INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING
Int. J. Numer. Meth. Engng 2010; **00**:1–6 *Prepared using* nmeauth.cls *[Version: 2002/09/18 v2.02]*

A frontal Delaunay quad mesh generator using the L^∞ norm

J.-F. Remacle[1], F. Henrotte[1], T. Carrier-Baudouin[1], E. Béchet[2], E. Marchandise[1], C. Geuzaine[3] and T. Mouton[2]

[1] *Université catholique de Louvain, Institute of Mechanics, Materials and Civil Engineering (iMMC), Bâtiment Euler, Avenue Georges Lemaître 4, 1348 Louvain-la-Neuve, Belgium*
[2] *Université de Liège, LTAS, Liège, Belgium*
[3] *Université de Liège, Department of Electrical Engineering and Computer Science, Montefiore Institute B28, Grande Traverse 10, 4000 Liège, Belgium*

SUMMARY

In a recent paper [1], a new indirect method to generate all-quad meshes has been developed. It takes advantage of a well known algorithm of the graph theory, namely the Blossom algorithm, which computes in polynomial time the minimum cost perfect matching in a graph. In this paper, we describe a method that allow to build triangular meshes that are better suited for recombination into quadrangles. This is done by using the infinity norm to compute distances in the meshing process. The alignment of the elements in the frontal Delaunay procedure is controlled by a cross field defined on the domain. Meshes constructed this way have their points aligned with the cross field directions and their triangles are almost right everywhere. Then, recombination with the Blossom-based approach yields quadrilateral meshes of excellent quality.

KEY WORDS: quadrilateral meshing; surface remeshing; graph theory; optimization; perfect matching

1. Introduction

There exist so far essentially two approaches to generate automatically quadrilateral meshes. With *direct methods*, quadrilaterals are constructed at once, using either advancing front techniques [2] or regular grid-based methods (quadtrees) [3]. *Indirect methods*, on the other hand, rely on an initial triangular mesh and apply merging techniques to recombine the triangles of the initial mesh into quadrangles [4, 5]. Other more sophisticated indirect methods use a mix of advancing front and recombination [6].

Indirect methods have the advantage to rely on triangle meshing algorithms that are relatively simple to implement and that have some mathematical properties that allow to build fast and robust surface meshers. In a recent paper [1], we have shown that it is always possible to build a mesh made of quadrilaterals only starting from any triangular mesh that contains an even number of triangles. The Blossom-quad algorithm described in [1] allows to recombine optimally any triangular mesh in a very efficient manner. Yet, triangular meshers

*Correspondence to: jean-francois.remacle@uclouvain.be

Received
Revised

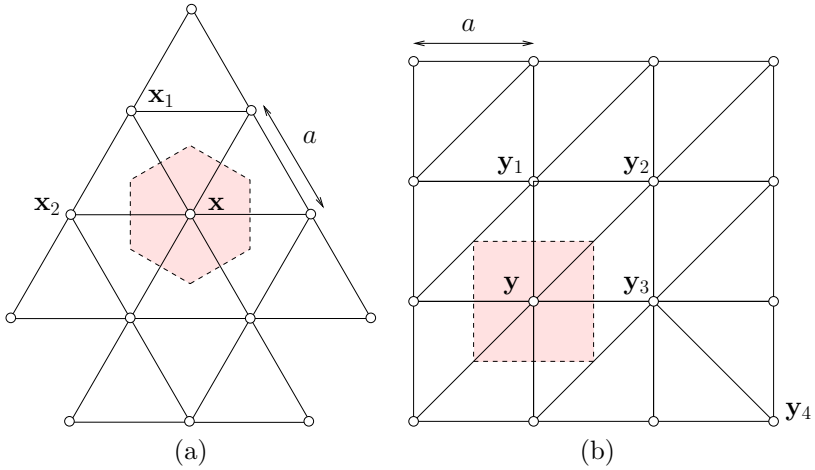


Figure 1. Voronoi cells of one vertex that belongs either to mesh of a equilateral triangles (a) or of right triangles (b).

aim at producing triangles that are close to equilateral and a set of equilateral triangles is not the optimal starting point of a indirect quadrilateralization algorithm.

In order to illustrate that problem, Figure 1-(a) shows a uniform triangular mesh in R^2 with equilateral elements; all elements and all edges are of size a. This mesh can be deemed perfect in the sense that optimality criteria, both in size and shape, are fulfilled. In this case, the Voronoi cell of each vertex $\mathbf{x}$ is an hexagon of area $a^2\sqrt{3}/2$, and the number of points per unit of surface is $2/(a^2\sqrt{3})$. Comparing with a uniform mesh made of right triangles of size a, Figure 1-(b), one sees that the Voronoi cells are now squares of area a^2. Filling R^2 with equilateral triangles requires thus $2/\sqrt{3}$ times more vertices (i.e. about 15% more) than filling the same space with right triangles. So, although quad meshes can be obtained by recombination of any triangular meshes, conventional triangular meshes are not the most appropriate starting point because they are essentially made of (nearly) equilateral triangles and contain therefore about 15% too many vertices.

This paper can be considered as the sequel of the Blossom-quad paper [1]. Its purpose is to introduce a method to generate triangular meshes suited for recombination into well-behaved quad meshes.

The mesh in Figure 1-(b) contains edges of different sizes. For example, $\|\mathbf{y} - \mathbf{y}_2\|_2 = a\sqrt{2}$ whereas $\|\mathbf{y} - \mathbf{y}_1\|_2 = a$. This mesh contains long edges at 45 degrees and short edges aligned with the axis. This explains why mesh (b) contains less points than mesh (a). Yet, long edges will be eliminated by the recombination procedure and the final mesh will be made of quadrilaterals with all edges of size a.

In devising a procedure to generate triangular meshes well-suited for recombination into quadrangles, one should recognize that the optimal size of an edge to be inserted at a point by the Delaunay algorithm depends on its orientation. A first possibility would be to encapsulate this directional information into some kind of anisotropic L^2 metric. This is however not

possible as such a metric $\mathcal{M}$ would have to ensure that (See Figure 1-(b))

$$(\mathbf{y}_1-\mathbf{y}_2)^T\mathcal{M}(\mathbf{y}_1-\mathbf{y}_2) = (\mathbf{y}_3-\mathbf{y}_2)^T\mathcal{M}(\mathbf{y}_3-\mathbf{y}_2) = (\mathbf{y}_3-\mathbf{y}_4)^T\mathcal{M}(\mathbf{y}_3-\mathbf{y}_4) = (\mathbf{y}-\mathbf{y}_2)^T\mathcal{M}(\mathbf{y}-\mathbf{y}_2),$$

which is clearly impossible. This suffices to conclude that standard metric-based triangulator are unable to produce meshes made exclusively of right triangles.

The approach proposed in this paper is based on the following observation. If distances between points are measured in the L^∞-norm, the triangular elements of Figure 1-(a) are no longer equilateral: $\|\mathbf{x}-\mathbf{x}_2\|_\infty = a$ and $\|\mathbf{x}-\mathbf{x}_1\|_\infty = a\sqrt{3}/2$. On the other hand, the elements of Figure 1-(b), which are right triangles in the L^2 norm, are equilateral in the L^∞-norm: $\|\mathbf{y}-\mathbf{y}_1\|_\infty = \|\mathbf{y}-\mathbf{y}_2\|_\infty = a$.

On this basis, a frontal Delaunay algorithm can be adapted to work in the L^∞-norm so as to produce triangular meshes with the right number of nodes and triangles suitably shaped for producing high quality quadrilaterals after recombination.

In their paper [17], Tchon and Camarero have already proposed a method for generating quad-dominant meshes based on the use of the L^∞-norm. In their approach, they start from a standard triangular mesh and apply vertex relocation that aim at aliging vertices on prescirbed directions. A very similar approach has been theorized and extended in 3D by Levy in its Lp Centroidal Voronoi Tessellation paper [7]. There are two advantages with our new approach :

1. Because they rely on the minimization of complex and stiff functionals, both smoothing approaches of Tchon and Levy are not CPU efficient (101 seconds for generating 972 quads [17] and 322 seconds for generating a surface mesh of about 1000 elements). Here, our approach allows to generate quads at the speed of triangle mesh generation, i.e. tens of thousands of elements a minute!

2. Both approaches of Tchon and Levy rely on existing triangulation that contain too much vertices. In [7], authors propose to remove vertices in the smoothing process in order to address that issue. In our new approach, we obtain directly the right number of points.

The paper is divided in two parts.

First, a new method to build the so-called "cross fields" [7] is presented. The cross fields represent at each point of the domain the preferred orientations of the quadrilateral mesh. In the finite element community, it is usually appreciated that quadrilateral elements have orientations parallel to the domain boundaries. With quadtree-based quadrilateral meshing techniques, for example, the mesh orientation is determined by the orientation of the initial quadrant of the quadtree. This leads to misoriented elements near the boundaries, which are rotated by about 45° with respect to the general quadtree orientation (see Figure 2). In the proposed approach, the mesh orientation will not be determined by means of a user-defined local reference frame or by advancing fronts in the mesh but by a cross field computed beforehand on a background mesh. The cross field is set parallel to the edges and boundaries of the domain and propagated smoothly across the bulk of the domain by means of a Laplace harmonic map.

In the second part, the mesh generation procedure is described in detail. A new frontal Delaunay approach inspired by [8] is proposed for determining the successive position of new points. Frontal meshers usually insert a point in the mesh so as to form an equilateral triangle (in L^2-norm). Here, we also aim at generating an equilateral triangle, yet in the sense of the local L^∞-norm aligned with the cross field. Meshes constructed this way have their points

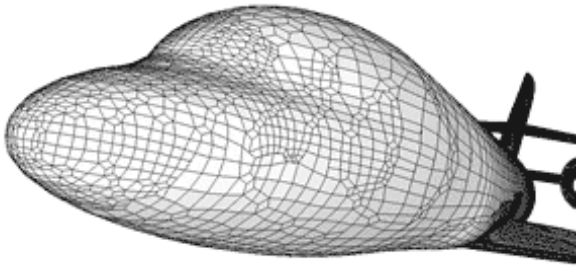


Figure 2. Surface mesh issued from an octree method

aligned with the cross field direction and their triangles are almost right everywhere. Then, recombination with our Blossom-quad approach [1] yields quadrilateral meshes of excellent quality.

