

24th International Meshing Roundtable (IMR24)

New Results on LEPP-Delaunay Algorithm for Quality Triangulations

Carlos Bedregal^a, María-Cecilia Rivara^a

^a*Department of Computer Science, University of Chile, Av. Beauchef 851, Santiago, Chile*

Abstract

In this paper, we provide proofs of termination and size-optimality of the LEPP-Delaunay algorithm, for the quality generation of triangulations. We first prove that the algorithm cannot insert points arbitrarily close to each other. We also show that the algorithm terminates, producing well-graded triangulations with internal angles greater than 25.66 degrees for geometries with input constrained angles of at least 30 degrees.

© 2015 The Authors. Published by Elsevier Ltd.

Peer-review under responsibility of organizing committee of the 24th International Meshing Roundtable (IMR24).

Keywords: Delaunay refinement, longest-edge refinement, longest-edge bisection

1. Introduction

Given an initial triangulation τ of an input planar straight-line graph (PSLG) geometry composed of points and segments, and an angle parameter θ , a generic Delaunay refinement algorithm generates a θ -quality triangulation τ_{ref} such that every triangle t of τ_{ref} has angles greater than or equal to θ . To this end, for each *bad quality triangle* t in τ (with smallest angle $\alpha_t < \theta$), the algorithm selects a point p according to a defined point selection strategy and performs a Delaunay insertion of p in the current triangulation. The process is repeated until τ satisfies the desired quality property.

Ruppert's algorithm is considered the first provably-good Delaunay refinement algorithm with good practical performance [1,2]. Given a bad quality triangle t , either the circumcenter of t , or the midpoint of an encroached associated segment, is Delaunay inserted in the mesh. Ruppert's theoretical results guaranteed size-optimality, good grading and refined triangulations with angles greater than 20.7° , requiring input constrained angles greater than 90° .

Several studies of circumcircle-based Delaunay refinement algorithm were also introduced. Chew [3] presented an algorithm ensuring refined triangulations with angles between 30° and 120° . Shewchuk [4] later improved upon the results of Ruppert and Chew, producing triangulations with most angles greater than 30° , while requiring constrained angles greater than 60° (Shewchuk's open-source software, Triangle [5], is supported by this results).

E-mail address: cbedrega@dcc.uchile.cl

Further studies of circumcenter-based Delaunay refinement algorithms aimed to reduce the number of Steiner points inserted during refinement. Erten and Üngör [6] proposed the insertion of *off-centers*. Given the bad triangle t , their algorithm selects a point over the line segment joining the circumcenter of t and the midpoint of its smallest edge s , such that the new triangle containing s is of acceptable quality (the current version Triangle implements this ideas). Foteinos *et al.* [7] discussed a generalized, optimization-based, Delaunay refinement algorithm that selects points inside geometrical regions related to the circumcircle of the bad quality triangle and close to encroached segments. They guaranteed good grading and a lower bound angle on the angles of the refined triangulation close to 30° for geometries with constrained angles of at least of 60° .

The implementation of a circumcenter-based refinement algorithm is rather a cumbersome task (See Cheng *et al.* [8], Section 6.3)). An efficient implementation should maintain priority queues for bad quality triangles and encroached segments. Furthermore, the number of points of the final triangulation is sensitive to the order in which triangles are processed (since circumcenters of small-angled triangles tend to eliminate bad quality triangles faster).

Based on a longest-edge strategy, the LEPP-Delaunay algorithm [9,10] is a different, simple approach for quality Delaunay refinement, producing well-graded triangulations with internal angles of at least 30° . The algorithm computes the longest-edge propagating path (LEPP) associated with a bad quality triangle to find a local longest edge in the triangulation and inserts the midpoint of such edge. The performance of the algorithm is not affected by the order in which bad quality triangles are processed, and no priority queues have to be maintained, unlike Ruppert's algorithm. Previous studies have not provided complete proofs of the algorithm's size-optimality, termination or good grading; even though the algorithm shows optimal performance in practice.

In this paper we provide the theoretical proofs of size-optimality and termination of the algorithm, establishing the LEPP-Delaunay algorithm in the set of provably good Delaunay refinement algorithms. We adapt a taxonomy on the longest-edge bisection of triangles [11] and improve upon previous geometrical results on the algorithm [10] to establish bounds based on the length of existing edges and on the circumradius of associated terminal triangles, respectively. Our results guarantee that the insertion radius is at least half the length of the smallest edge of a triangle, or at least half its circumradius. Finally, we prove that LEPP-Delaunay algorithm produces triangulations with internal angles between 25.66° and 128.68° for input geometries with constrained angles of at least 30° .

2. LEPP-Delaunay and Longest-Edge Refinement

The LEPP strategy is based on the concepts of terminal triangles and propagation paths introduced by Rivara [9] (See Figure 1).

Definition 1. An edge e_{term} is called a *terminal edge* in triangulation τ_{ref} if e_{term} is the longest edge of every triangle that shares e_{term} . The triangles sharing e_{term} are called *terminal triangles*. For 2-dimensional triangulations, if e_{term} is shared by two terminal triangles then e_{term} is an *interior edge*; if e_{term} is shared by a single terminal triangle then e_{term} is a *boundary edge*.

Definition 2. For any triangle t_0 in τ_{ref} , the longest edge propagating path of t_0 , $\text{LEPP}(t_0)$, is the ordered sequence $\{t_j\}_0^{N+1}$, where t_j is the neighbor triangle on the longest edge of t_{j-1} , and $\text{longest_edge}(t_j) > \text{longest_edge}(t_{j-1})$, for $j = 1, \dots, N$. Edge $e_{\text{term}} = \text{longest_edge}(t_{N+1}) = \text{longest_edge}(t_N)$ is an *interior terminal edge* in τ_{ref} and this condition determines N . Therefore, either e_{term} is shared by the couple of terminal triangles (t_N, t_{N+1}) , or e_{term} is shared by a unique terminal triangle t_N with boundary (constrained) longest edge.

Given an input PSLG \mathcal{P} and a quality parameter θ , the LEPP-Delaunay algorithm produces a θ -quality triangulation τ_{ref} of \mathcal{P} based on the *constrained Delaunay triangulation* (CDT) of the input data (a relaxed Delaunay triangulation that includes the input (constrained) edges).

