Probing for Surface Mesh Generation through Delaunay Refinement

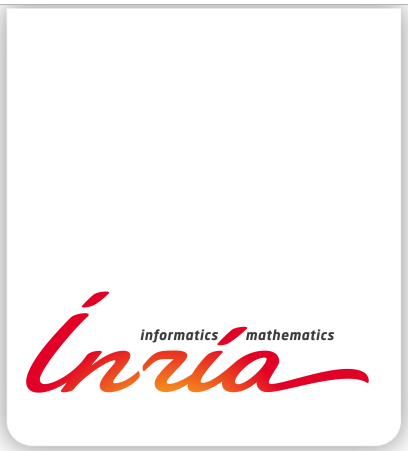
Hugo Férée, Pierre Alliez

Probing for Surface Mesh Generation through Delaunay Refinement

Hugo Férée & Pierre Alliez

RESEARCH REPORT
N° 8123
July 2009
Project-Team GEOMETRICA

ISSN 0249-6399 ISRN INRIA/RR--8123--FR+ENG



Probing for Surface Mesh Generation through Delaunay Refinement

Hugo Férée* & Pierre Alliez†

Project-Team GEOMETRICA

Research Report n° 8123 — July 2009 — 17 pages

Abstract: Surface mesh generation through Delaunay refinement is considered as a relevant alternative to the common marching cubes algorithm. One distinctive feature lies into the fact that it interleaves shape probing (through intersection with Voronoi edges) with refinement. The current implementations require seeding the refinement procedure for each connected component so as to guarantee that all components are properly discovered by the mesh refinement procedure. This task is often left to the user. Although this is an easy task for input polyhedral surfaces, it is not when the input surface is defined as an isovalue of an implicit function. In this report we propose an automatic seeding procedure which interleaves refinement, seeding and initialization steps with the qualities of the refinement and the guarantees of careful seeding.

Key-words: Delaunay refinement, surface probing, surface approximation, surface sampling

* hugo.feree@inria.fr
† pierre.alliez@inria.fr

RESEARCH CENTRE
SOPHIA ANTIPOLIS – MÉDITERRANÉE

2004 route des Lucioles - BP 93
06902 Sophia Antipolis Cedex

Sondage de surfaces pour la generation de maillages par raffinement de Delaunay

Résumé : La génération de maillage de surfaces par raffinement de Delaunay fournit une alternative pertinente à l'algorithme des «marching cubes». Son implantation actuelle nécessite une initialisation de la part de l'utilisateur, pour garantir que toutes les composantes connexes seront maillées lors de la procédure. Selon la représentation de la surface, cette étape peut être plus ou moins difficile. Nous proposons dans ce rapport une procédure automatique d'initialisation qui garantisse la complétude et la précision du maillage final.

Mots-clés : Raffinement de Delaunay, sondage de surfaces, approximation de surfaces, echantillonnage de surfaces

1 Introduction

We consider the issue of meshing through Delaunay refinement an input surface known only through a (virtual) probing device, i.e., an oracle providing the existence (and if so, intersection points) of intersections between the surface and a Voronoi edge. Although powerful on many aspects, such knowledge is in fact not sufficient to mesh the whole surface as the Delaunay refinement procedure requires to be properly seeded. More specifically, the user is requested to specify a bounded domain of interest and as many seed facets as the number of connected components.

Although the user-specified domain is often a reasonable thing to ask, there is often no prior knowledge about the number and location of the connected components in the practical applications. One way to tackle this issue is to resort to the same oracle through random initial queries, but this may have several drawbacks: the input shape can have arbitrarily small connected components which cannot be all detected in finite time. This is why we also ask for a strict lower bound of the minimum local feature size of the surface (the reach), which can also be seen as a precision parameter (we can ask for all the connected components greater than a given parameter to be meshed). Even with such reach parameter guaranteeing to seed all connected components would require systematic probing, which would generate overly complex meshes.

1.1 Previous Work

The well known marching cubes algorithm achieves the goal of meshing the whole surface with topological guarantees and no self intersection. The final mesh vertices are constructed as the intersections of the input surface with the edges of a fine regular grid which covers the whole domain (each edge of the grid may be seen as a short probe). As a result the output mesh is often overly complex (see figure 1 for a comparison with Delaunay refinement described later). In addition, as the grid is independent from the input surface, the mesh is most likely not adapted to it (albeit improvements exist [4]) and contains many badly-shaped elements.

A more data-adaptive method uses the Delaunay refinement based on Chew's algorithm [3] and further developed by Boissonnat and Oudot [2]. Based upon the notions of Delaunay refinement and filtering, it provides quality meshes which are intersection free by construction, have only well-shaped triangles (with large minimal angle), and approximate well the input surface in terms of topology and geometry (area, Hausdorff distance and normals). More specifically, given an input approximation parameter ε less than 0.091 times the reach of S, the mesh is a 2ε sample of the surface S, the Hausdorff distance between them is at most $4.5 diam(S)\varepsilon^2/(reach(S)$ and the area and normals are approximated with a bounded error $\mathcal{O}(\varepsilon)$).