2. Surfaces

Only surface meshing is considered in this paper. The parametrization of a surface $\mathcal{S}$ is a map

$$\mathbf{u} = \{u, v\} \in \mathcal{S}' \subset \mathcal{R}^2 \mapsto \mathbf{x}(\mathbf{u}) = \{x, y, z\} \in \mathcal{S} \subset \mathcal{R}^3. \tag{1}$$

The metric tensor

$$M = \begin{bmatrix} \frac{\partial \mathbf{x}}{\partial u} \cdot \frac{\partial \mathbf{x}}{\partial u} & \frac{\partial \mathbf{x}}{\partial u} \cdot \frac{\partial \mathbf{x}}{\partial v} \\ \frac{\partial \mathbf{x}}{\partial v} \cdot \frac{\partial \mathbf{x}}{\partial u} & \frac{\partial \mathbf{x}}{\partial v} \cdot \frac{\partial \mathbf{x}}{\partial v} \end{bmatrix} \tag{2}$$

allows to measure angles and distances in the parametric plane $\{u, v\}$.

A mapping $\mathbf{x}(\mathbf{u})$ is said to be conformal if the eigenvalues of M are identical. Conformal mappings preserve angles, and hence the information carried by the cross field. Moreover, it is possible to obtain a uniform isotropic surface mesh on the surface $\mathcal{S}$ by constructing a non-uniform isotropic mesh in the parameter plane. If $\delta(\mathbf{x})$ is the prescribed mesh size field on $\mathcal{S}$, the size field in the parametric plane is simply $\delta'(\mathbf{u}) = \delta(\mathbf{x}(\mathbf{u}))|\det M|^{1/4}$.

Surfaces considered in this paper are assumed to have a conformal parametrization. This might be seen as a severe limitation, for many standard parametrizations are not conformal (e.g. spherical coordinates or transfinite maps). The issue of non conformal maps will however be dealt with in a forthcoming paper where anisotropic quad meshing will be addressed. Moreover, in case of a non-conformal surface parametrization, it is always possible to build a conformal one using reparametrization techniques [9, 10, 11, 12].

As an example, let us consider the example of the car hood depicted in Figure 3. The CAD is made of 18 Bézier surfaces. A first surface triangulation is generated that is consistent with the CAD description, i.e. elements cover the surface patches exactly. This first triangulation is then used to build a conformal parametrization of the whole hood. For that, a PDE is solved on the initial triangulation to compute the parameter $\mathbf{u}$ of every vertex of the initial triangulation.

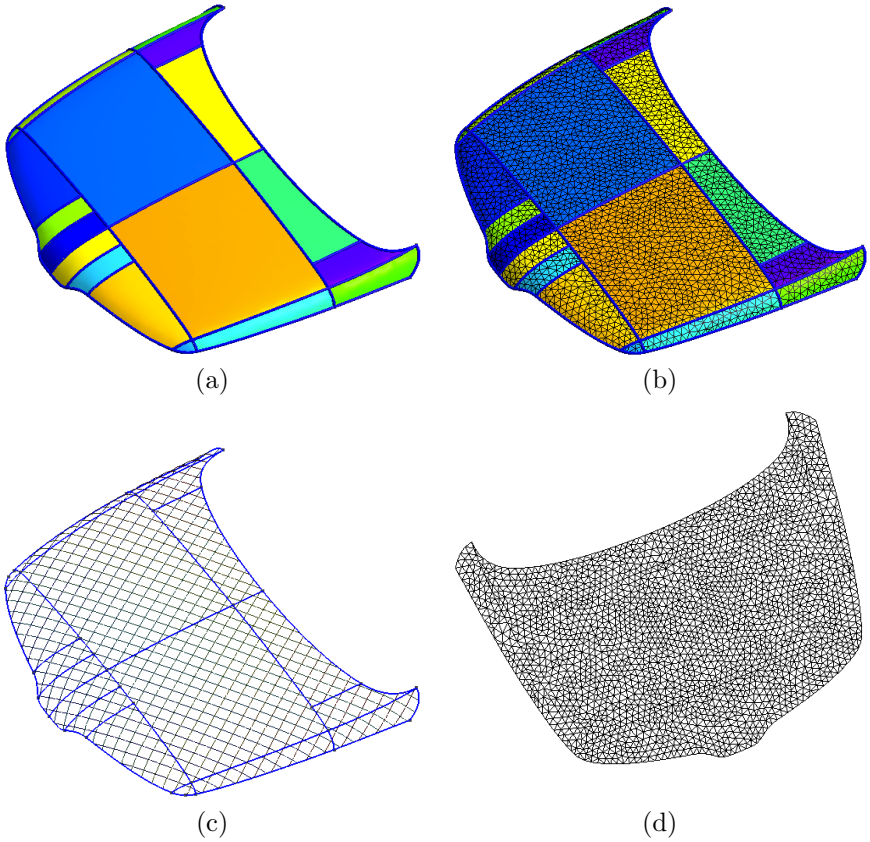


Figure 3. Conformal reparametrization of a car hood. Figure (a) shows the CAD decomposition of the hood into 18 Bézier surfaces. Figure (b) shows the initial conforming mesh. Figure (c) shows the result of the conformal reparametrization i.e. isolines of u and v on the whole hood. Figure (d) shows the projection of the initial mesh on the parameter plane $\mathbf{u}$.

3. Cross fields

The cross fields represent at each point of the domain the preferred orientations of the quadrilateral mesh. In two dimensions, a sufficient information for defining a cross field is the angle $\theta(\mathbf{u})$ that defines the orientation of a local frame at each point $\mathbf{u}$ in the parameter plane. The edges of the quadrilaterals generated around $\mathbf{u}$ should then be aligned with the the cross field. Figure 4 shows an annular domain, the cross field and the resulting quad mesh. The computer graphics community has already been confronted with the issue of computing "cross fields" in the context of (global) surface parametrization [13, 9]. Cross fields can be based on principal directions of curvature of the surface [7].

Here, we consider an *ad hoc* approach based on the following criteria:

- The cross field should be computed automatically;

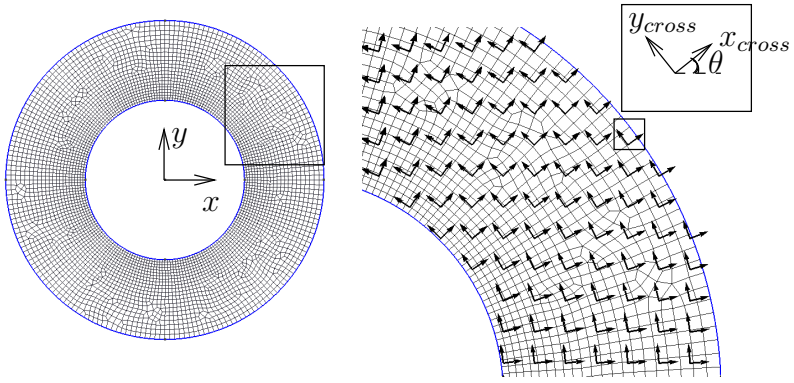


Figure 4. Mesh of annular domain and a zoom on the cross field.

- Mesh directions should be parallel to the boundaries of the domain at the vicinity of those boundaries;
- The cross field should be smooth.

In order to fulfill those constraints, we have chosen to compute θ using a boundary value problem. The value of θ is fixed at the boundary of the domain and is propagated inside the domain using an elliptic PDE. The value of θ at the boundary is chosen in such a way that the local axis at the boundary is aligned with the geometry of the boundary.

The angular orientation of a cross being defined up to the multiples of $\pi/2$, it cannot be represented univocally by the orientation of one branch of the cross. The complex valued function

$$\alpha(\mathbf{u}) = a(\mathbf{u}) + ib(\mathbf{u}) = e^{4i\theta(\mathbf{u})}$$

however offers a univocal representation, as it takes one same value for the directions of the 4 branches of a local cross.

A first triangular mesh $\mathcal{T}_0$ is generated using any available algorithm. If a parametrization of the surface needs to be computed, then we use the same mesh as the one that has been used for computing the parametrization. Two Laplace equations with Dirichlet boundary conditions are then solved for each surface of the mesh in order to compute the real part $a(\mathbf{u}) = \cos 4\theta$ and the imaginary part $b(\mathbf{u}) = \sin 4\theta$ of α:

$$\begin{aligned} \nabla^2 a = 0, \quad \nabla^2 b = 0 \quad &\text{on} \quad \mathcal{S}', \\ a = \bar{a}(\mathbf{u}), \quad b = \bar{b}(\mathbf{u}) \quad &\text{on} \quad \partial\mathcal{S}'. \end{aligned} \tag{3}$$

Then, we have to supply the boundary conditions $\bar{a}(\mathbf{u})$ and $\bar{b}(\mathbf{u})$ ensuring that θ is aligned with $\partial\mathcal{S}'$. Consider Figure 5. At each point $\mathbf{u}_0$ of the surface $\mathcal{S}'$, Dirichlet boundary conditions are associated to the local normal $\mathbf{n}$ of the surface. At point $\mathbf{u}_0$, we choose $\bar{a} = \cos 4\theta$ and $\bar{b} = \sin 4\theta$.

After solving, the cross field is represented by

$$\theta(\mathbf{u}) = \frac{1}{4}\text{atan2}(b(\mathbf{u}), a(\mathbf{u})).$$

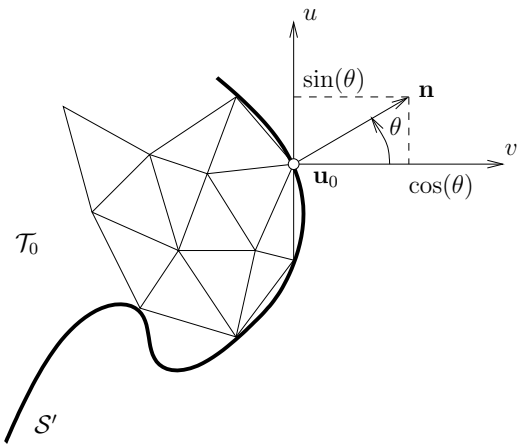


Figure 5. Boundary conditions for the computation of cross fields.