We only need to assume that no two constrained edges in the initial CDT meet at an angle less than 30° (in contrast, the proofs provided by Ruppert assume that input edges meet at angles of at least 90° [2], while later studies assumed input angles of at least 60° [4,7]).

For any bad quality triangle t (with smallest angle less than θ), the algorithm computes $\text{LEPP}(t)$ and finds the associated terminal triangles and terminal edge e_{term} . Then, it performs the constrained Delaunay insertion of the midpoint of e_{term} if neither of the terminal triangles has constrained midsize edge. Otherwise, the midpoint of a midsize

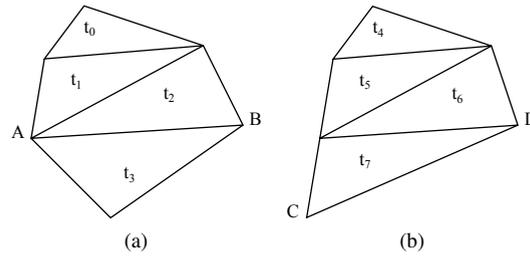


Fig. 1: (a) AB is an interior terminal edge shared by terminal triangles $\{t_2, t_3\}$ of $\text{LEPP}(t_0) = \{t_0, t_1, t_2, t_3\}$. (b) CD is a boundary terminal edge with terminal triangle $\{t_7\}$ of $\text{LEPP}(t_4) = \{t_4, t_5, t_6, t_7\}$.

edge is inserted in the triangulation (Figure 2(b)). The process is repeated until t is removed from the triangulation. Figure 2 illustrates the refinement process. Algorithm 1 describes the LEPP-Delaunay algorithm.

Algorithm 1 LEPP-Delaunay Algorithm

Input: PSLG polygon \mathcal{P} , tolerance parameter θ

Output: Constrained Delaunay triangulation τ_{ref} of quality defined by θ

Construct τ_{ref} the CDT of \mathcal{P} .

Find $S_{\text{target}} \in \tau_{\text{ref}}$, the set of bad quality triangles defined by θ

for each t in S_{target} **do**

if t has longest edge or midsize edge constrained **then**

 Perform constrained Delaunay insertion of midpoint of the constrained edge

else

while t remains in τ_{ref} **do**

 Find $\text{LEPP}(t)$, terminal triangles t_i, t_j and terminal edge e_{term} {triangle t_j can be null for boundary e_{term} }

 Select point P , midpoint of e_{term}

if e_{term} is not constrained **then**

if midsize edge of t_i is constrained **and** the longest-edge bisection of t_i produces a triangle with smallest angle $< 30^\circ$ **then**

P is the midpoint of constrained midsize edge of t_i

else

 Perform the same verification on triangle t_j

end if

end if

 Perform constrained Delaunay insertion of P into τ_{ref}

 Update S_{target}

end while

end if

end for

Rivara *et al.* [10] discussed the guarantees of termination for angle parameters greater than 22.4° . After the longest-edge bisection of a bad quality triangle, the algorithm must ensure that the triangles produced around the new point do not repeat the geometry of the refined triangle. A rare non convergence case could arise when $\theta \geq 22.4^\circ$ and the midsize edge of the terminal triangle is not constrained. This scenario is avoided by bisecting the midsize edge instead, the same way it is done when the midsize edge is constrained (See Figure 3).

Rivara and Calderon [12] introduce a variant of the LEPP-Delaunay algorithm which inserts the *centroid* of the pair of terminal triangles instead of the midpoint of the terminal edge. They showed that, in practice, this strategy generates triangulations with significantly less points than those produced by the original (midpoint) version. Moreover the non convergence cases associated to midsize (constrained and non constrained) edges are completely avoided.

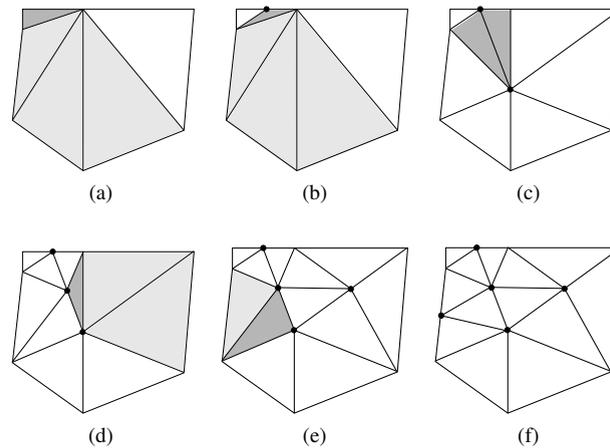


Fig. 2: The refinement process of the LEPP-Delaunay algorithm. Tolerance parameter $\theta = 30^\circ$ is the minimum angle allowed in the triangulation. Target triangles are shaded in dark gray. LEPPs are shaded in light gray. (a) Input triangulation. (b) - (e) Snapshots of each step of the process. (f) Final triangulation. In (b) the midpoint of a constrained midsize edge was inserted

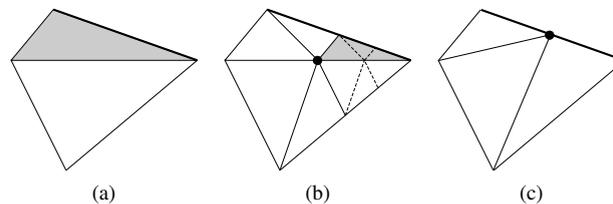


Fig. 3: (a) Terminal triangle with constrained midsize edge (bold). Bad quality triangles are shaded. (b) In this case, the insertion of the midpoint of the longest edge would reproduce the geometry, creating an infinite processing cycle. (c) The insertion of the midpoint of the constrained midsize edge successfully eliminates bad quality triangles without reproducing the geometry.

Using geometrical arguments, previous studies showed that the algorithm produces constrained Delaunay triangulation of at least 30° if non-constrained interior angles are included in the input geometry [10]. Note that for non-constrained terminal triangles, the insertion of the midpoint of the terminal edge is equivalent to a longest-edge bisection, followed by a sequence of local edge-flip operations needed to maintain the Delaunay property. In this sense, the algorithm takes advantage of the properties of LEPP-Bisection algorithm, recently discussed in [13].