The algorithm proceeds in refining the mesh until all user-specified criteria are matched (which eventually happens), by inserting the intersections of the surface with the Voronoi edges dual to the bad Delaunay facets. The edges of the Voronoi diagram are the probes which are used to both discover the surface and to refine it. The added value compared to the marching cubes algorithm comes from the fact that such probes are longer (hence more effective) and more adapted to the input surface as they get more and more orthogonal to it as we refine. This refinement procedure being based on the farthest-point paradigm, Steiner vertices are added where they are the most needed. It has been shown that the final mesh complexity is small (asymptotically within a constant factor of the optimum). One limitation comes from the fact that it is only a refinement algorithm hence requires to be seeded by generating at least one bad facet on each connected component of the input surface.

Such mesh generation algorithm has been extended [1] so as to require only one point per connected component. This method is restricted to non nested components since they aim at

meshing the surface with a probing device moving in the free space outside the surface (which can be seen as and applied for a robot with sensors discovering the space around him). The set of probes (divided in actual and former Voronoi edges and additional probes) is then connected and included in the free space. But because the Voronoi diagram changes, and that the insertion of a new vertex can disconnect this path, the additional probes has to prevent this and to be stored. The problem considered in this work is more general (e.g. the connected components can be nested) and has less restrictions (we are not restricted to probe from the outside of the surface, like a robot).

1.2 Contribution

We propose an automatic way to seed a surface mesh generator for compact smooth closed 2-manifold surfaces with arbitrary genus and number of connected components given:

- an intersection oracle
- a convex compact domain Ω including the surface (e.g. a bounding box or sphere)
- a positive lower bound ε of the reach of the surface (defined later)

This allows us to define a generic algorithm, independent from the form of the type surface, i.e. an implicit surface, a grey level in a $3D$ image, a parametric surface, a point cloud, etc. Contrary to the previous method, we do not ask for an initial point set since it is often difficult to provide and sometimes even the number of connected components is unknown. Moreover, we allow the connected components to be nested and use the Delaunay refinement algorithm to provide an adaptive and so parsimonious mesh contrary to the marching cubes method, and also provide the same guarantees of shape and approximation cited above. To do so, we use the Voronoi diagram and additional probes (like in [1]) but without storing the latter. We only require to store a subset of the Voronoi cells (or equivalently the vertices of the triangulation) in a priority queue.

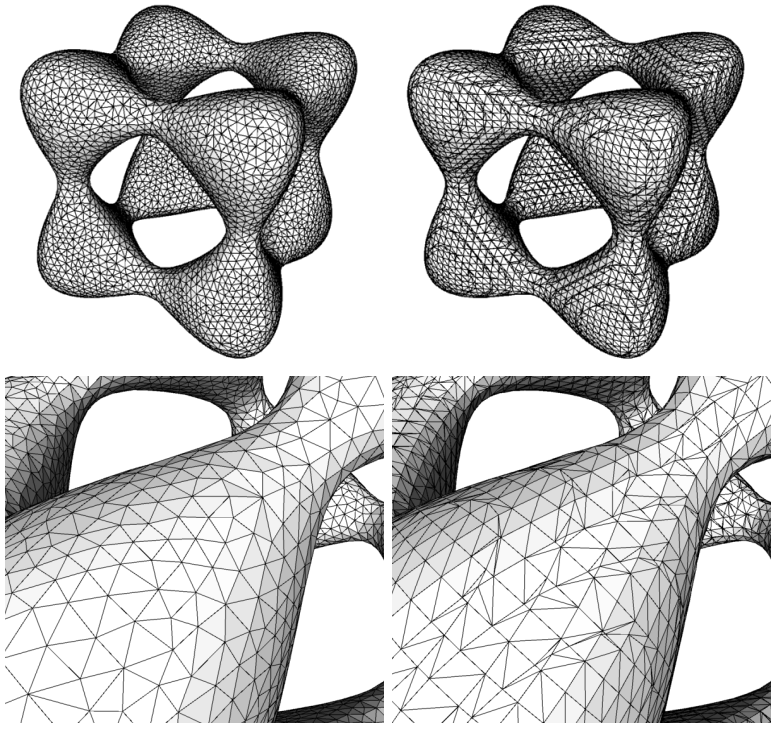


Figure 1: A comparion between the meshes obtained by the Delaunay refinement algorithm (left: 7,040 vertices)and the marching cubes method (right: 10,440 vertices). Notice the anisotropy and the variable shape of the triangles in the second case.

2 Definitions

All these definition are valid in dimension 3 (for our specific purpose), but are also worth or can be adapted for other dimensions.

Definition 1 (Smooth 2-manifold surface) *A smooth 2-manifold surface of* $\mathbb{R}^3$ *is a* $C^{1,1}$ *surface locally homeomorphic to a disc at each point.*

Definition 2 (Local feature size) *The local feature size (noted lfs) is the distance to the medial axis, where the medial axis of a surface* S *is the set of points with at least two closets points on* S *(i.e. the centers of the open balls disjoint from* S *and tangent to* S *in at least two points).*

Definition 3 (Reach) *We call reach the minimum local feature size on the surface. It is positive for a smooth surface and we note* ε *a positive strict lower bound of the reach in the rest of this paper. It basically gathers the distance between two connected components, the curvature and the thickness of the surface.*

Definition 4 (Voronoi diagram) *The Voronoi cell of a point* X_i *among a finite set of points* $E = (X_j)_{j=1..n}$ *is the set of points whose closest point in* $(X_j)_{j=1..n}$ *is* X_i*:*

$$V(X_i) = \{X,\ \forall j \neq i,\ d(X, X_i) \leq d(X, X_j)\}$$

where d is (in our case) the Euclidean norm. If the E is in general position (i.e. no four points are coplanar and no five points are cospherical), then a Voronoi cell is a non empty convex polyhedron, the intersection of two adjacent cells is a 2D convex polygon, the intersection of three cells is a line segment (finite or infinite), and the intersection of four cells is a point. This cellular complex forms the Voronoi diagram of E, noted $\mathcal{V}(E)$.

