a 3D-surface triangulation) combined with a simplification algorithm for terrain models in order to obtain good quality triangular meshes from digital elevation models.

A terrain is the graph of an scalar function of two variables. The function gives the elevation of each point in the domain. Terrain models are widely used in visualization and computer graphics applications such as geographic information systems, flight simulators and video games.

The most common source of terrain elevation data is the digital elevation model (DEM) which is basically a two-dimensional floating point height array. Several alternative representations have been proposed, including contour lines, quad-trees, and triangular irregular networks (TIN).

Triangulations stand out as being one of the most convenient formats for rendering and other geometric manipulation operations. The automatic generation of TINs from DEM models is an important research area and is the main topic of this section.

8.1 The 3D surface LEPP-Delaunay algorithm

In this context the sphere criterion is used in the Delaunay algorithm: the sphere criterion for data dependent Delaunay triangulations states that a triangulation T of a set of points P is Delaunay if and only if no point of P is interior to the circumsphere of any triangle of T. It means that most of the three-dimensional triangles are nearly equiangular.[13]

If the diagonal of any quadrilateral formed by two adjacent triangles of a Delaunay triangulation T is replaced by the opposite diagonal, then points of T will be interior to the circumsphere of these adjacent triangles. If the application of this criterion to each edge of an arbitrary triangulation T does not cause any edge swapping (ie. every edge is locally optimal), then T is a data dependent Delaunay triangulation.

The LEPP concept of the Definition 6 is easily extended to 3D surface triangulations by using the Euclidean distance in R^3.

However, since the 3D surface LEPP-Delaunay algorithm does not work properly for triangles located in areas with high local curvatures, a new geometric strategy has been developed to handle this problem.

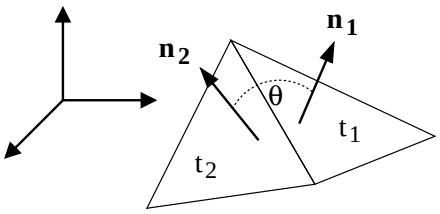

Figure 13: Local curvature angle used for three dimensional triangulations.

The local curvature angle θ of a pair of adjacent triangles t_1 and t_2 (Figure 13) is defined as $\theta = \arccos(n_1 \cdot n_2)$ where n_i is the normal to the triangle t_i. We define constrained edges e_c as all those edges located at the boundary of the mesh or whose local curvature angle θ_c is greater than or equal to a θ_{max} threshold. These edges will not be swapped by the data dependent Delaunay algorithm.

Triangles with minimum angle α formed by two constrained edges will not be improved (the data dependent Delaunay triangulation algorithm will never swap constrained edges), and consequently, these minimum angles will not be removed from the triangulation.

Finally, by combining all the previous techniques of this section, we can formulate a simple and effective algorithm to improve the geometric quality of terrain triangulations, which guarantees the construction of good quality triangulations. The algorithm depends both on a threshold angle θ_{max}, used to decide which edges are constrained (having value smaller than or equal to 180^0), and on a threshold angle α_{min} used to specify the minimum angle of all the triangles in the final triangulation.

8.2 Terrain simplification

Simplification of terrain surfaces is an important research area in computer graphics and geometric modeling. The terrain simplification algorithm is based on the combination of a greedy point insertion[14] and the extended LEPP-Delaunay algorithm for terrain meshes. The algorithm starts with a simple triangulation of the domain and, on each step, applies the refinement algorithm over the worst triangles in the triangulation to reduce the interpolation error in the current approximation of the digital elevation model. Whenever a new point is inserted in the triangulation by the LEPP-Delaunay

algorithm, its elevation is adjusted to fit the underlying digital elevation model.

8.3 Empirical results

The simplification LEPP-Delaunay algorithm for terrain surfaces produces good quality approximations (including good quality triangles) from DEM models.

Figure 14 illustrates different triangulations generated by the LEPP simplification algorithm from a digital elevation model of 346x452 points. The triangulations obtained include 4, 43, 275 and 937 vertices respectively, with maximal vertical error of 16.75, 6.1, 2.65 and 1.4 meters respectively, with the error defined as follows:

$$Error = \max_{(x,y)}(\overline{H(x,y) - H^*(x,y)}),$$

where $H(x,y)$ is the height value corresponding to DEM model and $H^*(x,y)$ is the height value corresponding to the triangular approximation.

