Reordering Ruppert's Algorithm

Alexander Rand

Fall Workshop on Computational Geometry, 10/31-11/1, 2008.

1 Overview

Ruppert's algorithm [6] is an elegant method for generating size-competitive meshes, but admits a poor worst case run-time. Recent time-efficient Delaunay refinement algorithms [2] rely on bounding the the degree of each intermediate triangulation and thus ensure that all local operations in the Delaunay triangulation are efficient. We propose a simple alternative to Ruppert's algorithm which maintains this additional property that the all intermediate triangulations have bounded degree.

The algorithm combines three main ideas. First, the yielding procedure of Ruppert's algorithm is eliminated by instead deleting a nearby circumcenter whenever a midpoint or input point is inserted in the mesh in the spirit of Chew's second algorithm [1]. Second, quality of the mesh is maintained before conformity in a similar fashion to the SVR algorithm [2]. Finally, the triangles on the priority queue are prioritized by circumradius with the largest simplices processed first.

The resulting algorithm produces a conforming Delaunay, size-competitive quality mesh. Additionally, there is an explicit bound on the degree of each triangulation produced by the algorithm which depends only on the minimum angle acceptable for output triangles, denoted κ. The algorithm is simple and can be implemented by making a few changes to Ruppert's algorithm.

2 Preliminaries

Given a non-acute piecewise linear complex (PLC) $\mathcal{C} = (\mathcal{P}, \mathcal{S})$ composed of sets of points and segments, we seek a refinement $\mathcal{C}' = (\mathcal{P}', \mathcal{S}')$ which conforms to the input and contains no triangles with angles less than κ°. The algorithm incrementally builds a refinement for this purpose.

Definition 1. Let $q \in \mathcal{P}'$ be a vertex added to the mesh.

- n_q is a **nearest neighbor** to q in the triangulation.
- r_q is the **insertion radius** of q: the distance from q to n_q when q is inserted.
- q is called a **quality point** if q was inserted as the circumcenter of a poor quality triangle.
- q is called a **conformality point** if q is an input point or was inserted as the midpoint of a segment.

Definition 2. Given a PLC $\mathcal{C}$, the **local feature size** of a point p, denoted $\text{lfs}(p, \mathcal{C})$ or $\text{lfs}(p)$, is the radius of the smallest disk centered at p that intersects two disjoint features of $\mathcal{C}$.

Local feature size will always be evaluated with respect to the input PLC $\mathcal{C}$, and the second argument will be omitted.

Definition 3. A simplex is **unacceptable** if it is

- an input point which has not been inserted,
- a segment with a nonempty diametral disk, or
- a triangle with an angle less than κ°.

3 Algorithm

The following algorithm is very similar to Ruppert's algorithm: it maintains a queue of unacceptable simplices and processes the front simplex by inserting its circumcenter. The main difference from Ruppert's algorithm is that quality points do not yield to segments they encroach, but when conformality points are inserted, a nearby circumcenter is removed (if it exists).

Algorithm 1 Reordered Ruppert

```
Initialize the Delaunay triangulation of a bounding box.
Form a priority queue of all unacceptable simplices.
while the queue is nonempty do
    Insert the circumcenter q of the top simplex s.
    if q is a conformality point then
        if n_q is a quality point then
            Remove n_q.
        end if
    end if
    Update the priority queue of unacceptable simplices.
end while
```

When processing queued simplices, triangles are given the highest priority, followed by input points and finally segments. Triangles are prioritized by circumradius with larger triangles processed first.

4 Results

First, the algorithm is shown to produce meshes with the same desirable properties as Ruppert's algorithm: the resulting Delaunay triangulation conforms to the input PLC, and the output is graded to the local feature size. These first two theorems mirror the standard results for Ruppert's algorithm.

κ	Ruppert	Algorithm 1
5°	266	266
10°	284	297
15°	312	366
20°	352	477
25°	473	647

Table 1: Number of points in the resulting example meshes for different minimum angles.

Theorem 1. *The triangulation produced by Algorithm 1 conforms to the input PLC and contains no angles less than κ.*

Theorem 2. *For $\kappa < \arcsin\left(\frac{1}{4\sqrt{2}}\right)$, Algorithm 1 terminates. Moreover, there exists $C_\kappa > 0$ such that for each vertex q inserted into the mesh,* $\text{lfs}(q) \leq C_\kappa\, r_q$.