Definition 5 (Delaunay triangulation) *The Delaunay triangulation of a set of points is the triangulation (i.e. simplicial complex) dual to the corresponding Voronoi diagram. A Voronoi cell is dual to its generating point, a Voronoi facet is dual to a segment, a Voronoi edge is dual to a triangle, and a Voronoi point is dual to a tetrahedron.*

Definition 6 (Restricted Delaunay) *The Delaunay triangulation of a point set restricted to a surface is the sub-complex of the Delaunay triangulation defined by the facets whose dual Voronoi edge intersects the surface.*

Definition 7 (Surface Delaunay ball) *The surface Delaunay ball of a facet of the Delaunay triangulation restricted to a surface S is the ball centered on the intersection of S with the edge dual of the facet and circumscribed to the latter. The interior of a surface Delaunay ball does not contain any point of the mesh.*

Definition 8 (Bad facet) *A bad facet of a restricted Delaunay triangulation in a Delaunay refinement is a facet which does not match the size (i.e. the radius of its surface Delaunay ball is too long), shape (i.e. an angle is too small) or approximation criterion (the distance between the center of its surface Delaunay ball and the center of its circumcenter is too large).*

We also define the notion of loose μ-sample introduced in [2]:

Definition 9 (Loose μ-sample) *A set of points E is a loose μ-sample of a surface S if:*

- $\forall x \in S \cap \mathcal{V}(E), E \cap \mathcal{B}(x, \mu \times lfs(x)) \neq \emptyset$ *In other words, the radius of all the surface Delaunay balls is less than* μ
- *The Delaunay triangulation of E restricted to S has vertices on all connected components of S*

3 Experimental study

The Delaunay refinement builds the Voronoi diagram and the dual Delaunay triangulation of a set of points on the surface, and then the Delaunay triangulation restricted to the surface. These structures are updated throughout the algorithm.

Then, the bad facets are refined (i.e. the intersection of the surface with their dual edge is inserted) until there is no more of them.

It has been shown in [2] that for a small enough size criterion (a fraction of the reach), the Delaunay refinement provides a mesh isotopic to the input surface, with the requested criteria and with other good properties like a bounded error approximation and variation of the normals to the mesh.

These results only hold for a sufficiently good initialization of the mesh, i.e. a bad facet for each connected component is required.

It is always possible to initialize at least one connected component through random queries since the domain is bounded, and some other components can be meshed during the course of the algorithm which probes the domain by querying the Voronoi edges (see figure 2 for an example).

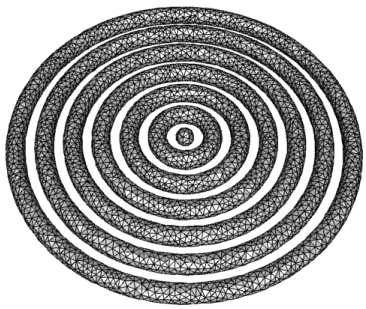


Figure 2: The mesh of 6 toruses of minor radius 1 around a sphere of radius 1, and at distance 1 from each other, resulting from the Delaunay refinement algorithm by only initializing four points on the sphere. The toruses are not initialized but meshed during the refinement (with the following criteria: minimum angle = 30 degrees, maximum size = approximation error = 0.5)

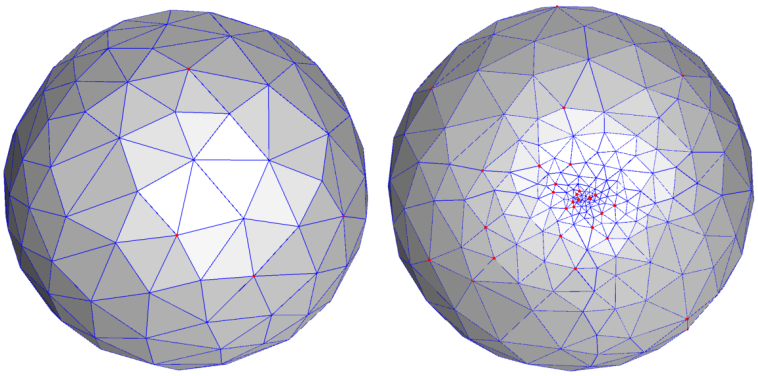


Figure 4: The same sphere meshed with the same criteria, but with a different number of initial points (initialized with the same method as in figures 3 and 5). The left one has 5 initial points and 171 vertices in the refined mesh, and the right one has 50 initial points and 303 final points. Notice that in the second case there are very close initial points which make very small triangles, so much more points are added to adapt the mesh (since the size of the triangles cannot vary too quickly).

Figure 3 show statistics made on the current mesher of CGAL [5] with random surfaces (union of non intersecting spheres) with a constant number of initial points. The method used to initialized (random segments from the center of the bounding sphere to its boundary) is the method currently used in the surface mesher of CGAL when no initial point set is provided.

Notice that even with two spheres and 50 random initial points, we are not sure that both will be meshed and that the ratio of meshed components does not increase very fast with respect to the number of initial points.