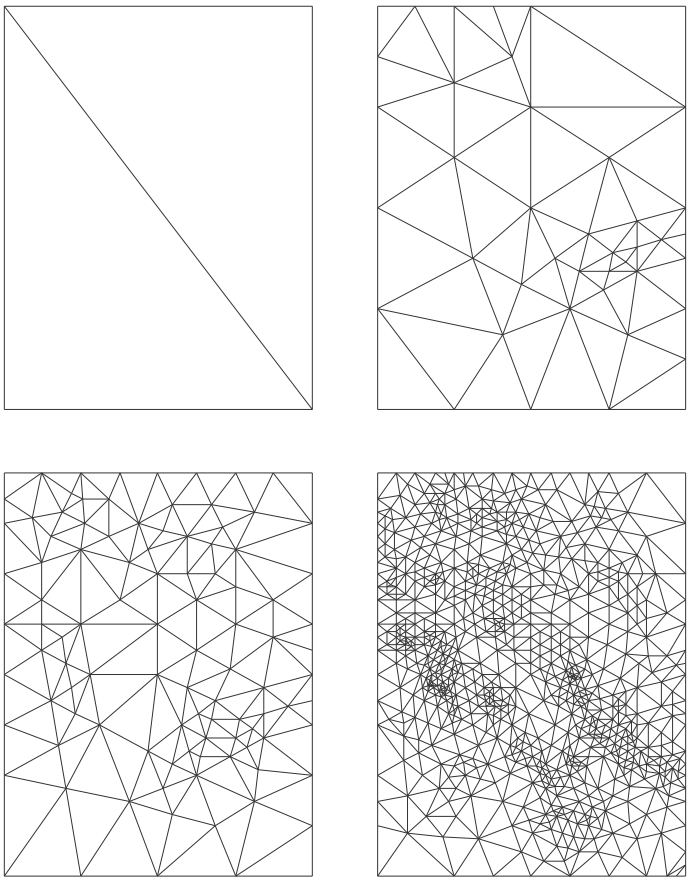

Figure 14: Triangulations generated by the simplification algorithm from a digital elevation model.