Finally, the degree of each Delaunay triangulation is bounded throughout the duration of the algorithm. A similar result in [2] relies on ensuring no small angles occur at any point during the algorithm and used this quality bound to imply the degree bound. While Algorithm 1 allows arbitrarily small angles to occur in intermediate Delaunay triangulations, it is still possible to compute and explicitly bound the degree of these triangulations.

Theorem 3. *There exists $D(\kappa)$ depending only on κ such that the degree of the Delaunay triangulation at any step during Algorithm 1 is bounded by $D(\kappa)$.*

The key idea in this proof is that whenever a point is inserted into the mesh, any new triangle formed has no angles larger than $180 - \kappa$ degrees. So, while the algorithm allows arbitrarily small angles to occur in the triangulation, it does not allow angles which are near 180°.

Only Theorem 3 relies on the specific order of the triangles in the priority queue in the algorithm: the other results hold as long as triangles are processed before segments. Algorithm 1 does not lead to a degree bound if the smallest triangles are processed first. A bound on the value of $D(\kappa)$ in Theorem 3 can be computed explicitly. For $\kappa = 10.2^\circ$, this bound is 248.

Figure 1 gives an example of a mesh refined using Algorithm 1 and Ruppert's algorithm for several different κ values. Table 1 contains the number of vertices in the resulting meshes.

5 Extensions

This algorithm can be extended to higher dimensions with a slight modification. Simplices queued for mesh quality are processed at a higher priority than those for mesh conformity. As in the 3D extension of Ruppert's algorithm [3], insertions for conformity are prioritized by dimension with the lowest dimension handled first. As in the SVR algorithm [2], insertions for quality are prioritized by dimension with the highest dimension handled first.

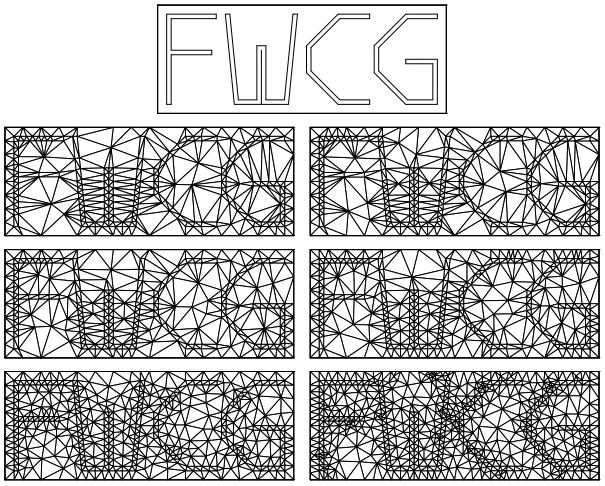

Figure 1: (top) An input PLC. (left) Output of Ruppert's algorithm using $\kappa = 10^\circ$, 15°, and 25°. (right) Output of Algorithm 1 using $\kappa = 10^\circ$, 15°, and 25°. While Theorem 2 only ensures that Algorithm 1 terminates for κ less than 10.2° (compared to 20.7° for Ruppert's algorithm), Algorithm 1 terminates in practice for higher κ values.

This modification also enlarges the allowable range for the minimum angle parameter to $\kappa < \arcsin(\frac{1}{4})$. Ruppert's algorithm still provides a wider range for this minimum angle parameter, accepting any $\kappa < \arcsin(\frac{1}{2\sqrt{2}})$.

We expect that the degree bound on intermediate triangulations will also hold if the largest first priority queue on triangles is replaced with a first in-first out queue. In this case, it is likely that the explicit degree bound on the triangulations is weaker.

We hope to integrate this approach with techniques for applying Delaunay refinement to domains with acute angles [4, 5].

References

[1] L. P. Chew. Guaranteed-quality mesh generation for curved surfaces. In *Symposium on Computational Geometry*, 1993.

[2] B. Hudson, G. Miller, and T. Phillips. Sparse Voronoi refinement. In *15th International Meshing Roundtable*, 2006.

[3] G. Miller, S. Pav, and N. Walkington. Fully incremental 3D Delaunay refinement mesh generation. In *11th International Meshing Roundtable*, 2002.

[4] G. Miller, S. Pav, and N. Walkington. When and why Ruppert's algorithm works. In *12th International Meshing Roundtable*, 2003.

[5] A. Rand and N. Walkington. 3D Delaunay refinement of sharp domains without a local feature size oracle. In *17th International Meshing Roundtable*, 2008.

[6] J. Ruppert. A Delaunay refinement algorithm for quality 2-dimensional mesh generation. *J. Algorithms*, 18(3):548–585, 1995.