A simple solution would be to initialize the mesh with queries on a fine grid like a marching cubes, to guarantee that every connected component will be meshed. But this often adds much more point than necessary and can create very small or bad shaped triangles which will be costly to refine. Indeed, if two points are very close, their corresponding triangles will be very small

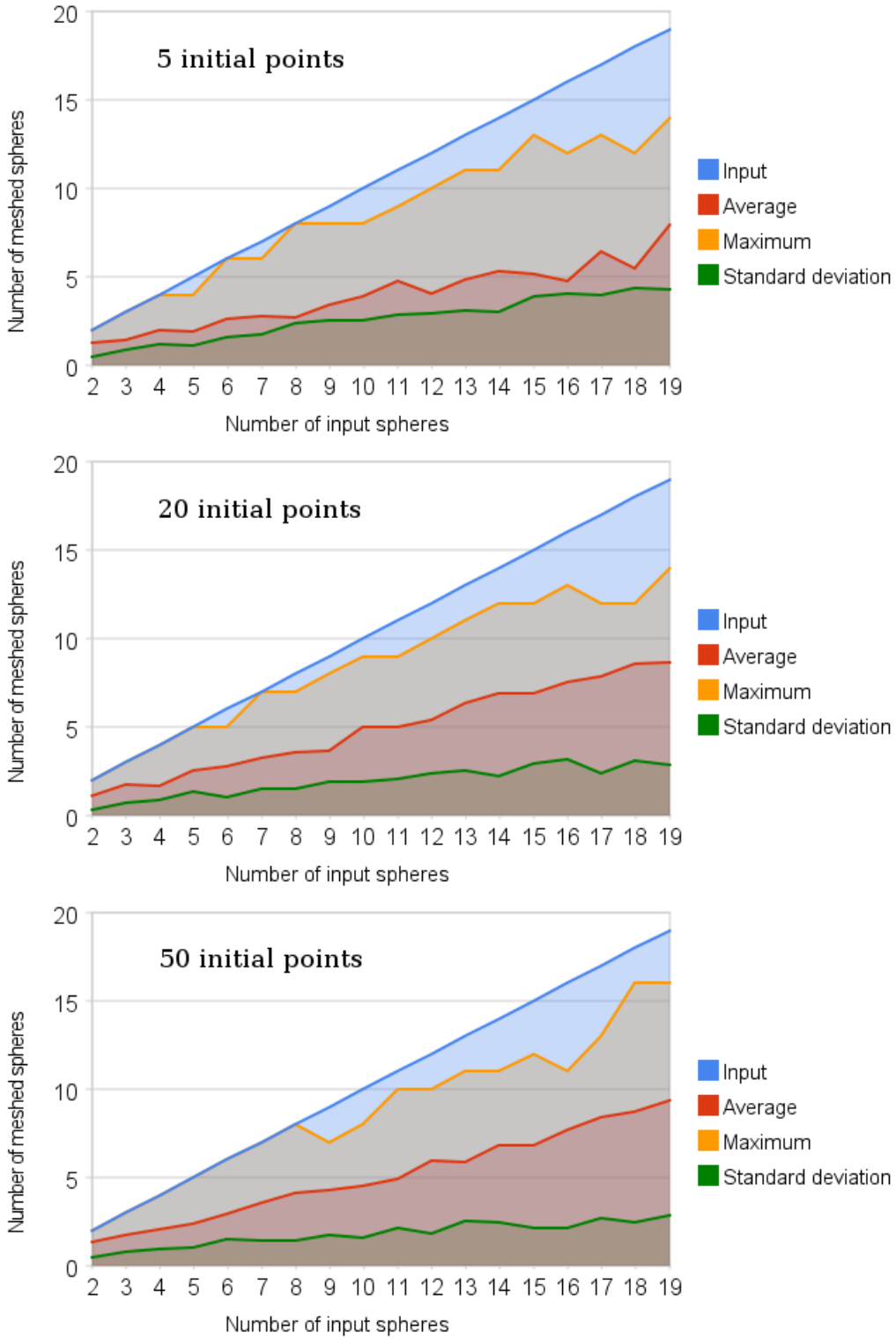


Figure 3: Statistics (50 tests for each point) made on random unions of non intersecting spheres (possibly nested) with respectively 5, 20 and 50 initial points. The latter are found by intersecting the surface with radial segments (radial for the bounding sphere $\partial\Omega$) with uniformly random direction. The spheres are of radius at least 1 and are at distance at least 1 form one another (so $\varepsilon < 1$) and included in a bounding sphere of radius 100.

too, and many points will be inserted around them to correct this since the size of the triangles cannot vary briskly. See figure 4 for an example.

At first thought, we can believe that with sufficiently strong criteria (thus with a sufficiently dense sampling of the initialized components), the whole surface can be meshed. But the following result shows that for some cases, non initialized components will never be meshed.