From a practical point of view, the algorithm presents the following important advantages:

- (a) The LEPP strategy makes the algorithm independent of the order in which the triangles are processed. This simplifies the implementation of the algorithm and makes it easily parallelizable.
- (b) Selecting the midpoint of an existing edge is a *robust* operation and does not require additional strategies or computations to find the triangles containing the point being inserted.

2.1. Delaunay Terminal Triangles

Terminal triangles in a Delaunay triangulation satisfy the following geometric property [10]:

Proposition 1. For any Delaunay triangulation τ_{ref} , it holds that every pair of Delaunay terminal triangles in τ_{ref} have largest angles less than or equal to 120° .

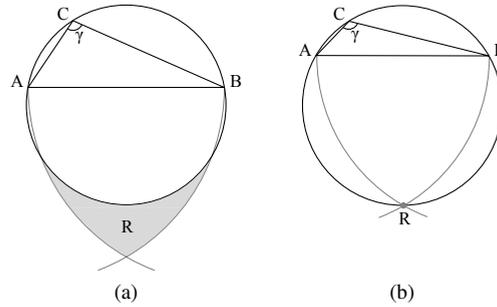


Fig. 4: (a) Region R defines the area for point E of longest-edge neighbor triangle ABE . (b) Region R becomes a point when $\gamma = 120^\circ$ (and triangle ABE is equilateral).

For any pair of Delaunay terminal triangles ABC and ABE sharing longest edge AB , vertex E of triangle ABE must lie outside the circumcircle of triangle ABC . Let R be the region for point E where AB is the longest edge of ABE . Region R is reduced to a point when the largest angle of ABC is 120° . For a complete proof see [10]. Figure 4 illustrates this property.

Note that triangles with angles greater than 120° are never eliminated by inserting the midpoint of their longest edge. These triangles are indirectly eliminated by the insertion of a point over other pair of terminal triangles $(t_i, t_j) \in \text{LEPP}(t)$. Whenever the midpoint of the terminal edge lies inside the circumcircle of t , the edge-flip operations performed to maintain the Delaunay property eliminate t and locally improve the angle distribution in the triangulation.

3. Analysis of the Insertion Radius

We call the *insertion radius* of a point p , denoted r_p , the distance from p to the nearest point of the triangulation. After the insertion of p , r_p becomes the length of the shortest edge adjoining p in the triangulation. The intrinsic bound on the largest angles of terminal triangles defines a bound on the insertion radius, which in turn can be used to guarantee that new points cannot be inserted arbitrarily close to each other, and correspondingly, that new edges cannot be arbitrarily small.

3.1. Bounds Based on the Circumradius

Lemma 1. For any Delaunay terminal triangle $t(ABC)$ with circumradius r , the distances from the midpoint D of its longest edge AB to any vertex of t is greater than or equal to $r/2$. Furthermore:

- $d(D, A) = d(D, B) \geq r\sqrt{3}/2$, and
- $d(D, C) \geq \begin{cases} r/2 & \text{if } t \text{ is obtuse} \\ r & \text{if } t \text{ is acute} \end{cases}$

Proof. Consider triangle $t(ABC)$ with $|AB| \geq |BC| \geq |CA|$, and angles $\gamma = \angle ACB$ and $\alpha = \angle ABC$ respectively the largest and smallest angles of t . Let r be the circumradius of t , and point D be the midpoint of longest edge AB .

Since $|AB| = 2r \sin \gamma$, it follows that $d(D, A) = d(D, B) = r \sin \gamma$, which has the smallest possible value of $r\sqrt{3}/2$ either when $\gamma = 120^\circ$ or $\gamma = 60^\circ$.

Distance to point C will depend on the geometry of the triangle. If t is an obtuse triangle, $d(D, C)$ is lower bounded by $r(1 + \cos \gamma)$, the distance from D to the closest point in the circumcircle. In this case $d(D, C)$ has the smallest possible value of $r/2$ when $\gamma = 120^\circ$. On the other hand, if t is an acute triangle, vertex C lies farther away from D than from the circumcenter, then $d(D, C)$ approaches r either when $\gamma \approx 90^\circ$ or $\alpha \approx 0^\circ$.

□

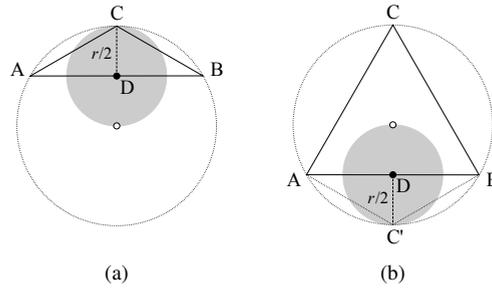


Fig. 5: Lower bounds for the insertion radius of point D , midpoint of longest edge AB of a triangle $t(ABC)$. Shaded ball of radius $r/2$ is fully contained in the circumcircle of t . (a) Largest angle $\gamma = 120^\circ$ and smallest angle $\alpha = 30^\circ$. (b) $\alpha = \gamma = 60^\circ$.

Lemma 1 shows that the distance from the midpoint of a terminal edge to the vertices of the terminal triangle can be bounded by the circumradius. The following theorem translates this result into a bound on the insertion radius.

Theorem 1. *For any Delaunay terminal triangle t it holds that the insertion radius of point D , the midpoint of t 's longest edge, is at least $r/2$, where r is the circumradius of t . Furthermore,*

$$r_D \geq \begin{cases} r(1 + \cos \gamma) & \text{if } t \text{ is an obtuse triangle} \\ r & \text{if } t \text{ is a right triangle} \\ r(1 - \cos \gamma) & \text{if } t \text{ is an acute triangle} \end{cases}$$

where γ is the largest angle of t .