As an example, Figure 6 shows the cross field for a surface with many circular holes.

4. Triangulation in the L^∞-norm

In this section, usual geometrical notions defined in the L^2-norm are extended in the L^∞ norm.

4.1. Distances and norms

In the R^2 plane, the distance between two points $\mathbf{x}_1(x_1, y_1)$ and $\mathbf{x}_2(x_2, y_2)$ is usually based on the Euclidean norm (L^2-norm). Other distances can be defined however, based on other norms:

- The L^1-norm distance $\|\mathbf{x}_2 - \mathbf{x}_1\|_1 = |x_2 - x_1| + |y_2 - y_1|$,
- The L^2-norm distance $\|\mathbf{x}_2 - \mathbf{x}_1\|_2 = (|x_2 - x_1|^2 + |y_2 - y_1|^2)^{1/2}$,
- The L^p-norm distance $\|\mathbf{x}_2 - \mathbf{x}_1\|_p = (|x_2 - x_1|^p + |y_2 - y_1|^p)^{1/p}$,
- The L^∞-norm distance $\|\mathbf{x}_2 - \mathbf{x}_1\|_\infty = \lim_{p\to\infty}\|\mathbf{x}_2 - \mathbf{x}_1\|_p = \max(|x_2 - x_1|, |y_2 - y_1|)$.

Figure 7 shows unit circles in different norms. One important thing to remark is that only the L^2-norm is rotation invariant. All the other norms, including the L^∞-norm, depend on the local orientation of the coordinate axes.

In order to simplify the notations, in what follows we consider that (x, y) are local coordinates aligned with the cross field (x_{cross}, y_{cross}) (see for example the cross field in Fig. 4).

4.2. Bisectors in the L^∞-norm

The perpendicular bisector, or bisector of the segment delimited by the points $\mathbf{x}_1 = (-x_p, -y_p)$ and $\mathbf{x}_2 = (x_p, y_p)$ is by definition the set of points $\mathbf{x} = (x, y)$ equidistant to $\mathbf{x}_1$ and $\mathbf{x}_2$. It is the union of the intersections of circles centered at $\mathbf{x}_1$ and $\mathbf{x}_2$ and having the same radius. In the

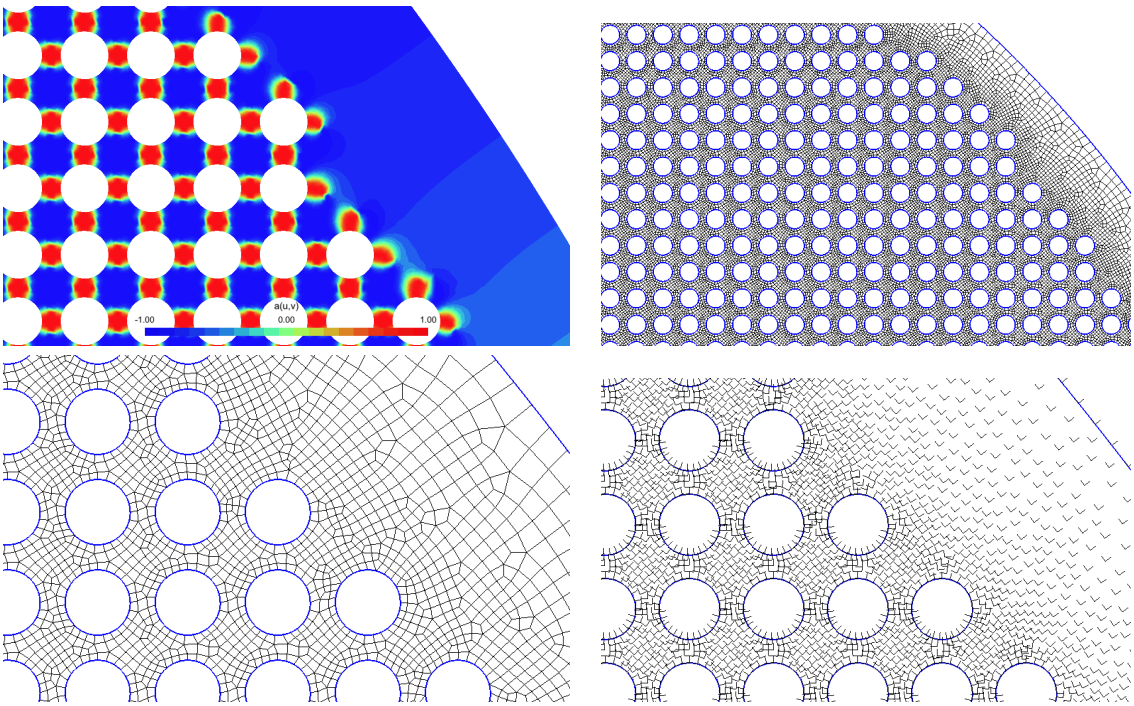


Figure 6. Computation of the cross field for one surface with many holes. Top left figure shows the $a(\mathbf{u})$ field. Bottom right figure shows the cross field. The two other figures show the meshes resulting from the new algorithm.

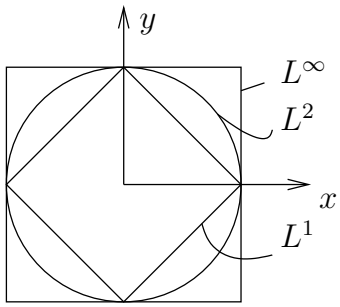


Figure 7. Illustration of the unit circle in different norms $\|\mathbf{x}\|_p$

L^2-norm, those intersections are each time two points and their union forms a straight line. In the L^∞-norm, the L^∞-circles have the geometric appearance of squares and their intersections 2 by 2 are either 2 points or a segment. The bisector is then a broken line in general but it

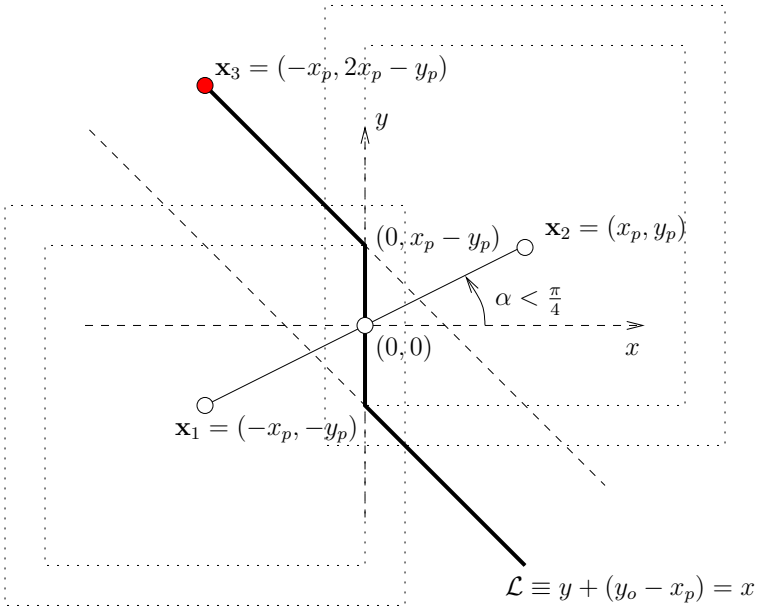


Figure 8. Bisector of two points $\mathbf{x}_1 = (-x_p, -y_p)$ and $\mathbf{x}_2 = (x_p, y_p)$ using the L^∞-norm.

can also form a diabolo-shaped region whenever the points are aligned with an axis ($x_1 = x_2$ or $y_1 = y_2$), Fig. 9.

This is shown on an example, Figure 8. It is assumed, without loss of generality, that $x_p \geq y_p$. The bisector of the segment in the L^∞-norm is the set

$$\{\mathbf{x} = (x, y), \max(|x - x_p|, |y - y_p|) = \max(|x + x_p|, |y + y_p|)\}.$$

The vertical segment Figure 8 through the origin $(0, 0)$ is the intersection of the L^∞-circles of L^∞-radius x_p centered at $\mathbf{x}_1$ and $\mathbf{x}_2$. It thus belongs to the bisector. Increasing now the radius progressively, the intersection of the two L^∞-circles is a pair of points forming two half lines oriented at $3\pi/4$ and starting at $(0, x_p - y_p)$ and $(0, -x_p + y_p)$ respectively. The distance in the L^∞-norm between a point $\mathbf{x} = (x, y)$ of the bisector and $\mathbf{x}_1$ and $\mathbf{x}_2$ is $x_p - x$ or $y_p - y$. There exists an ambiguity when $y_p = 0$. In this case, the bisector contains 2D regions of the plane, as depicted in Fig. 9. It is assumed in what follows that points are in general position, i.e. that there does not exist two points that share either the same x or y coordinate.