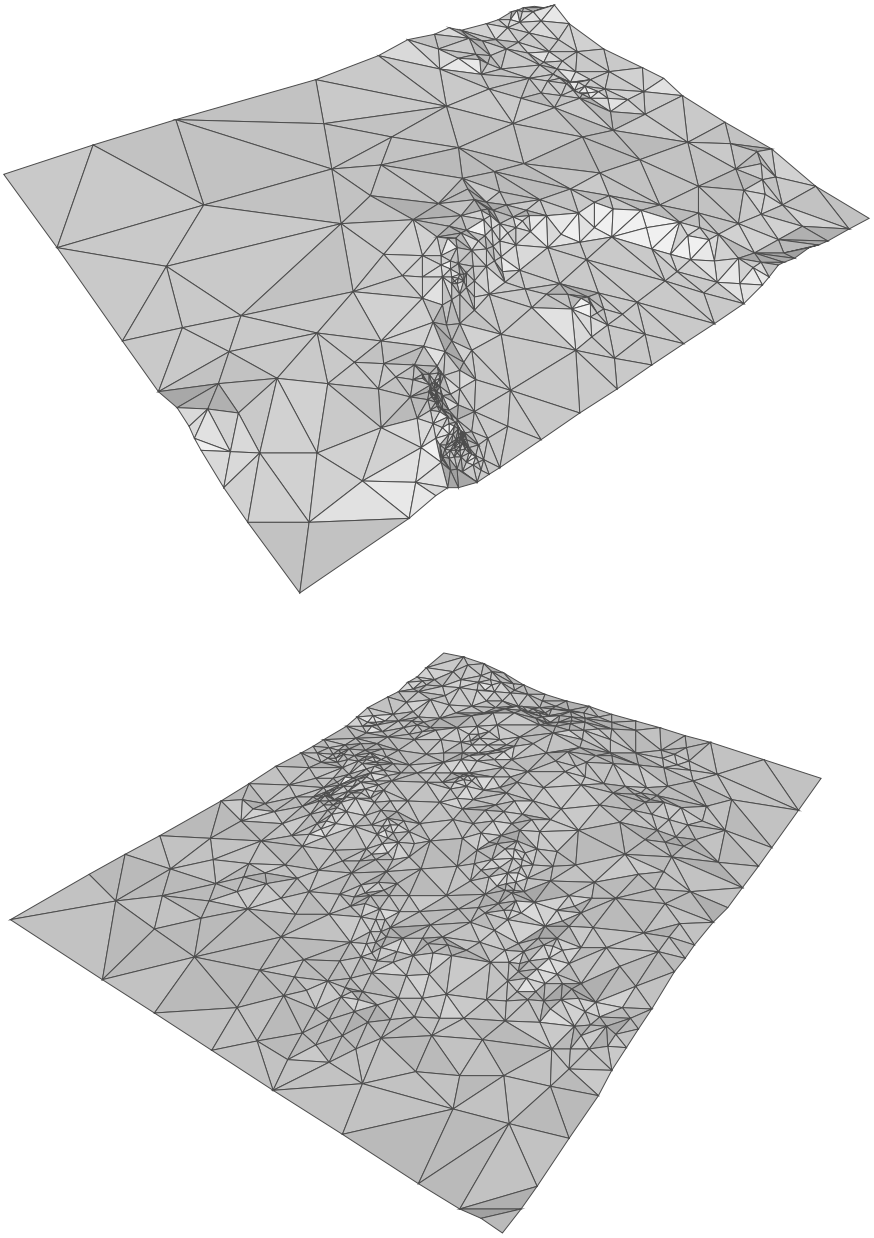

Figure 15: Triangular irregular networks generated by the LEPP simplification algorithm from digital elevation models of terrain surfaces.

Acknowledgements.

Some of the results presented in this paper were supported by Fondap AN-1 and Fondecyt 1981033.

References

[1] M.C. Rivara. Algorithms for refining triangular grids suitable for adative and multigrid techniques. *International journal for numerical methods in Engineering*, 20:745–756, 1984.

[2] M.C. Rivara. Selective refinement/derefinement Algorithms for Sequences of nested Triangulations. *International journal for numerical methods in Engineering*, 28:2889–2906, 1989.

[3] M.C. Rivara and C. Levin. A 3d Refinement Algorithm for adaptive and multigrid Techniques. *Communicatons in Applied Numerical Methods*, 8:281–290, 1992.

[4] M.C. Rivara and G. Iribarren. The 4-triangles longest-side Partition of Triangles and linear Refinement Algorithms. *Mathemathics of Computation*, 65(216):1485–1501, october 1996.

[5] M. C. Rivara. New mathematical tools and techniques for the refinement and/or improvement of unstructured triangulations. *Proceedings 5th International Meshing Roundtable. Pittsburgh*, pages 77–86, 1996.

[6] M.C. Rivara. New longest-edge algorithms for the refinement and/or improvement of unstructured triangulations. *International journal for numerical methods in Engineering*, 40:3313–3324, 1997.

[7] M.C. Rivara and M. Venere. Cost Analysis of the longest-side (triangle bisection) Refinement Algorithms for Triangulations. *Engineering with Computers*, 12:224–234, 1996.

[8] M. C. Rivara and P Inostroza. A discussion on mixed (longest side midpoint insertion) delaunay techniques for the triangulation refinement problem. *Proceedings 4th International Meshing Roundtable. Albuquerque*, pages 335–346, 1995.

[9] P. Inostroza M.C. Rivara. Using longest-side bisection techniques for the automatic refinement of delaunay triangulations. *International journal for numerical methods in Engineering*, 40:581–597, 1997.

[10] J Ruppert. A delaunay refinement algorithm for quality 2-d mesh generation. *Journal of algorithms*, 18:548–585, 1995.

[11] J. F. Bürgler. *Discretization and Grid Adaptation in Semiconductor Device Modeling*. PhD thesis, ETH Zürich, 1990. published by Hartung-Gorre Verlag, Konstanz, Germany.

[12] N. Hitschfeld and M.C. Rivara. Automatic construction of quality non-obtuse boundary delaunay triangulations. *Submitted to Mathematics of Computation. TR/DCC 98-2, Department of Computer Science, U. de Chile*, June, 1998.

[13] M. S. Shepard H. L. Cougny. Surface meshing using vertex insertion. *Scientific Computation Research Center, Rensselaer Polytechnic Institute, Troy, NY*, 1995.

[14] P. Heckbert M. Garland. Fast polygonal approximation of terrains and height fields. *School of Computer Science, Carnegie Mellon, University, Pittsburg, PA*, 1995.