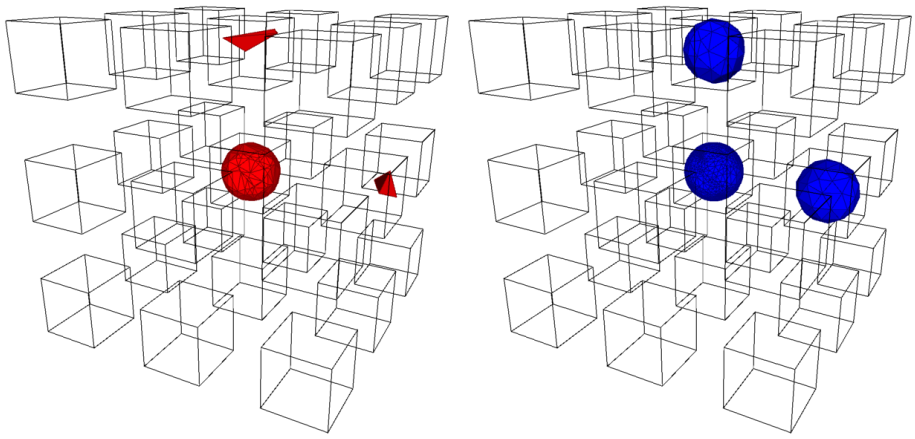


Figure 5: An attempt to mesh a regular grid of spheres (represented by their bounding cubes) using the Delaunay refinement algorithm implemented in CGAL, with 200 random initial points. The mesh after refinement (right) only includes the spheres correctly initialized (left).

Proposition 1 *If S_0 is a connected component of diameter d, whose volume contains no other component, at distance at least d of any other connected component and containing at most two points of the triangulation, then S_0 will not be meshed by Delaunay refinement (i.e. it will never contain more than two points).*

Proof

Let's assume that during one step of the algorithm S_0 contains two points A and B of the mesh and that an edge e of the Voronoi diagram intersects S_0 at a point P (this is the only way to insert a point). By duality, since A and B are the closest point from P (because the volume delimited by S_0 contains no other component and because of the distance hypothesis), P (as each point of e) is equidistant from A, B and another point C necessarily on another component. This provides the contradiction

$$\|\overrightarrow{CP}\| = \|\overrightarrow{AP}\| \leq d < \|\overrightarrow{CP}\|$$

and ends the proof. □

This basically means that to be meshed, a non initialized connected component must at least be closer to its neighbours than its own size. Else, it is included in the cell or the union of the cells of its (one or two) points.

Then, there exists simple examples, like a regular grid of size $3r$ of spheres of radius r (see figure 5) where only well initialized connected components are meshed. So there can exist points in the triangulation that are not part of the mesh (we name them isolated in the rest of the paper).

Given that the Voronoi edges of the refined mesh define a set of queries that have already been done (the edges dual to a face of the restricted Delaunay intersect the surface, and the distance to the intersection point is bounded by the approximation criterion, and the other edges do not intersect the surface), we can use them to query the whole domain more smartly than purely random queries, and more sparingly than a with a regular grid which could not reuse these probes.

4 Algorithm

It only needs (in addition of the data structures already defined during the refinement) a priority queue storing the Voronoi cells that have not been probed yet, sorted with respect to their size.

Algorithm Meshing algorithm

Require:

- An surface S defined by an oracle and bounded by a domain Ω
- A size criterion $< 0.019\varepsilon$
- approximation and size criteria

Ensure:

- A mesh of S with same topology and the requested approximation and shape qualities

$Q \leftarrow \emptyset$
$P_0 =$ `probe(`Ω`)`
`initialize_component(`P_0`)`
while $Q \neq \emptyset$ **do**
  `refine`
  if $\exists P$ isolated **then**
    `initialize_component(`P`)`
  else
    repeat
      $C =$`pop`Q
      `probe(`C`)`
    until `intersection_found` or $Q = \emptyset$
  end if
end while

We use the refinement algorithm with the constraint of an output two manifold without boundary mesh, with a size criterion inferior to 0.091ε and with any shape and approximation criteria.

The `probe(`C`)` function tries to find intersection points in the Voronoi cell C (intersected with Ω) dual to a point P. It is first used at the beginning of the algorithm to find the first point. Else,it is called only after a refinement step which produced no isolated point.

The intersections closer from the mesh than ε are ignored. Indeed, for sufficiently strong refinement criteria, the Hausdorff distance between the mesh and the surface (restricted to the connected component of P) is less than ε, and thus, any point on another connected component is at least at distance ε of the mesh. Moreover, since the component of P has already been correctly meshed, adding another point on it could prompt a useless and unwanted refinement of this component for it could make bad shaped triangles and hence add a large number of vertices (see figure 4 for an illustration).

To do so, we probe the cell such that any point in the cell it at distance at most ε of a probed segment. Moreover, the set of probes must be connected. This will be useful later to prove that no component was missed.

In practise (but there may be some other ways), we include the cell in a bounding box as thin as possible to minimise the number of probes (since we prefer probing few, long segments).

The probes are made successively (until an intersection is found by the oracle) along the lengthiest direction of the cuboid, and clipped on the faces of the cell (or on the boundary of the domain if the cell is infinite). The number of such requests is about $\mathcal{A}/(2\varepsilon^2)$, where $\mathcal{A}$ is the area of the smallest face of the cuboid (basically the thickness of the cell). The order of probing is important to maximise the probability to intersect the surface during the first probes. Indeed, the edges bounding the cell are already probed, so the first place to probe is the central axis of the cell, and more generally where the distance of the closest probe is maximal.

Then additional probes are done to make this set of segments connected and connect them to the Voronoi diagram (since it is itself connected). This can be done with about $\mathcal{O}(1/\varepsilon)$ requests.

If an intersection is found, it is inserted in the triangulation, its dual cell is added to the queue, and the adjacent cells are updated in the queue. This update consists in decreasing the size of the cells already in the queue (since the cells after insertion are included in the old ones). The other cells were already probed, so the new ones (by inclusion) are probed too. The refinement step must also be adapted the same way to keep Q up-to-date each time the Voronoi diagram is modified.