4.3. The equilateral triangle using the L^∞-norm

A triangle $T(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3)$ is equilateral in the L^∞-norm if

$$\|\mathbf{x}_2 - \mathbf{x}_1\|_\infty = \|\mathbf{x}_3 - \mathbf{x}_1\|_\infty = \|\mathbf{x}_3 - \mathbf{x}_2\|_\infty.$$

It is possible to build such a triangle starting from Figure 8. We take again $\mathbf{x}_1 = (-x_p, -y_p)$ and $\mathbf{x}_2 = (x_p, y_p)$ and look for a point on the bisector of $\mathbf{x}_1\mathbf{x}_2$ located at a distance $2x_p$ from

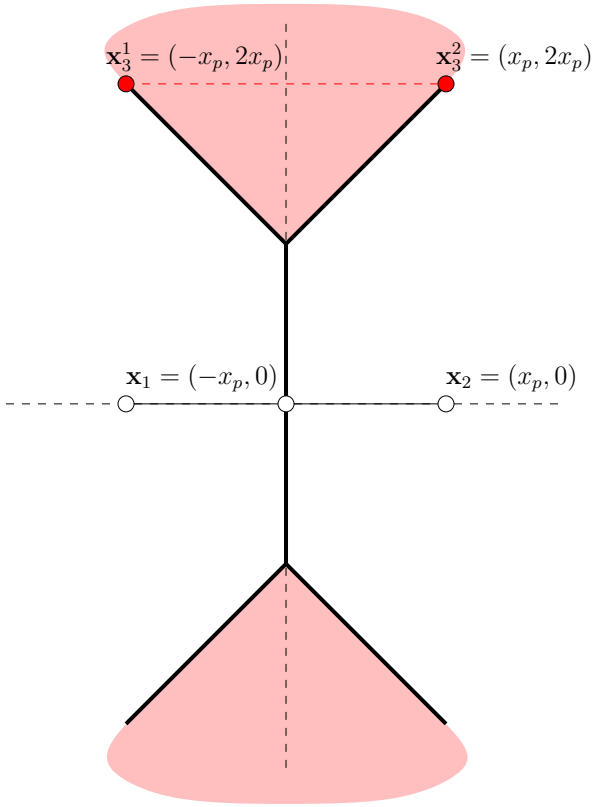


Figure 9. Bisector of two points $\mathbf{x}_1 = (-x_p, 0)$ and $\mathbf{x}_2 = (x_p, 0)$ using the L^∞-norm. The pink regions belong to the bisector as well. The dotted red line is the locus of points that make an equilateral triangle with $\mathbf{x}_1$ and $\mathbf{x}_2$.

the endpoints of the segment. This point is $\mathbf{x}_3 = (-x_p, 2x_p - y_p)$. There are two interesting special cases to consider.

The first one, Figure 9, is when the segment $\mathbf{x}_1\mathbf{x}_2$ is aligned with the $x-$axis, $y_p = 0$. The third vertex is then any point on the segment between $\mathbf{x}_3^1(-x_p, 2x_p)$ and $\mathbf{x}_3^2(x_p, 2x_p)$ and the equilateral triangle in the L^∞-norm is geometrically a half square (note that it is isosceles in the L^2-norm).

The second case is when $x_p = y_p$. The third vertex of the equilateral triangle is then $\mathbf{x}_3 = (-x_p, x_p)$ and the equilateral triangle is again a half square.

4.4. Circumcenter, circumradius and circumsquare in the L^∞-norm

Consider a triangle $T(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3)$. Its circumcenter $\mathbf{x}_c = (x_c, y_c)$ in the L^∞-norm verifies

$$\|\mathbf{x}_1 - \mathbf{x}_c\|_\infty = \|\mathbf{x}_2 - \mathbf{x}_c\|_\infty = \|\mathbf{x}_3 - \mathbf{x}_c\|_\infty.$$

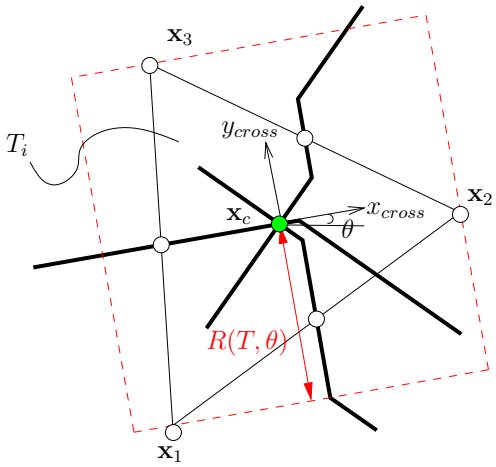


Figure 10. Circumcenter $\mathbf{x}_c$ and circumradius $R_\infty(T,\theta)$ of a triangle T_i using the L^∞-norm. The circumsquare is the red dotted square.

The L^∞-circumcenter of the triangle is located at the intersection of the L^∞-perpendicular bisectors.

The circumcircle in the L^∞-norm (also called circumsquare), is the smallest square centered at the circumcenter that encloses the triangle—see Figure 10. The circumradius $R_\infty(T,\theta)$ is the distance in the L^∞-norm between the circumcenter and anyone of the three vertices. It is given by:

$$R_\infty = \frac{1}{2}\max\left((\max(x_1,x_2,x_3)-\min(x_1,x_2,x_3),(\max(y_1,y_2,y_3)-\min(y_1,y_2,y_3)\right).$$

Consider now the right triangle of Figure 11 with two orthogonal sides aligned with the axis. This example illustrates two counterintuitive properties of circumcenters and circumradii in L^∞-norm. First, the circumcenter is not unique: any point $\mathbf{x}_c$ on the semi-infinite broken line $\mathbf{x}_c^1\mathbf{x}_c^2\mathbf{x}_c^3$ is equidistant to $\mathbf{x}^1,\mathbf{x}^2$ and $\mathbf{x}^3$, i.e. $\|\mathbf{x}_1-\mathbf{x}_c\|_\infty = \|\mathbf{x}_2-\mathbf{x}_c\|_\infty = \|\mathbf{x}_3-\mathbf{x}_c\|_\infty$. The circumradius is thus not unique and the circumcenters located between $\mathbf{x}_c^1$ and $\mathbf{x}_c^2$ correspond to the smallest circumradius. The circumsquares C^1 and C^2 centered at $\mathbf{x}_c^1$ and $\mathbf{x}_c^2$ have the same size. Then, for $\mathbf{x}_c$ between $\mathbf{x}_c^1$ and $\mathbf{x}_c^2$ the circumradius increases, as depicted on the Figure. Note that circumcenter and circumsquare are unique when the points are in general position.

One interesting fact is that the computation of circumcenters and circumradii in theL^∞-norm is a very stable numerical operation.

5. A frontal-Delaunay mesher in the L^∞-norm

Let us recall briefly the pros and cons of the two main approaches for mesh generation. Advancing front techniques start from the discretization of the boundary (edges in 2D). The

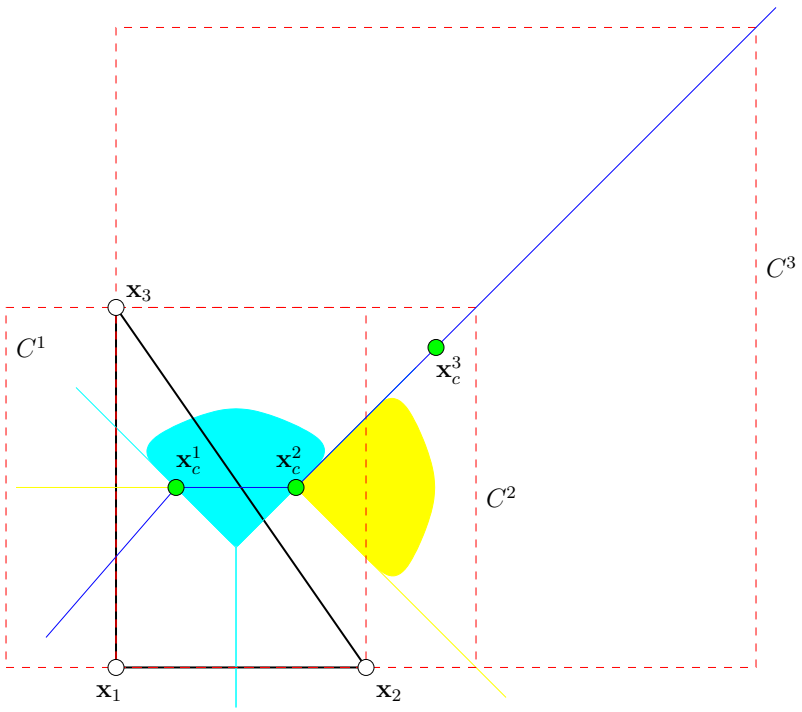


Figure 11. A right triangle. Perpendicular bisectors of the three segments are coloured in yellow (edge $\mathbf{x}_1\mathbf{x}_3$), blue (edge $\mathbf{x}_2\mathbf{x}_3$) and cyan (edge $\mathbf{x}_1\mathbf{x}_2$). Points $\mathbf{x}_c^1$, $\mathbf{x}_c^2$ and $\mathbf{x}_c^3$ are three circumcenters that correspond to the three circumsquares C^1, C^2 and C^3.

set of edges of the boundary discretization is called the front. A particular edge of this front is selected and a new triangle is formed with this edge as its base and the front is updated accordingly. The algorithm advances in the domain until the front is emptied and the domain fully covered by triangles. The main advantage of advancing front techniques is that they generate points and triangles at the same time, which makes it possible to build optimum triangles, e.g. equilateral triangles. The main drawback of the method is that parts of the front advance independently, leading to possible clashes when they meet.

Delaunay-based mesh generation techniques are more robust because a valid mesh exists at each stage of the mesh generation process. Yet, inserting a point using the Delaunay kernel requires the creation and the deletion of a number of triangles, so that there is less control on the shapes of the element than in the advancing front technique.

The frontal Delaunay approach makes the best of both techniques. As it is based on a Delaunay kernel, a valid mesh is maintained at every stage of the process. Yet, some kind of front is defined in the triangulation and points are inserted in a frontal manner. The process stops when every element of the mesh has the right size according to the size field $\delta(\mathbf{x})$.

The ideas of the new frontal-quad algorithm are inspired by the frontal Delaunay approach of [8].