Proof. Because of the Delaunay property, t has an empty circumcircle. Therefore, the distance from the closest point of the triangulation to D is bounded by the distance from D to the circumcircle of t :

- If t is an obtuse triangle, then $r_D > d_{\min}(D, C)$, where $d_{\min}(D, C) = r(1 + \cos \gamma)$ is the shortest distance between D and vertex C (Lemma 1).
- If t is a right triangle, then $r_D = r$ since D corresponds the circumcenter of t (Lemma 1).
- If t is an acute triangle, r_D is equivalent to the insertion radius of an obtuse terminal triangle sharing the circumcircle of t . Let $t'(AC'B)$ be this triangle, D is the midpoint of its longest edge AB , and $\gamma' = 180^\circ - \gamma$ is its largest angle; see Figure 5(b). Then $r_D > d_{\min}(D, C')$, where $d_{\min}(D, C') = r(1 + \cos \gamma') = r(1 - \cos \gamma)$.

Consider the pessimistic case where D is located as close as possible to the circumcircle; see Figure 5. Triangle t corresponds either to the obtuse triangle with $\gamma = 120^\circ$ and $\alpha = 30^\circ$ (Figure 5(a)), or to the equilateral triangle (Figure 5(b)). In both scenarios $r(1 + \cos \gamma)$ has the smallest possible value, $r/2$.

Let δ_D be the circle of radius r_D centered at point D . Then, no other point of the triangulation could lie inside δ_D because δ_D is contained by the empty circumcircle of t . □

3.2. Bounds Based on the Shortest Edge

Lemma 2. *For any Delaunay terminal triangle t with shortest edge of length l , the distance from the midpoint of its longest edge to any vertex of t is greater than or equal to $l/2$.*

Proof. Consider terminal triangle $t(ABC)$ with $|AB| \geq |BC| \geq |CA|$, $l = |CA|$, γ and α respectively the largest and smallest angles of t , and point D the midpoint of terminal edge AB .

The proof is based on the diagram for similarity regions proposed by Gutierrez *et al.* [11], later expanded by Bedregal and Rivara [14]; see Figure 6. Candidate Delaunay terminal triangle t has vertex C above arc C_5 , corresponding to

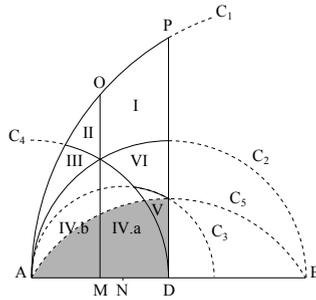


Fig. 6: Diagram for triangle $t(ABC)$. Vertex C lies in one of the regions defining a triangle t with $|AB| \geq |BC| \geq |CA|$. Shaded region corresponds to triangles with $\gamma > 120^\circ$.

triangles with $\gamma \leq 120^\circ$. We consider triangles such that $d(D, C) \leq |CA|$ or $d(D, A) \leq |CA|$. Consider $c_1 > 0$ the ratio between l and the distance from D to the closest vertex of t . The possible values of c_1 are discussed below.

- $d(D, C) \leq |CA|$: Corresponding to triangles with vertex C lying to the right of line OM of Figure 6. Constant c_1 decreases as C moves away from the intersection of arc C_3 and line PD . At the same time, constant c_1 decreases as C approaches line OM , acquiring the lowest value $c_1 = 1$ when C actually lies over OM . In this case $d(D, C) = |CA|$ regardless of the triangles geometry. On the other hand, c_1 increases as vertex C approaches PD , acquiring the highest values for isosceles triangles where C lies over PD . Furthermore:
 - If t is an acute triangle (region above arc C_2), then c_1 is maximized when t corresponds to the isosceles triangle with $\gamma \approx 90^\circ$. Therefore $c_1 < \sqrt{2}$.
 - If t is obtuse (region below arc C_2), then c_1 is maximized when t corresponds to the isosceles triangle with $\gamma = 120^\circ$, and $d(D, C) = |CA|/2$. Therefore $c_1 \leq 2$.
- $d(D, A) \leq |CA|$: Corresponding to triangles with vertex C lying above arc C_4 of Figure 6. Constant decreases as C approaches arc C_4 , acquiring the lowest value $c_1 = 1$ when C actually lies over C_4 . In this case $d(D, A) = |CA|$ regardless of the triangles geometry. On the other hand, c_1 increases as vertex C moves away from arc C_4 :
 - If t is an acute triangle, then the equilateral triangle maximizes the value of c_1 , and $d(D, A) = |CA|/2$. In this scenario $c_1 \leq 2$.
 - If t is an obtuse triangles, then the isosceles triangle with $\gamma \approx 90^\circ$ maximizes the value of c_1 , and $d(D, A) = |CA|/\sqrt{2}$. In this scenario $c_1 \leq \sqrt{2}$.

Since $c_1 \leq 2$, it follows that the distance from D to the closest vertex of t is at least $l/2$. □

Lemma 2 shows that, after introducing the midpoint of a terminal edge, the length of the edges adjoining the new point can be bounded by the length of the shortest edge of the terminal triangle. The following theorem translates this result into a bound on the insertion radius.

Theorem 2. For any Delaunay terminal triangle t with shortest edge of length l , it holds that the insertion radius of the midpoint of its longest edge is at least $l/2$.

Proof. The proof follows the same logic of the proof for Theorem 1. Consider the pessimistic case where D is located as close as possible to t 's circumcircle (e.g., when $\gamma = 120^\circ$ and $\alpha = 30^\circ$). Then, $l = 2r \sin 30^\circ = r$, $d(D, C) = 1 + \cos 120^\circ = r/2$, and $r_D = l/2$. Due to the Delaunay property, any other point of the triangulation must lie outside the circumcircle of t . Then $r_D > l/2$. □

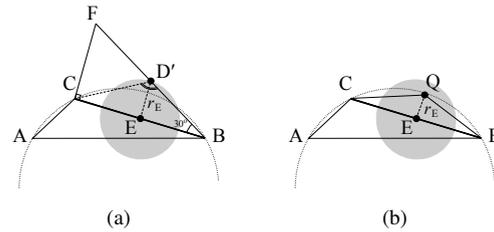


Fig. 7: Lower bounds for the insertion radius of point E , midpoint of constrained midsize edge BC of a triangle $t(ABC)$. Shaded ball has radius $\frac{d(E,B)}{\sqrt{3}}$. (a) After insertion of point D' triangle CBD' has largest angle $\angle BD'C = 120^\circ$ and smallest angle $\angle CBD' = 30^\circ$. (b) Input point Q is the closest point to E therefore $r_E = d(E, Q)$.