`initialize_component` allows to initialize the connected component of an isolated point (evoked in section 3) by adding two points close to it, such that the corresponding surface Delaunay ball is of radius at most $\varepsilon/2$ almost as it is done in [1]. This will be useful later to prove that this facet stays in the mesh. To do so, we find the second point by probing the edges of a small cube (of size less than ε to be sure to find an intersection) centered on the isolated point. The third one is found by probing the edges (of a square of about the same size) centred in the middle of the previous ones and included in their medial plane. This technique allows to control the minimum and maximum distance between theses points and to form an almost equilateral triangle.

4.1 Analysis

The following lemma and properties prove the correctness and the termination of the algorithm.

This proposition is adapted from theorem 4.4 of [2]:

Proposition 2 *Let E be a loose μ-sample of S, with $\mu \leq 0.091$. Then the Delaunay triangulation of E restricted to S is topologically equivalent to S.*

Proposition 3 *Before each call of `probe`, the mesh is topologically equivalent to the subset of the connected components of S which include at least one facet of the mesh.*

This means that proving the correctness of the algorithm comes to proving that the algorithm ends with at least one facet of the mesh on each connected component.

Proof
The `probe` procedure is only called when the mesh is completely refined. So there is no more bad facet, and the size criteria is fulfilled for each facet of the mesh. This means that each surface Delaunay ball $\mathcal{B}(x, r)$, verifies

$$r \leq 0.019\varepsilon < 0.019lfs(x).$$

This means that the mesh is a 0.019-sample of the set of components containing a restricted Delaunay facet. The result is then proved by applying proposition 2. □

We know need to define the notion of persistent facet and prove the following lemma adapted from [2]. Note that its authors worked with a positive 1-lipschitz function lesser than the local feature size instead of our a simple positive lower bound ε (which is in fact a particular case of the initial notion). Our work can be extended in the same way but for the sake of clarity (and for it is much simpler to find a constant lower bound than a functional lower bound) we did not present the more general case in this paper.

Definition 10 (Persistent facet) *A facet is called persistent when its surface Delaunay ball is of radius less than half the size criterion (i.e.* $0.091 = 0.0455$ *in our case).*

Lemma 1 *Every persistent facet stays in the mesh throughout the refinement algorithm.*

Lemma 1 and proposition 3 together show that since the `initialize_component` procedure creates a persistent facet, and that the corresponding component will be meshed at the beginning of the next turn of the while loop (during the `refine` step) will stay meshed during the rest of the algorithm since `initialize_component` and `probe` cannot break a persistent facet.

Indeed, the first one can only add points on a non meshed component and the second one prevents the insertion of points too close (closer than ε) from points of the mesh.

Proving the termination is now simple: `initialize_component` cannot be called on two points of the same component (since after the first call, the component is meshed and the second point cannot be isolated). Moreover, if `probe` inserts a point, its component contained none, as explained earlier.

So if the number of connected components is k, the algorithm can only pass k times through the first branch of the if statement (since the number of initialized components increases) and k times in the second one (since the number of components containing a point of the triangulation increases too). This proves the termination of the algorithm.

The next proposition expresses that no connected component was missed, and with the help of proposition 3 and since there is no more isolated point in the mesh, proves that the mesh is topologically equivalent to the surface and has the approximation properties given by the refinement algorithm.

Proposition 4 *Each connected component contains at least one point of the triangulation.*

Proof
If the algorithm ends, then the domain is covered with probed Voronoi cells, i.e. every point in the domain is at distance at most ε of the set of probes. Given that the volume limited by a connected component includes a ball of radius at least ε (by definition of ε), there exist a probed segment which intersects this ball. This defined a probed point inside the volume. There also

exists at an infinite Voronoi edge, which gives a point outside this volume when clipped on the boundary of the domain. The set of probes being connected and the surface being closed, there exists a probed segment which intersects the surface (on a path joining these two points).

If this intersection point was not added to the triangulation, then it was at distance less than ε of a face of the mesh, thus at distance at most 2ε of a meshed connected component. But by definition of ε, two connected components are at least at distance 2ε from each other, so this component is meshed and contains points of the triangulation. □

We can then show three representative cases:

- If the connected components are close from each other, like in figure 2, once a component is initialized, the other are meshed by the only refinement step. Then, all the cells are probed but no intersection point is found.
- If the components are not close enough to be meshed by only refinement, like in figure 5, in the worst case, all the components have successively an isolated point which is initialized immediately. The cells are then probed only at the end.
- If the components are very small and very spaced, a lot of probes will be done to find a first point, whose component will be meshed. Then, in the worst case, the other components are so small that they are not intersected by the Voronoi edges (so there are no isolated points) and they need to be find by the `probe` procedure.

Notice that since the cost of the refinement can be considered as necessary to mesh the surface correctly and that the *initialize_component* procedure is cheap and also necessary to initialize the components, the real cost of the algorithm lies in the analysis of the Voronoi cells.

This is why the choice of the bounding volume Ω can greatly influence the execution time. Informally, it is at most proportional to the volume of Ω (e.g. if Ω is a cube of size a, in the worst case, the Voronoi cells to analyse are $(a/\varepsilon)^3$ cubes of size ε and are all probed in constant time).