Consider a surface $\mathcal{S}$, a mesh size field $\delta(\mathbf{x})$, a cross field $\theta(\mathbf{u})$ and a conformal

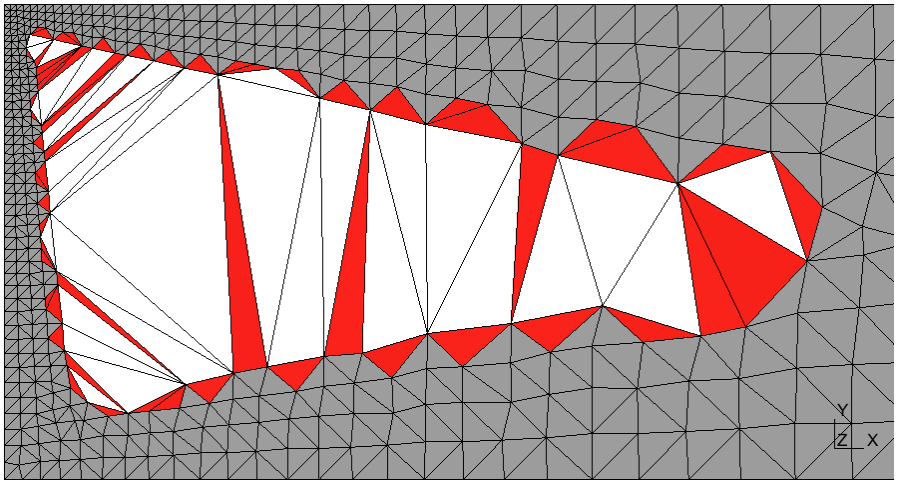


Figure 12. Illustration of the frontal algorithm with resolved (grey), active (red) and waiting (white) triangles.

parametrization with a metric $M(\mathbf{u})$. An initial mesh $\mathcal{T}_0$ is constructed in the parameter plane that contains the points of the discretization of the boundaries only. An adimensional L^∞-meshsize

$$h_i = \frac{R_\infty(T_i, \theta(\mathbf{u}))}{\delta(\mathbf{u})|\det M(\mathbf{u})|^{1/4}} = \frac{R_\infty(T_i, \theta(\mathbf{u}))}{\delta'(\mathbf{u})}$$

is defined for each triangle T_i of $\mathcal{T}_0$. Quantities $\theta(\mathbf{u})$, $M(\mathbf{u})$ and $\delta(\mathbf{x}(\mathbf{u}))$ are evaluated at the usual centroid of the triangle. Triangles are then classified into three categories

1. A triangle is *resolved* if $h_i \leq h_{\max}$;
2. A triangle is *active* if $h_i > h_{\max}$ and, either one of its three neighbors is resolved or one of its sides is on the boundary of the domain;
3. A triangle is *waiting* if it is neither resolved nor active.

We choose $h_{\max} = 4/3$. This choice is standard in the domain of mesh generation [14]. Figure 12 is an illustration of the way triangles are classified in the algorithm. The front is defined as the set of active triangles. Active triangles are sorted with respect to h_i. Front edges are therefore defined as those edges separating active and resolved triangles.

The frontal algorithm inserts a new point so as to form an optimal triangle with the edge corresponding to the largest active triangle. Consider the edge $\mathbf{x}_2\mathbf{x}_3$ in Figure 13 and assume it corresponds to the largest active triangle of the mesh (the red triangle on Figure 13). For the discussion, the coordinate system has been centered at the mid-edge point $\mathbf{x}_m = \frac{1}{2}(\mathbf{x}_2 + \mathbf{x}_3)$ and aligned with the local cross field, this can be achieved by a translation and a rotation of angle $\theta(\mathbf{x}_m)$.

The position of the new point $\mathbf{x}_n$ on the L^∞-perpendicular bisector $\mathcal{L}$ of $\mathbf{x}_2\mathbf{x}_3$ is chosen in order to fullfill the size criterion $\delta(\mathbf{x}_m)$. In order to create a new triangle $T_i(\mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_n)$ of size

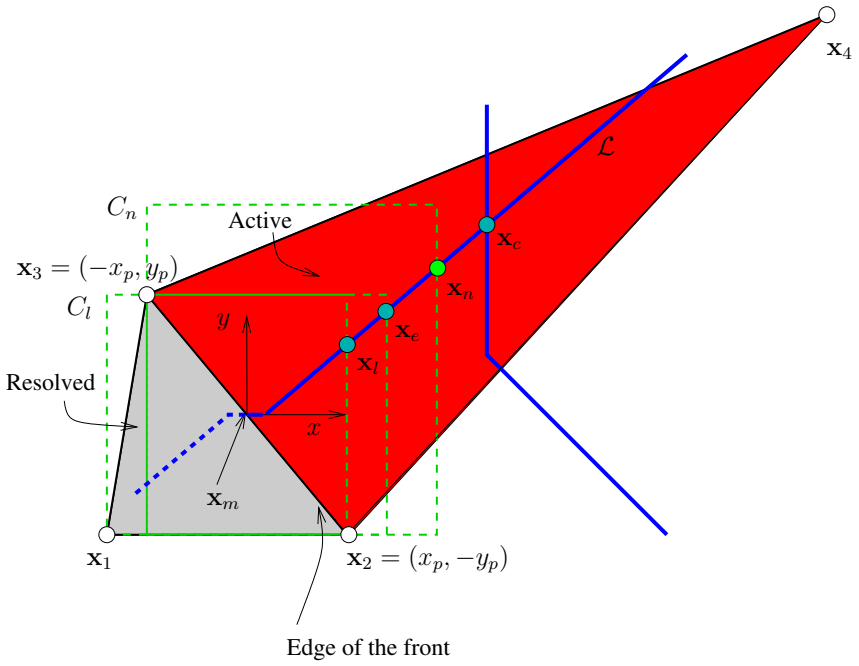


Figure 13. Illustration of the point insertion algorithm.

$R_\infty(T_i, \theta) = \delta'(\mathbf{x}_m)$, $\mathbf{x}_n$ is placed at the intersection of $\mathcal{L}$ with the square C_n of side $\delta'(\mathbf{x}_m)$ passing through points $\mathbf{x}_2$ and $\mathbf{x}_3$ (see Figure 13).

The following considerations should be made.

- The new point should not be placed beyond the center $\mathbf{x}_c$ of the circumsquare of the active triangle (red triangle in Figure 13), as this would create a triangle with a small edge $\mathbf{x}_n\mathbf{x}_4$. Note that this limit case corresponds to a classical point insertion scheme where new points are inserted at the center of the circumcircle of the worst triangle, yet in the L^∞-norm in this case.
- The new point should not be placed below the intersection $\mathbf{x}_l$ of the bisector $\mathcal{L}$ and the circumsquare C_l of the resolved triangle $(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3)$. Inserting a point inside C_l would make the resolved triangle invalid by means of the Delaunay criterion.
- If $\delta'(\mathbf{x}_m) = \|\mathbf{x}_3 - \mathbf{x}_2\|_\infty$, then the optimal point is $\mathbf{x}_n = \mathbf{x}_e$. It corresponds to the largest triangle $T_i(\mathbf{x}_e, \mathbf{x}_2, \mathbf{x}_3)$ that verifies $R_\infty(T_i, \theta) = \delta'(\mathbf{x}_m)$.

The equation of the rightmost segment of the bissector $(y > 0, x > y_p - x_p)$ is

$$\mathcal{L} \equiv y + (y_p - x_p) = x.$$

One has

$$\mathbf{x}_c = \left(\frac{1}{2}(x_4 - x_p), \frac{1}{2}(x_4 + x_p) - y_p\right),$$

$$\mathbf{x}_e = (\delta'(\mathbf{x}_m) - x_p + y_p, \delta'(\mathbf{x}_m)) \quad \text{and} \quad \mathbf{x}_l = (\delta'(\mathbf{x}_m), \delta'(\mathbf{x}_m) + x_p - y_p),$$

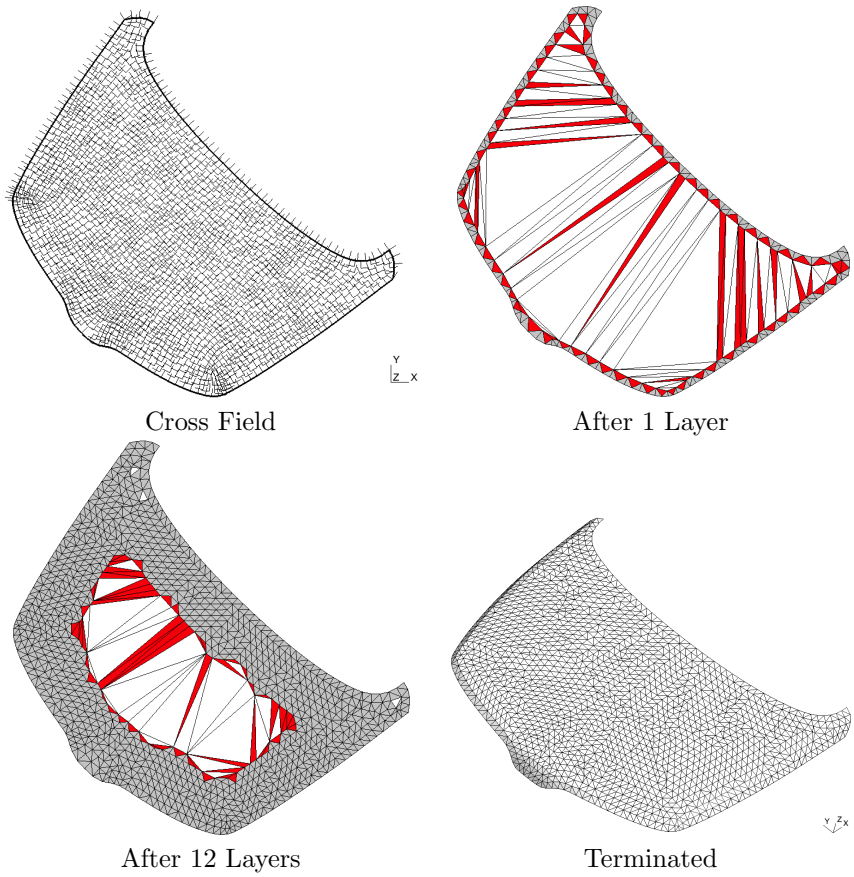


Figure 14. Illustration of the advancing front process with a uniform size field for the car hood problem. White triangles are waiting triangles, red triangles are active triangles and grey triangles are resolved triangles.

and the position of the optimal point is computed as follow:

$$\mathbf{x}_n = \mathbf{x}_e + t(\mathbf{x}_c - \mathbf{x}_e)$$

with

$$t = \min\left(\max\left(1, \frac{\|\mathbf{x}_3 - \mathbf{x}_2\|_\infty - \delta'(\mathbf{x}_m)}{\|\mathbf{x}_3 - \mathbf{x}_2\|_\infty - \|\mathbf{x}_c - \mathbf{x}_2\|_\infty}\right), -\frac{\|\mathbf{x}_l - \mathbf{x}_e\|_2}{\|\mathbf{x}_c - \mathbf{x}_e\|_2}\right).$$

Another important ingredient of the advancing front strategy holds in the fact that successive fronts are initiated layer by layer. An initial front consists of the edges of the 1D boundary discretization. The algorithm inserts points until every edge of the active front has been treated. Then new fronts are created and emptied until no active triangle is left in the mesh as illustrated in Figure 14.