3.3. Bounds for Constrained Midsize Edge

When the algorithm selects the midpoint of the constrained midsize edge of a terminal triangle, the insertion radius of the new point will be defined by the length of the midsize edge. Among the vertices of the terminal triangle, the endpoints of the midsize edge become the closest points to the point selected.

Lemma 3. *Let $t(ABC)$ be a Delaunay terminal triangle with constrained midsize edge BC . Let E be the midpoint of this edge. Then, the insertion radius of E , r_E , is greater than $d(E, B)/\sqrt{3} = |BC|/2\sqrt{3}$.*

Proof. Consider that no point from the input lies too close to constrained edge BC . If at some point before the insertion of E the midsize-edge neighbor of t was refined, say t' , the algorithm inserted the midpoint of the longest edge of t' , say D' . As observed in Figure 7(a), due to the requirement of 30° to insert the midpoint of longest edge instead of the midpoint of the constrained midsize edge, the smallest triangle that the algorithm could produce is the obtuse triangle with largest angle of 120° and smallest angle of 30° . After a few calculations, distance $d(E, D')$ is equal to $\frac{|BC|}{2\sqrt{3}}$ which in turn is equal to $\frac{d(E,B)}{\sqrt{3}}$, defining a lower bound on the insertion radius of E . \square

It must be noted that, if an input point Q lies within distance $\frac{d(E,B)}{\sqrt{3}}$ from point E (Figure 7(b)), or point Q is the midpoint of a constrained edge $e \neq BC$ previously inserted by the algorithm (and Q lies within distance $\frac{d(E,B)}{\sqrt{3}}$ from E), then Q is the point of the triangulation closest to E . In both scenarios $r_E = d(E, Q)$, as stated below.

Lemma 4. *Let $t(ABC)$ be a Delaunay terminal triangle and E be the midpoint of its constrained midsize edge BC . If there is a point Q within distance $d(E, B)/\sqrt{3}$ from E , then Q is either an input point or Q is the midpoint of another constrained edge. The insertion radius of point E is $d(E, Q)$.*

The following theorem synthesizes the bounds on the insertion radius. We define a bound on the insertion radius of the midpoint of a constrained midsize edge of a terminal triangle in terms of the circumradius of the triangle and in terms of the smallest edge of the triangle.

Theorem 3. *For any Delaunay terminal triangle t the insertion radius of the midpoint of its constrained midsize edge is at least $r/2\sqrt{3}$, where r is the circumradius of t ; or at least $l/2\sqrt{3}$, where l is the length of the shortest edge of t .*

Proof. The bound is based on the worst-case scenario where t is the obtuse triangle with largest angle $\gamma = 120^\circ$ and smallest angle $\alpha = 30^\circ$, as illustrated in Figure 8.

Consider triangle $t(ABC)$ with constrained midsize edge BC , shortest edge CA and point E the midpoint of BC . From Lemma 3 we know that $r_E > |BC|/2\sqrt{3}$, which applies to the general case of refinement of terminal triangles. Since $|BC|$ is lower bounded either by $r/2$ or by $|CA|/2$, it follows that $r_E > \frac{r}{2\sqrt{3}}$, or that $r_E > \frac{l}{2\sqrt{3}}$. \square

Theorem 3 holds when there are no input points or points over a constrained edge lying within the bound stated above, otherwise the insertion radius is defined by the distance to this point (Lemma 4).

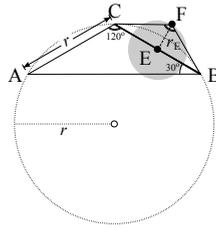


Fig. 8: Lower bounds for the insertion radius of point E , midpoint of constrained midsize edge BC of a terminal triangle $t(ABC)$ with largest angle $\gamma = 120^\circ$ and smallest angle $\alpha = 30^\circ$. Midsize edge neighbor triangle, CBF , is similar to t . The shaded ball has a radius of $\frac{r}{2\sqrt{3}}$.

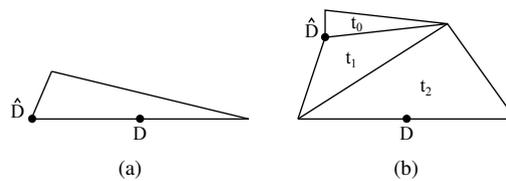


Fig. 9: Point D and its parent point \hat{D} . (a) When the triangle is both a terminal triangle and the current target triangle processed. (b) Triangle t_2 is the terminal triangle in $LEPP(t_0)$.

4. Proof of Termination

A geometric argument of the algorithm’s termination is based on the facts that the algorithm improves the triangles as they are processed, and that the algorithm constrains how close together two points can lie and how short an edge can be. Points inserted by the algorithm cannot introduce edges arbitrarily short, so the algorithm cannot run indefinitely creating ever smaller triangles.

The key idea behind the termination of the algorithm is that newly inserted points have an insertion radius lower bounded by insertion radius of existing points of the triangulation. We call the *parent* of a point D , denoted \hat{D} , the vertex of the target triangle u responsible for the selection of D (i.e., the first triangle in the propagation path), shared by the longest edge and shortest edges of u (Figure 9).

We assume that the algorithm processes the target triangles in set S_{target} in a specific order, selecting at each step the triangle in S_{target} with the smallest edge l_{min} . The idea of a *processing order* is used to facilitate the analysis of the algorithm. In practice, a processing order is not necessary since the algorithm is order-independent.

Lemma 5. Consider a tolerance parameter θ , point Q the midpoint of a terminal edge, and point P the parent of Q . Then $r_Q \geq \lambda r_P$, where λ is a constant determined by θ .

Proof. Since Q is the midpoint of a terminal edge, the ratio between the insertion radius of Q and that of its parent is maximized for the target terminal triangle with largest angle $\gamma = 120^\circ$. For a terminal triangle with smallest angle $\alpha = \theta$ the ratio between the insertion radius of Q and P is bounded by $\frac{1+\cos 120^\circ}{2\sin \theta}$, which corresponds to the ratio between the smallest edge of the target terminal triangle and the distance from the midpoint of its longest edge to the circumcircle.