But our algorithm is adaptable if further information is known about the surface, for example if:

- some connected components are already initialized
- the number of connected components is known. Then we can stop the algorithm earlier instead of probing the whole space.
- a positive number r such that for any connected component, the volume it defines contains at least one ball of radius r. Then, each point in a probed cell must only be at distance r from the set of probes (rather than ε). See the proof of proposition 4 for an explanation. The maximum value of r is greater than ε by definition, and can be arbitrarily great compared to ε, so this knowledge could also prevent to search the whole space (see figure 6 for an illustration).

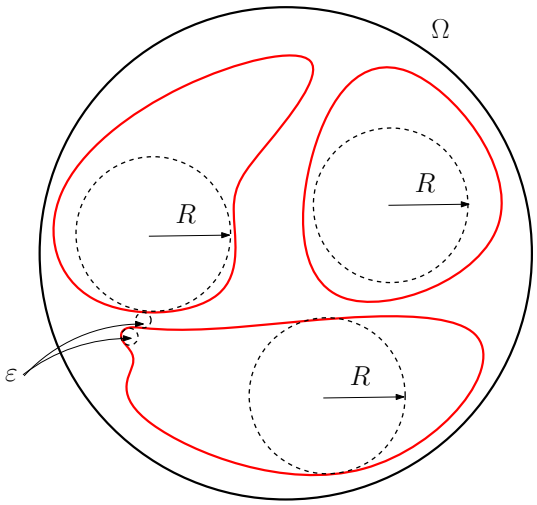


Figure 6: $2D$ illustration of the absence of correlation between the maximum radius r and the minimum local feature size ε. Once the three components are meshed, very few probes are necessary to guarantee that no component was missed. It would require much more queries to cover the domain with probes at distance at most ε of any point.

5 Experimental results

We have implemented a partial version of the algorithm including only the treatment of isolated points. The mesh is intitialized and refined while there exists isolated points. At a given step, one can initialized only one point or all the isolated points found. The second method is faster since it allows to initialize several components at once and so less refinement steps are done, but can initialize twice the same components since several isolated points can lie on the same component. The first method, described in the complete version of the algorithm, spares then the number of vertices, but in both cases, the same components are meshed in the end. This partial algorithm already provides good results, and only let the smallest components unmeshed.

Figure 7 shows statistics made with this implementation on the same surfaces as in figure 3. We notice that in average at least 80% of the components are meshed, in the best case all components are meshed (wich was not the case in figure 3 from a given number of spheres) and, according to the standard deviation and the smoothness of the curve, the results are less unpredictable.

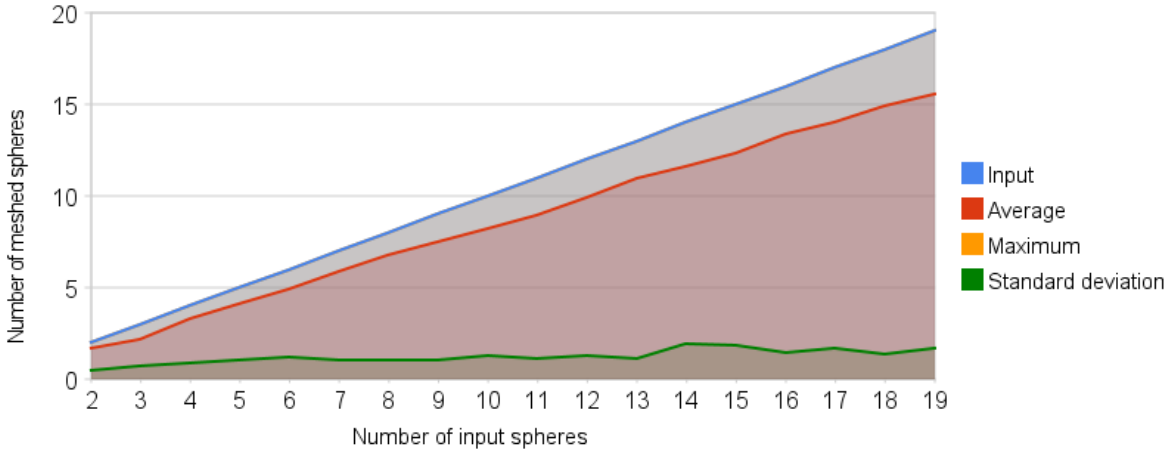


Figure 7: The same statistics as in figure 3, with the meshing algorithm dealing with isolated points.

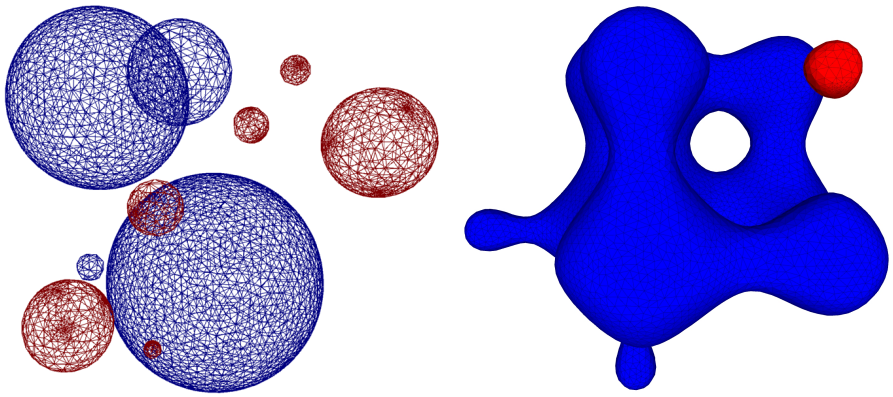


Figure 8: Left: 10 random spheres (as in figures 3 and 7), with in blue the spheres meshed with simple refinement and in red the additional spheres meshed with the previous extension dealing with isolated points. Right: the implicit surface defined by the equation $x^2+y^2+z^2+sin(4x)+sin(4y)+sin(4z)=0$. The red component verifies the hypothesis of proposition 1 and is only meshed after an isolated point is initialized.