One last and important point is the choice of the Delaunay kernel to connect nodes at each stage of the point insertion procedure. All arguments that have been developed concerning the

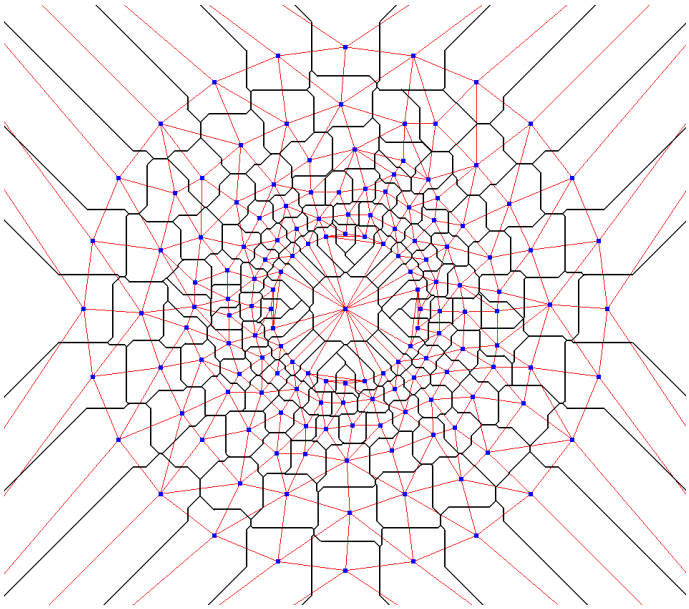


Figure 15. Voronoi diagram (dark lines) and Delaunay triangulation, both in the L^∞ norm.

point insertion strategy advocates for using a Delaunay kernel in the infinity norm.

Yet, it has been observed experimentally that, in the case of finite element meshes with decent point distribution properties, the Delaunay kernel in the standard L^2-norm and the Delaunay kernel in the L^∞-norm give similar triangulations. We have then chosen to use a standard Delaunay kernel for connecting points because such a kernel is available in most mesh generators[†]. As an example, Figure 15 shows the Voronoi diagram and the Delaunay triangulation of a set of points in the L^∞-norm. Although the Voronoi diagram is significantly different from a standart L_2-based Voronoi diagram, the triangulation (in red) is very similar to a usual Delaunay triangulation. To be specific, only 5 edge swaps are necessary to modify the mesh of Figure 15 and transform it into a standard Delaunay triangulation. Levy et al. reached the same conclusion in their L^p Central Voronoi Tesselation approach of [7] where they minimize the Lloyd energy in high order norms on the standard Voronoi.

6. Mesh Quality Measures

For mesh validatation purposes, a number of useful quality measures for quadrilateral elements are now defined.

[†]This is the case in Gmsh [15].

6.1. Quadrilateral Shape Measure

Consider a quadrilateral element q and its four internal angles $\alpha_k, \quad k = 1, 2, 3, 4$. We define the quality $\eta(q)$ of q as:

$$\eta(q) = \max\left(1 - \frac{2}{\pi}\max_k\left(\left|\frac{\pi}{2} - \alpha_k\right|\right), 0\right). \tag{4}$$

This quality measure is 1 if the element is a square and 0 if one of its angles is either ≤ 0 or $\geq \pi$. The average element quality over a mesh $\bar{\eta}$ and the worst element quality in a mesh η_w are also useful in the context of finite element simulations.

6.2. Size Field Efficiency

The adimensional length of the vector $\mathbf{y} = \mathbf{b} - \mathbf{a}$ of R^3 with respect to the non uniform size field $\delta(\mathbf{x})$ is defined as

$$l = \|\mathbf{y}\| \int_0^1 \frac{1}{\delta(\mathbf{a} + t\mathbf{y})}\, dt. \tag{5}$$

An optimum mesh in terms of size is a mesh for which the adimensional size l_i of all edges is equal to one. It is of course impossible in practice to have such an optimum mesh. Therefore, the *efficiency index* [14] τ of a mesh is defined as the exponential of the mean value of the difference d_i between each edge's length l_i and one, i.e.

$$\tau[\%] = 100 \exp\left(\frac{1}{n_e}\sum_{i=1}^{n_e} d_i\right), \tag{6}$$

with $d_i = l_i - 1$ if $l_i < 1$, $d_i = \dfrac{1}{l_i} - 1$ if $l_i > 1$ and n_e the number of edges in the mesh. Surface mesh algorithms usually produce triangular meshes with typical values of τ around $\tau = 85\%$, i.e., with adimensional sizes around $1/\sqrt{2} \leq l_i \leq \sqrt{2}$.

6.3. Mesh Topology Criteria

Good quadrilateral meshes are made of long layers of quadrilaterals that typically intersect orthogonally with each others. The optimal topology of a vertex in such a mesh is a vertex with 4 adjacent quadrangles. The parameter d_4 is the percentage of vertices in the mesh that have 4 quadrilateral neighbors. Boundary vertices are not taken into account here. Finally, $d_{\max}$ is the maximum number of quadrilaterals adjacent to one vertex.

6.4. A simple example

As a first example of the application of Delquad, let us consider the benchmark problem that has been first defined by Lewis in [16]. We consider a rectangular domain with $(x, y) \in [-1.25, 1.25] \times [-0.5, 1.25]$. We choose

$$\delta(x, y) = \frac{1}{100}\left(1 + 30(x - y^2)^2 + (1 - x)^2\right).$$

The cross fields directions have been analytically chosen in such a way that it correspond to the orientation of the gradient of the size field δ.

This simple test case has been used for benchmarking quad meshers in [5] and in [17]. We therefore compare our results with the results of those two papers. The technique proposed in [5] is an indirect technique that produces nearly all quad meshes. The algorithm proposed in [17] is a mesh adaptation algorithm. It produces quad dominant meshes using L^∞ norm arguments for optimizing point positions and element connectivities.

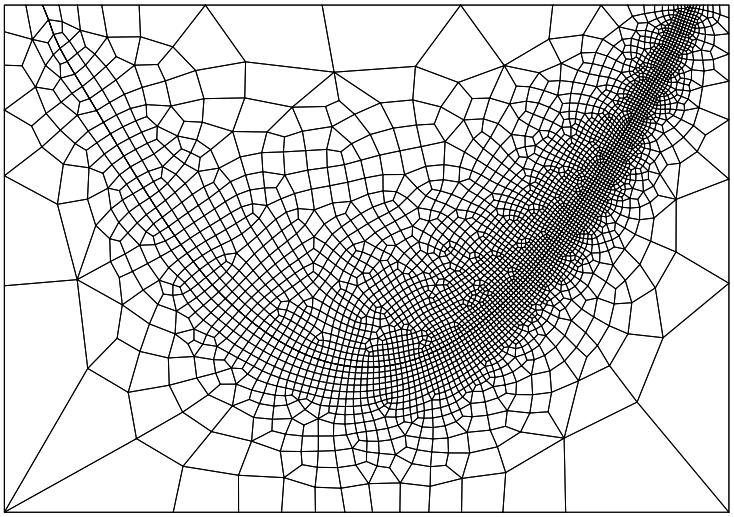


Figure 16. Mesh with the size field proposed by [16]. The Delquad point position algorithm was used and the triangular mesh was subsequently recombined using Blossom-Quad.

Algorithm	Quad quality		Degree vertices			Efficiency	Perc. quads
	η_w	$\bar{\eta}$	$d_4(\%)$	d_{min}	d_{max}	τ	
Borouchaki [5]	0.23	0.63	91	3	6	–	97 %
Tchon [17]	0.53	0.89	–	–	–	–	84 %
Delquad	0.25	0.85	87	3	6	88%	100 %

Table I. Quality of the quad meshes for the analytical test case of [16]. We present values for the minimum quality η_w, mean quality $\bar{\eta}$, percentage of vertices of degree 4, minimal and maximal values for the degree of vertices, and efficiency index τ.

The quality of the quadrilaterals in this mesh is substantially superior to the quality of the merged elements obtained by Borouchaki and Frey [5], i.e., an average quality of 0.85 versus 0.63. The average quality produced bt Delquad is similar to the one found in [17] while the percentage of quadrilateral elements in [17] is only 84%.

7. Indirect quadrilateralization

The Delquad algorithm is an indirect algorithm. It first builds a triangular mesh, and a triangle merging procedure is subsequently used to generate quadrilaterals. We have shown in [1] that it is possible to find an optimal merging procedure that never leaves isolated triangles in the mesh.

The main advantage of the Delquad algorithm is that the points are placed in such a way that the triangle elements merge into quadrangles of optimal size and orientation. In order to illustrate this, quadrilateral meshes have been generated from different initial triangular meshes, obtained by a frontal algorithm [8] (favoring equilateral triangles) or the Delquad algorithm (favoring right triangles).