For any tolerance parameter $\theta \leq \text{deg } 14.48$, it holds that $\lambda > 1$, and the insertion radius of new points is always greater than that of their parents. For $\theta > \text{deg } 14.48$, constant λ approaches $1/2$ as θ approaches 30° . \square

Lemma 5 establishes a theoretical limit on the rate the insertion radius evolves as new points are inserted. A configuration of triangles maximizing the decreasing rate of insertion radius (by half) can only happen for a tolerance parameter $\theta = 30^\circ$: the algorithm selects a bad quality terminal triangle with angle $\gamma = 120^\circ$ and angle $\alpha = 30^\circ$.

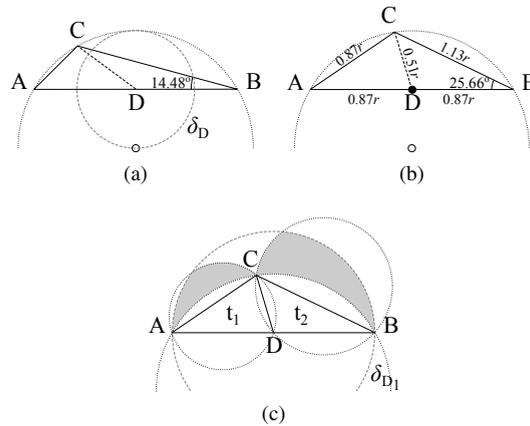


Fig. 10: Terminal triangle $t(ABC)$ with $\gamma = 120^\circ$. (a) $\alpha = 14.48^\circ$: Circle δ_D of radius $|CA|$ centered at point D is empty of any point of the triangulation. (b) $\alpha = 25.66^\circ$: Relationship between edges and circumradius r . (c) Shaded areas define the regions of potential points that could produce edges smaller than CA through edge-flip operations. δ_{D_1} is the circle of radius $|CA|$ centered at D . Dotted circles represent the circumcircles of triangles t_1 and t_2 .

For any triangle with smallest angle $\alpha < 14.48^\circ$ it is guaranteed that the insertion radius of any point inserted by the algorithm is greater than the insertion radius of its parent. If we guarantee that the insertion radius cannot decrease during refinement process, we can prove that the algorithm terminates. The following lemma, adapted from [4], formalizes this idea.

Lemma 6. Consider l_{\min} the shortest distance between two points of the input PSLG \mathcal{P} , and no two segments of \mathcal{P} meet an angle less than 30° . Given an angle tolerance parameter $\theta < 14.48^\circ$, the LEPP-Delaunay algorithm terminates and the output triangulation has no edge shorter than l_{\min} .

Proof. Consider the obtuse terminal triangle $t(ABC)$ with largest angle $\gamma = 120^\circ$ and smallest angle $\alpha = \arcsin(1/4) \approx 14.48^\circ$. Let t be the target triangle in S_{target} with smallest angle, then $l_{\min} = CA = r/2$. Since the minimum distance from D to the circumcircle of t is $r/2$, then any edge adjoining D will be necessarily greater than CA . Consider circle δ_D of radius $|CA|$ centered at point D , as shown in Figure 10(a). δ_D is fully contained in the circumcircle of t . Moreover, due to the Delaunay property no point of the triangulation lies inside δ_D . \square

Although the tolerance parameter proposed in Lemma 6 ensures an output refined triangulation without smaller edges, such a low angle bound limit the improvement properties of the algorithm. For example, an angle bound $\theta = 25.66^\circ$ (with $\lambda = \frac{1}{\sqrt{3}}$) ensures that edges AD and DB are at least as long as the shortest edge CA . Moreover, there is no pessimistic scenario where the algorithm continuously decreases the insertion radius of new points. The following lemma presents the relaxed proof for the algorithm's termination.

Theorem 4. Consider l_{\min} the shortest distance between two points of the input PSLG \mathcal{P} , and no two segments of \mathcal{P} meet an angle less than 30° . Given an angle tolerance parameter $\theta < 25.66^\circ$, the LEPP-Delaunay algorithm terminates and the output constrained Delaunay triangulation τ_{ref} has no edge shorter than λl_{\min} , with $\lambda > \frac{1}{\sqrt{3}}$.

Proof. The generation of smaller edges happens when terminal triangle t has largest angle γ close to 120° and smallest angle α close to 25.66° . Since the minimum distance from D to the circumcircle of t is $r/2$, then any edge adjoining D will be necessarily greater than $(\frac{r/2}{2r \sin 25.66^\circ})|CA| = |CA|/\sqrt{3}$. Furthermore, in this configuration the algorithm creates edge DC which is smaller CA ($|DC| > 0.59|CA|$), as observed in Figure 10(b). Additional edges smaller than CA (but longer than DC) could be created only through edge-flip operations if a point lies inside the region illustrated in Figure 10(c). This region is defined by the intersection of circle δ_{D_1} (of radius $d(A, C)$ and centered at point D) and the circumcircles of triangles $t_1(ADC)$ and $t_2(BCD)$. Note that any edge-flip operation performed over triangles outside δ_{D_1} can only produce edges longer than CA .



In practice, the algorithm is observed to terminate for tolerance parameters of up to 37° [9,10].

5. The LEPP-Delaunay Algorithm in Practice

A comparison of the LEPP-Delaunay algorithm with the Triangle software [5] is included in [12]. The LEPP-Delaunay algorithm produces triangulations of quality analogous to those obtained with the Triangle software without requiring the use of complex point location strategies nor priority queues for target triangles. An evaluation of the algorithm's performance in three-dimensional scenarios was discussed in [15].

Figure 11 shows triangulations obtained by the LEPP-Delaunay algorithm for a few input geometries. Figure 12 shows the behavior of both the original LEPP-Delaunay midpoint algorithm and the LEPP-Delaunay centroid algorithm [12] for different geometries and angle qualities. We compared the performance with the current version of the open-source Triangle software (which is optimized to use also off-centers). The LEPP-Delaunay centroid algorithm shows a better performance than its midpoint counterpart, and comparable to that of the Triangle software (note that neither midpoint nor centroid variants of the LEPP-Delaunay algorithm uses priority queues).