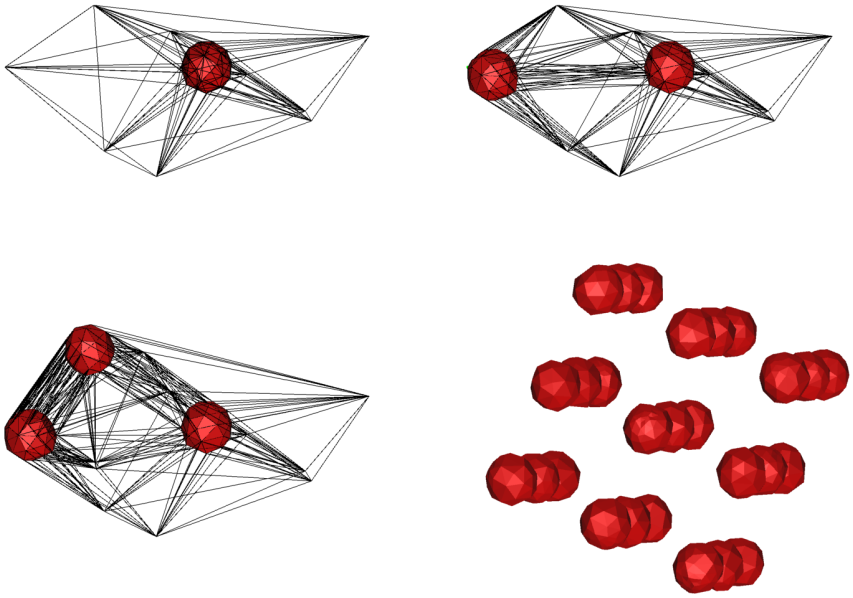


Figure 9: Example of mesh of a grid of spheres. First, one component is meshed, but many others are seeded. One isolated point is initialized, the meshed is refined, and this process is iterated until all the components are meshed. The mesh is shown in red, and the $3D$ triangulation in black (respectively after 0, 1, 2 and 26 iterations).

An example is given by figure 9 and some comparisons with the initial method (i.e. random initialization and one refinement step) are provided by figure 8. Notice the accretion of points on some red spheres due to the initialization of the component: to be sure that the facet created will be persistent, the points have to be close enough. This is the same problem as described in figure 4. We think of a remeshing step which could solve this by deleting or relocating the vertices. Indeed, in our special case, we know precisely where a remeshing is needed without looking at all the mesh. This remeshing step must especially preserve the topology and the approximation and shapes criteria. We have also thought of an other way to initialize a component which would first attempt to form a quite large triangle, and then refine and check if the component is meshed. If not, the previously described method is applied. This would prevent to do such cluster of vertices in most cases (persistent facets will then only be used to guarantee the mesh of the component).

6 Conclusion

We have designed an meshing algorithm which does not require any initial point set with the same topological, approximation and shape guarantees as the refinement algorithm. Dealing with isolated points is already a good improvement compared with refinement with random initial points, and an implementation of the spatial searching part would allow to guarantee the meshing of all the components.

References

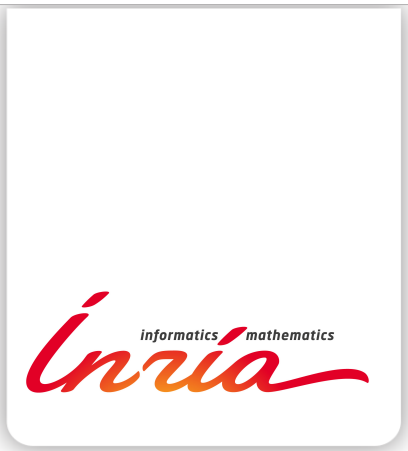
[1] J.D. Boissonnat, L.J. Guibas, and S. Oudot. Learning smooth shapes by probing. *Computational Geometry: Theory and Applications*, 37(1):38–58, 2007.

[2] J.D. Boissonnat and S. Oudot. Provably good sampling and meshing of surfaces. *Graphical Models*, 67(5):405–451, 2005.

[3] L.P. Chew. Guaranteed-quality mesh generation for curved surfaces. In *Proceedings of the ninth annual symposium on Computational geometry*, pages 274–280. ACM New York, NY, USA, 1993.

[4] Leif P. Kobbelt, Mario Botsch, Ulrich Schwanecke, and Hans-Peter Seidel. Feature sensitive surface extraction from volume data. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 57–66, New York, NY, USA, 2001. ACM.

[5] Laurent Rineau and Mariette Yvinec. 3d surface mesh generation. In CGAL Editorial Board, editor, *CGAL User and Reference Manual*. CGAL Project, 3.4 edition, 2008.



RESEARCH CENTRE
SOPHIA ANTIPOLIS – MÉDITERRANÉE

2004 route des Lucioles - BP 93
06902 Sophia Antipolis Cedex

Publisher
Inria
Domaine de Voluceau - Rocquencourt
BP 105 - 78153 Le Chesnay Cedex
inria.fr

ISSN 0249-6399