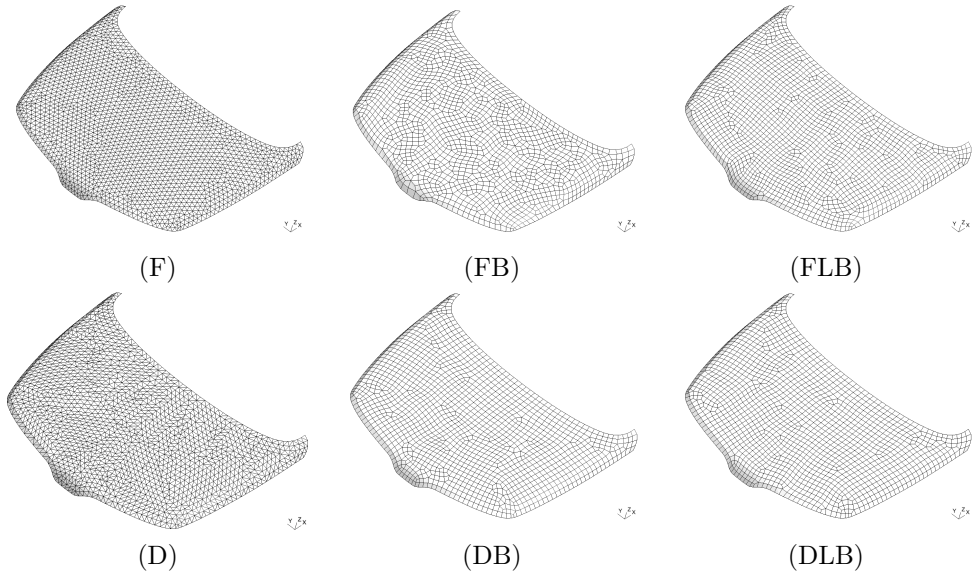
We use the following nomenclature. Concerning the point insertion algorithm, F stands for the Frontal algorithm of Gmsh [8, 15] while D stands for the new Delquad procedure. Concerning the recombination, the letter B denotes a mesh recombined using the Blossom-quad algorithm [1].

On Figure 17, the frontal mesh (F) contains 3524 triangles while the Delquad mesh (D) contains only 3066 triangles. The ratio $3066/3524 = 0,87$ is very close to the theoretical ratio $\sqrt{3}/2 = 0,866$ (see §1). The meshes (FB) and (DB) of Figure 17 are recombined meshes based on the frontal and the Delquad algorithm respectively. The (FB) mesh is typical of a quadrilateral mesh obtained by recombination of a triangular mesh with mostly equilateral triangles. Clearly, the mesh (DB) is better with respect to all criteria. It contains 13% less elements, which is very close to the expected value. As a consequence, the efficiency index of (DB) is much better than the one of (FB). The unstructured orientation in the (FB) implies that the mesh has to accommodate various changes of direction of the quadrilateral layers, leading to many vertices that do not have the optimal number of 4 adjacent quadrangles. The Delquad algorithm not only generates meshes with the right number of points, but it also generates meshes with the right topology: about 81% of the vertices of mesh (DB) have exactly 4 adjacent quadrangles.

Even though triangular meshes generated with the new Delquad algorithm are close to optimal nearly everywhere, the (ABL) is the quad mesh obtained with a L_p lloyds energy minimization computed before the Blossom recombination. of the triangulation remains sub-optimal in regions where fronts meet (see mesh (D) of figure 17). In a recent paper, Levy et al. [7] have proposed an interesting approach, called Lp Central Voronoi Tesselation (LPCVT), for quadrilateral and hexahedral smoothing based on an extension of the well-known Lloyd's algorithm [18]. In [19], an alternative version of Levy's approach has been proposed and applied to finite element meshes by the authors of this paper.

The LPCVT smoothing largely enhances the topology and the shape of the quadrilaterals when it is applied to a frontal triangulation. Yet, because our version of the LPCVT does not remove vertices, the mesh efficiency index is not enhanced. Meshes smoothed with the LPCVT technique are denoted by the letter L.

Mesh (DLB) that combines Delquad, LPCVT and Blossom-quad gives the best results with respect to all mesh statistics. Note that the (DB) mesh is already a very good mesh with statistics that are close to (DLB). The LPCVT is a rather complex procedure that is typically more expensive in CPU than the triangulation itself, even though there is room for improvement in our LPCVT algorithm.



	N	$\bar{\eta}$	η_w	$\tau(\%)$	$d_4(\%)$	$d_{\max}$
(FB)	1726	0.77	0.37	86	71	6
(FLB)	1744	0.86	0.39	87	81	5
(DB)	1527	0.90	0.37	93	81	5
(DLB)	1543	0.91	0.43	94	83	5

Figure 17. Pictures of different meshes of the car hood as well as mesh statistics.

8. Examples

8.1. Mechanical Part

The first example is a mechanical part composed of 114 surfaces (Figure 18). The quadrilateral mesh has been generated automatically with the new algorithm, the only control parameter being a uniform mesh size field. The mesh is composed of 66998 quads and was generated in 40 seconds. This time includes the generation of a first triangular mesh, the computation of cross fields on all surfaces and the conformal reparametrization of all non planar surfaces. The mesh statistics are as follows: $\tau = 97\%$, $d_4 = 91\%$, $d_{\max} = 6$, $\bar{\eta} = 0.95$, $\eta_w = 0.1$. The mesh generated by our approach for some of the planar and cylindrical surfaces of this particular geometry was a structured mesh. This happened however automatically, without user interaction or internal heuristics of the mesher. This explains the exceptionally high value of τ and $\bar{\eta}$ in this specific problem.

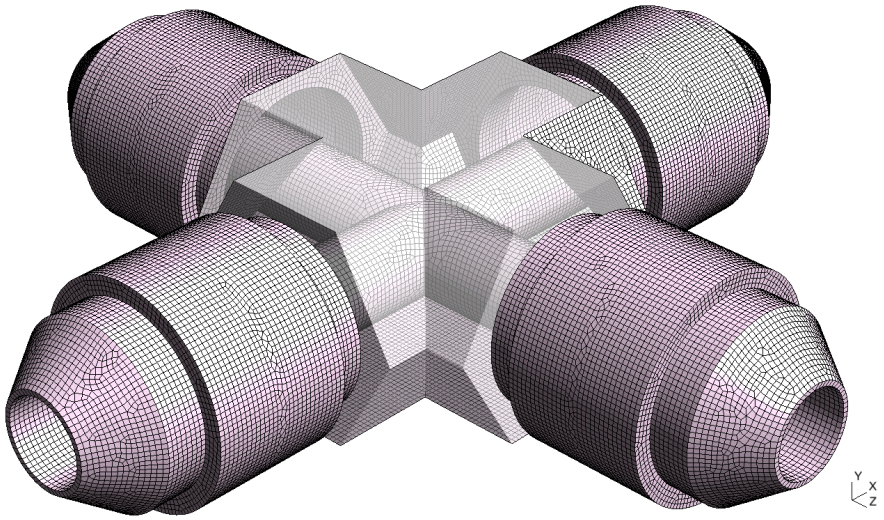


Figure 18. Quadrilateral mesh (DB) of a mechanical part.

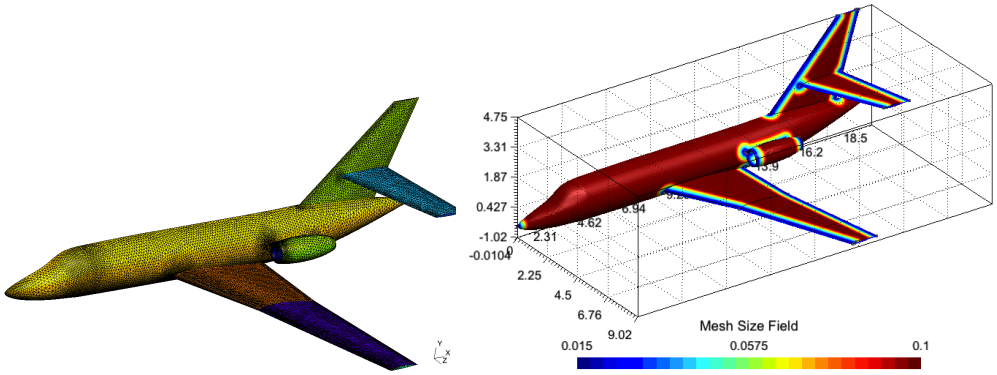


Figure 19. Surface triangular mesh of a Falcon aircraft (left) and mesh size field (right). Colors in the left Figure are indicative of the different surfaces of the model.

8.2. Falcon aircraft

As a second example, the Falcon aircraft of Figure 19 is considered. A surface mesh has been generated using a standard surface mesher, whose triangles were patched together to create the *compounds of surfaces* represented with different colors in the figure. Each compound surface has been reparametrized separately by means of a conformal map created using the techniques described in [7, 10, 11, 12]. The mesh size field is composed of a uniform bulk size field $\delta_b = 0.1$ augmented with line and point sources at critical zones of the aircraft, as depicted on Figure 19

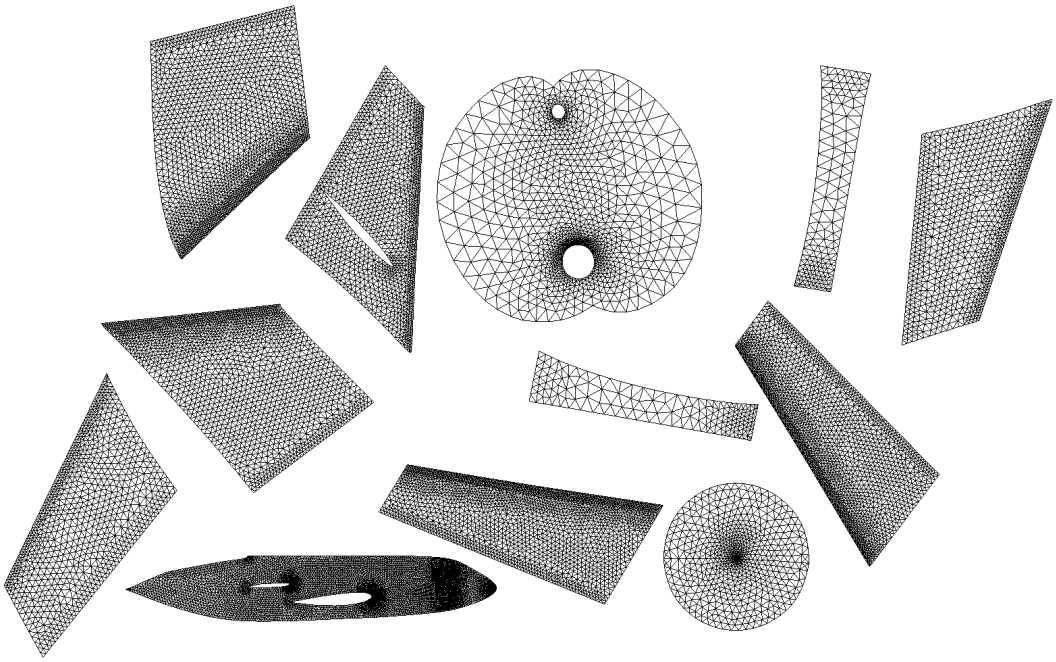


Figure 20. Parametrizations of the surfaces of the Falcon aircraft in the $\{u, v\}$ plane.