6. Discussion and Future Work

We have proved that the midpoint selection strategy does not introduce points too close to existing points nor does it create edges arbitrarily small. For a tolerance parameter $\theta = 25.66^\circ$, we show that, in most cases, the algorithm produces edges that are longer than l_{\min} , the smallest edge in the initial constrained Delaunay triangulation. Moreover, in the scenario in which the algorithm creates an edge smaller than l_{\min} , then: (1) the length of the new smallest edge is at least $l_{\min} / \sqrt{3}$; (2) only quality triangles will be associated with the new smallest edge; and (3) every edge introduced by the algorithm will be longer than the new smallest edge. So far, this guarantees good performance and termination for tolerance parameters of up to 25.66° (this same methodology could be adapted to provide proofs for $\theta = 30^\circ$).

Using the proper data structures to allow constant-time access to neighborhood information and to a triangle's LEPP, an implementation of the LEPP-Delaunay algorithm consistently takes $O(n \log n + N)$ time in practice, where n is the number of points in the initial PSLG and $N \geq n$ is the number of points in the output triangulation. The term $n \log n$ covers the cost associated with the construction of the initial triangulation, while the term N covers the cost of refinement, since points are inserted in constant time.

The methodology introduced in this paper can be used in the analysis of LEPP-Delaunay centroid and future variants of the algorithm. The next step in our research is to extend this methodology to establish the first theoretical proofs of the LEPP-Delaunay algorithm in three-dimensional refinement. In practice, the algorithm has shown good performance [15] and it maintains the advantages of the two-dimensional version, such as independence of the processing order of triangles or the assurance that new points lie in the interior of the geometry.

The bounds on insertion radius presented in this paper can be fed into Ruppert's methodology for the analysis of Delaunay algorithms [2] (later improved by Shewchuk [4]) to produce the theoretical guarantees on the LEPP-Delaunay algorithm's good grading and size-optimality based on the *local feature size* of points. Some of this work has been recently discussed in [16].

Acknowledgements The authors would like to thank the Department of Computer Science at the University of Chile and CONICYT Chile for their support. The authors would also like to thank Pedro Rodriguez for his contributions in obtaining the data used in Section 5.

References

- [1] J. Ruppert, A new and simple algorithm for quality 2-dimensional mesh generation, in: Proceedings of the Fourth Annual ACM/SIGACT-SIAM Symposium on Discrete Algorithms, 1993, pp. 83–92.
- [2] J. Ruppert, A Delaunay refinement algorithm for quality 2-dimensional mesh generation, Journal of Algorithms 18 (1995) 548 – 585.

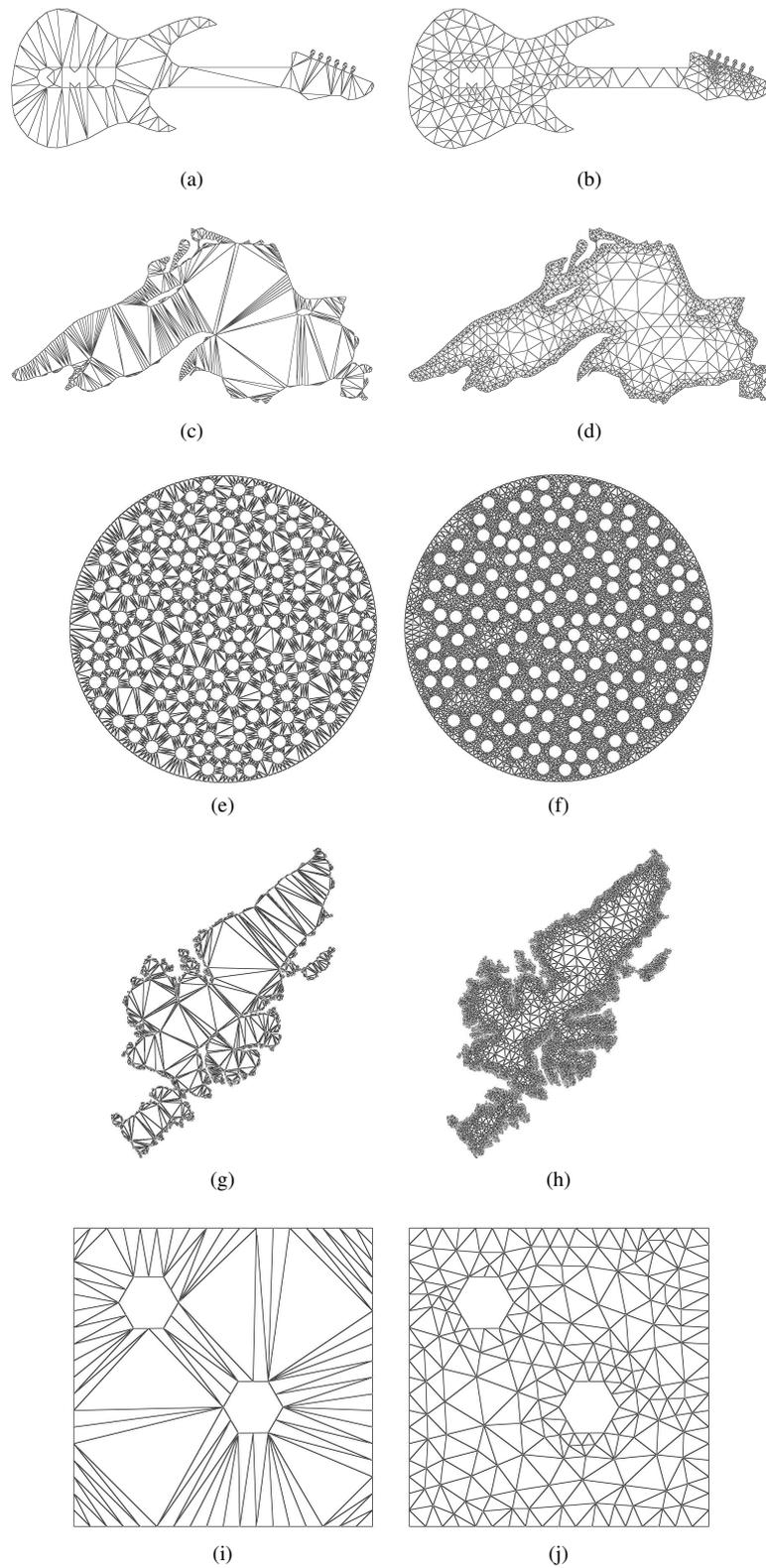


Fig. 11: (left) Input geometries. (right) Refined triangulations with $\theta = 30^\circ$.