The images of the different surfaces in their respective parameter plane can be seen on Figure 20. The resulting mesh is presented on Figure 21. The mesh is composed of 53297 quadrangles. The total time for the surface meshing was 22 seconds. This time includes

- The reparametrization of the 12 surfaces (3 sec.),
- The Delquad algorithm applied to the 12 surfaces (10 sec.),
- The Blossom-quad recombination algorithmithm applied to the 12 surfaces (9 sec.).

The average and worst quality of the mesh are $\bar{\eta} = 0.86$ and $\eta_w = 0.17$, which can be considered as excellent. The efficiency of the mesh is $\tau = 0.92$ which is again very good.

9. Conclusion

A new method for automatic surface quadrilateralization has been proposed. The new algorithm uses distances in the L^∞ norm as a base for inserting new points and generate edges of the right size and orientation.

The new method has the following advantages:

- It is easy to implement as an extension of any robust 2D Delaunay kernel;
- The local orientation of the quadrilateral mesh is controlled by the cross field and not by the meshing algorithm itself;

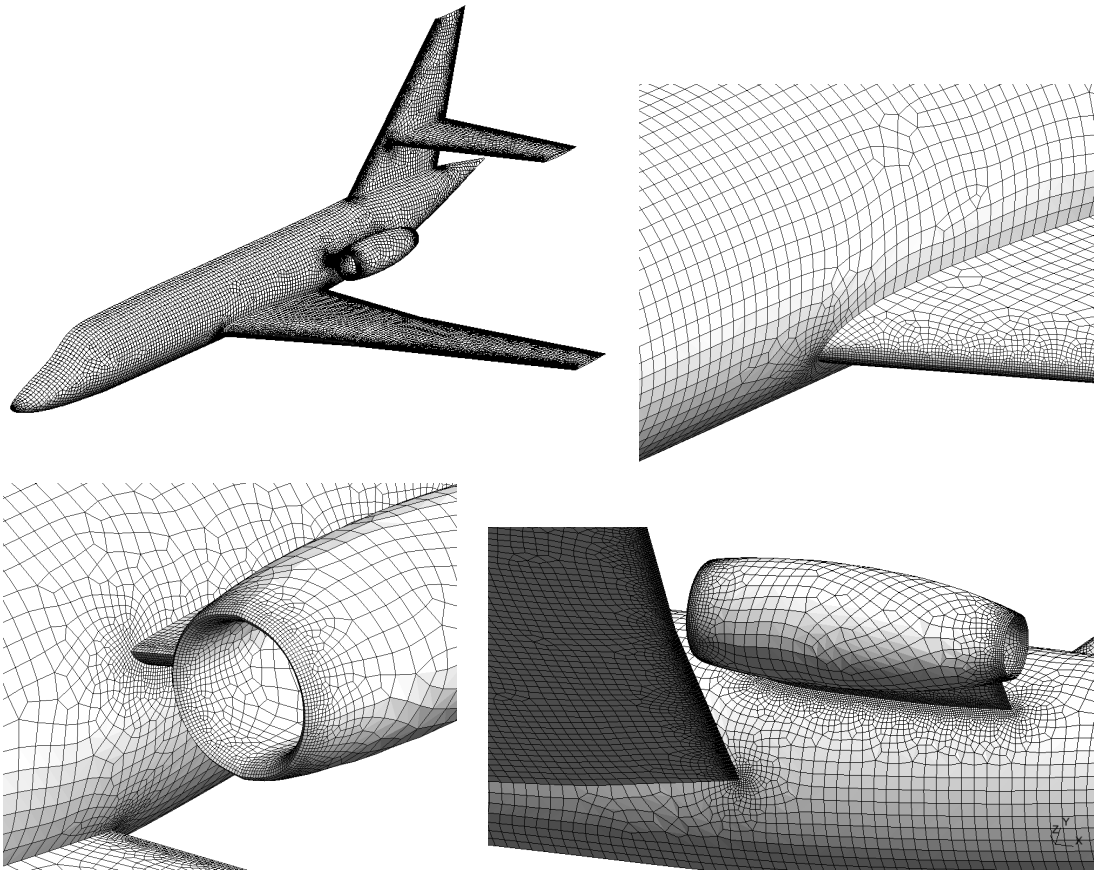


Figure 21. Final (DB) quadrilateral surface mesh of the Falcon aircraft.

- The new method accommodates size fields with strong variations;
- The method is fully automatic;
- The quality of the quadrilateral elements is very high in all aspects.

The method as it presented in this paper requires the availability of a conformal parametrization of the surfaces. In this work, we overcome this drawback by systematically reparametrizing the surfaces that we deal with in a conformal way [11, 12]. Even though reparametrization techniques are cheap and robust, this reparametrization step can be seen as a limitation of the method. One first extension of this work will be to extend the Delquad approach to anisotropic quad meshing. An anisotropic quadrilateral mesh ideally composed of parallelograms of controlled sizes and shear. Anisotropic metrics in the L^∞ norm can be defined (unit circles become parallelograms) and the whole approach presented in this paper can be reproduced, yet using an anisotropic Delaunay kernel. This extension will allow to deal

with surfaces that are not parametrized in a conformal way.

The automatic generation of hex-dominant meshes has been considered a challenge in the finite element community for many years now. There exists an extension of the Delquad approach proposed here to a Delhex algorithm that would generate 3D tetrahedral meshes with the right number of points and the right edge orientation to be recombined optimally into hexaedra.

REFERENCES

1. Remacle JF, Lambrechts J, Seny B, Marchandise E, Johnen A, Geuzaine C. Blossom-quad: a non-uniform quadrilateral mesh generator using a minimum cost perfect matching algorithm. *International Journal for Numerical Methods in Engineering* 2011; Submitted.
2. Blacker TD, Stephenson MB. Paving: A new approach to automated quadrilateral mesh generation. *International Journal for Numerical Methods in Engineering* 1991; **32**:811–847.
3. Frey P, Marechal L. Fast adaptive quadtree mesh generation. *in: Proceedings of the Seventh International Meshing Roundtable*, Citeseer, 1998.
4. Lee CK, Lo SH. A new scheme for the generation of a graded quadrilateral mesh. *Computers and Structures* 1994; **52**:847–857.
5. Borouchaki H, Frey P. Adaptive triangular–quadrilateral mesh generation. *International Journal for Numerical Methods in Engineering* 1998; **45**(5):915–934.
6. Owen SJ, Staten ML, Canann SA, Saigal S. Q-morph: An indirect approach to advancing front quad meshing. *International Journal for Numerical Methods in Engineering* 1999; **9**:1317–1340.
7. Lévy B, Liu Y. Lp centroidal voronoi tesselation and its applications. *ACM Transactions on Graphics (SIGGRAPH conference proceedings)*, 2010.
8. Rebay S. Efficient unstructured mesh generation by means of delaunay triangulation and bowyer-watson algorithm. *Journal of Computational Physics* 1993; **106**(1):125–138.
9. Lévy B, Petitjean S, Ray N, Maillot J. Least squares conformal maps for automatic texture atlas generation. *ACM Transactions on Graphics* 2002; **21**(3):362–371.
10. Remacle JF, Geuzaine C, Compère G, Marchandise E. High quality surface remeshing using harmonic maps. *International Journal for Numerical Methods in Engineering* 2010; **83**(403-425).
11. Marchandise E, de Wiart C, Vos W, Geuzaine C, Remacle J. High-quality surface remeshing using harmonic maps–part ii: Surfaces with high genus and of large aspect ratio. *International Journal for Numerical Methods in Engineering* 2011; **86**:1303–1321.
12. Marchandise E, Compère G, Willemet M, Bricteux G, Geuzaine C, Remacle J. Quality meshing based on stl triangulations for biomedical simulations. *International Journal for Numerical Methods in Biomedical Engineering* 2010; **26**(7):876–889.
13. Bommes D, Zimmer H, Kobbelt L. Mixed-integer quadrangulation. *SIGGRAPH '09: ACM SIGGRAPH 2009 papers*, ACM: New York, NY, USA, 2009; 1–10, doi:http://doi.acm.org/10.1145/1576246.1531383.
14. Frey P, George PL. *Mesh Generation - Application To Finite Elements*. Wiley, 2008.
15. Geuzaine C, Remacle JF. Gmsh: a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities. *International Journal for Numerical Methods in Engineering* 2009; **79**(11):1309–1331.
16. Lewis RW, Zheng Y, S UA. Aspect of adaptive mesh generation based on domain decomposition and delaynay triangulation. *Finite elements in analysis and design* 1997; **20**(47-70).
17. Tchon K, Camarero R. Quad-dominant mesh adaptation using specialized simplicial optimization. *Proceedings of the 15th International Meshing Roundtable*, Springer, 2006; 21–38.
18. Lloyd S. Least squares quantization in pcm. *Information Theory, IEEE Transactions on* 1982; **28**(2):129–137.
19. Carrier-Baudouin T, Remacle JF, Marchandise E, Lambrechts J. l_p lloyd's energy minimization for quadrilateral surface mesh generation. *Submitted to the 20th International Meshing Roundtable*, 2011.

Acknowledgements

This work has been partially supported by the Belgian Walloon Region under WIST grants ONELAB 1017086 and DOMHEX 1017074.