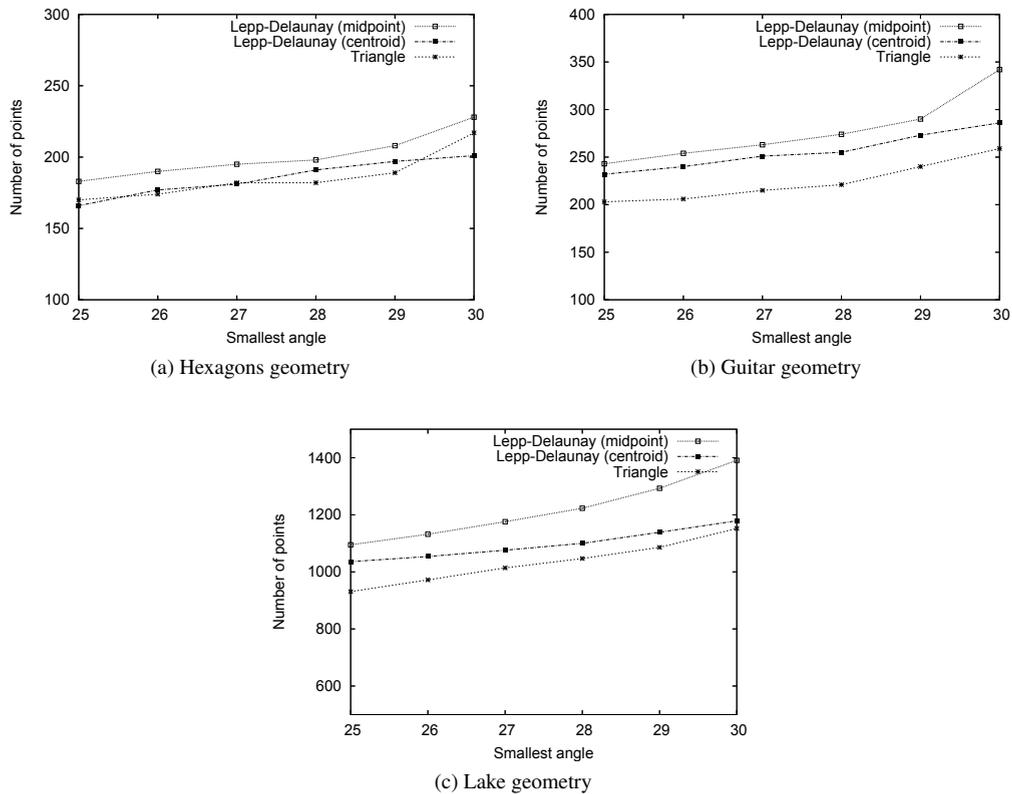


Fig. 12: Comparison of mesh size (number of points) obtained by LEPP-Delaunay algorithm using the midpoint and the centroid point selection strategies and Triangle. Angle tolerance parameters between 25° and 30° . (a) Hexagons geometry (Figure 11(i)). (b) Guitar geometry (Figure 11(a)). (c) Lake geometry (Figure 11(c)).

- [3] L. P. Chew, Guaranteed-quality mesh generation for curved surfaces, in: Proc. of the Ninth Annual Symposium on Computational Geometry, 1993, pp. 274–280.
- [4] J. R. Shewchuk, Delaunay refinement algorithms for triangular mesh generation, Computational Geometry 22 (2002) 21 – 74. 16th ACM Symposium on Computational Geometry.
- [5] J. R. Shewchuk, Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator, in: M. C. Lin, D. Manocha (Eds.), Applied Computational Geometry: Towards Geometric Engineering, volume 1148 of *Lecture Notes in Computer Science*, Springer-Verlag, 1996, pp. 203–222.
- [6] H. Erten, A. Üngör, Quality triangulations with locally optimal steiner points, SIAM Journal on Scientific Computing 31 (2009) 2103–2130.
- [7] P. A. Foteinos, A. N. Chernikov, N. Chrisochoides, Fully generalized two-dimensional constrained delaunay mesh refinement, SIAM Journal on Scientific Computing 32 (2010) 2659–2686.
- [8] S.-W. Cheng, T. K. Dey, J. R. Shewchuk, Delaunay Mesh Generation, Chapman and Hall / CRC computer and information science series, CRC Press, 2013.
- [9] M.-C. Rivara, New longest-edge algorithms for the refinement and/or improvement of unstructured triangulations, Int. Journal for Numerical Methods in Engineering 40 (1997) 3313–3324.
- [10] M.-C. Rivara, N. Hitschfeld, B. Simpson, Terminal-edges Delaunay (small-angle based) algorithm for the quality triangulation problem, Computer-Aided Design 33 (2001) 263 – 277.
- [11] C. Gutierrez, F. Gutierrez, M.-C. Rivara, Complexity of the bisection method, Theoretical Computer Science 382 (2007) 131–138.
- [12] M.-C. Rivara, C. Calderon, Lepp terminal centroid method for quality triangulation, Computer-Aided Design 42 (2010) 58–66.
- [13] C. Bedregal, M.-C. Rivara, Longest-edge algorithms for size-optimal refinement of triangulations, Computer-Aided Design 46 (2014) 246–251.
- [14] C. Bedregal, M.-C. Rivara, A study on size-optimal longest edge refinement algorithms, in: X. Jiao, J.-C. Weill (Eds.), Proc. of the 21st Int. Meshing Roundtable, Springer Berlin Heidelberg, 2013, pp. 121–136.
- [15] M.-C. Rivara, M. Palma, New LEPP algorithms for quality polygon and volume triangulation: implementation issues and practical behavior, Trends in Unstructured Mesh Generation AMD 220 (1997) 1–8.
- [16] C. Bedregal, Analysis of Longest Edge Algorithms for 2-Dimensional Mesh Refinement, Technical Report, Department of Computer Science, University of Chile, 2015.