Chapter 22

P-methods and hp-methods

Introduction

P-methods and hp-methods represent solutions to mesh adaptation which provide, in some sense, an alternative to h-methods. In brief, a p-method, in contrast to a h-method (Chapter 21), consists in varying the degree p in the approximation while keeping the mesh sizes h unchanged. In this way, the quality (the richness) of the approximation is adapted to the way in which the solution varies.

Well established for some model problems, particularly when the geometry of the domain is relatively simple, p-methods are somewhat delicate to implement when the geometry is really complex. The fact that the size is constant while the degree of the approximation is the sole parameter results in a strong constraint which is not simple to overcome. Therefore, hp-methods have been introduced which combine a p-adaptation with a h-adaptation and thus offer the advantages of both methods while avoiding the unflexibility mentioned above.

★
★ ★

This chapter is organized in two distinct non symmetric parts. The first part concerns the construction of finite elements other than straight (linear) elements (P^1 type elements) which are naturally obtained by using automatic mesh generation methods. The second part discusses p and hp-methods, mainly with regard to an adaptive process.

This last aspect will only be dealt with briefly. In contrast, we discuss in greater detail some possible approaches for the construction of finite elements other than P^1, which are used in p-methods (where p varies) and also in all calculus with finite elements with a (constant) degree p other than 1. This type of construction, mostly seen in the case of P^2 triangles leads us firstly to consider how to mesh a curve (assumed to be planar for the sake of simplicity) by means of parabola arcs. Then, we turn to the creation of P^2 elements using this curve mesh as input data. In this context, we present several approaches which essentially involve adequate post-processing a P^1 type mesh. As will be seen, various technical difficulties can be expected. Some examples of such problems are given in three dimensions.

Let us recall that we are mostly concerned with the impact of the solution methods on mesh generation techniques. Therefore, we mainly focus on this particular aspect and less on using p-methods for solution control from an adaptive point of view.

22.1 P^2 mesh

We consider here a mesh where the elements are P^2. For the sake of simplicity, we limit ourselves to a problem in two dimensions. First, we discuss how to mesh a curve (a domain boundary in this case). In other words, our concern is to show how to obtain a polygonal approximation where the elements (the segments) are parabola arcs. Then, we show how to construct P^2 elements.

22.1.1 Meshing a curve (review)

Let us consider a parametric curve Γ whose equation is $\gamma(s)$, where s is the curvilinear abscissa. In addition, we assume that the curve has the required regularity when needed. In Chapter 14, we saw how to discretize such a curve by means of line segments (a P^1 mesh), in such a way as to control the gap between the curve and its discretization. Now, we return to the same problem in the case of a P^2 discretization, say one composed of parabola arcs.

Local behavior of a curve. We give again, with the same notations, the expansion of Relation (14.4), say, in some vicinity of $\gamma(s_0)$ and for small enough Δs :

$$\gamma(s) = \gamma(s_0) + \Delta s\, \vec{\tau} + \frac{\Delta s^2}{2\,\rho(s_0)}\, \vec{\nu} - \frac{\Delta s^3}{6\,\rho(s_0)^2}(\rho'(s_0)\, \vec{\nu} + \vec{\tau}) + \dots .$$

The curve can be approximated using the first three terms in its expansion at ε, if we take $\Delta s = \alpha\, \rho(s_0)$ such that (Relationship (14.7)) :

$$\alpha \approx \frac{\sqrt{6\,\varepsilon}}{\sqrt[4]{1 + \rho'(s_0)^2}} .$$

Also in Chapter 14, we saw that replacing the parabola for approximation (of the expansion) by means of a line segment discretization leads to a gap between this discretization and the curve in the order of :

$$\delta \approx \left(\frac{\varepsilon \sqrt{6\,\varepsilon}}{\sqrt[4]{1 + \rho'(s_0)^2}} + \frac{3\,\varepsilon}{4\,\sqrt{1 + \rho'(s_0)^2}} \right) \rho(s_0) .$$

In this estimate, the first term controls the gap related to the approximation by the expansion while the second term controls the discretization gap.

An approximation of type P^2 consists of discretizing the parabola[1] by means of of parabola arcs, thus by itself. The mesh element in the discretization is then the portion of parabola passing through the three points $\gamma(s_0)$, $\gamma(s_0 + \frac{\Delta s}{2})$ and $\gamma(s_0 + \Delta s)$. The above relation is then :

$$\delta \approx \left(\frac{\varepsilon \sqrt{6\,\varepsilon}}{\sqrt[4]{1 + \rho'(s_0)^2}} + 0 \right) \rho(s_0)$$

and therefore the total gap has the order of the first terms (as the gap in discretization is now null). This gap is then :

$$\frac{\varepsilon \sqrt{6\,\varepsilon}}{\sqrt[4]{1 + \rho'(s_0)^2}}\, \rho(s_0) .$$

Thus, for a given relative tolerance of ε, the result obtained in this way is much more precise than in the case of a P^1 approximation. As a consequence, the analysis may be based on a coarser approximation of the expansion. Indeed, one can look for a value of ε' such that :

$$\frac{\varepsilon' \sqrt{6\,\varepsilon'}}{\sqrt[4]{1 + \rho'(s_0)^2}} \approx \frac{\varepsilon}{\sqrt[4]{1 + \rho'(s_0)^2}} ,$$

which means :

$$\varepsilon' \sqrt{6\,\varepsilon'} \approx \varepsilon .$$

An easy computation gives :

$$\varepsilon' \approx \sqrt[3]{\frac{\varepsilon^2}{6}} . \tag{22.1}$$

To give an idea, a tolerance value $\varepsilon = 0.01$ leads to $\varepsilon' = 0.025$ while $\varepsilon = 0.001$ leads to $\varepsilon' = 0.0055$. Therefore, one can take a larger tolerance value while obtaining a nice control. In the case of a circle, for $\varepsilon = 0.025$ we find $\alpha = 0.391$, this shows that about 16 mesh entities are necessary to obtain a control in 0.01 (in fact, somewhat better), while in comparison, such a control in P^1 requires using 26 mesh elements (Chapter 14).

Exercise 22.1 *Let us consider the circle of radius ρ (centered at the origin, for simplicity). Discretize it by means of* 16 *P^2 mesh elements such that each of these parabola arcs has its endpoints A and B and its midpoint M in the circle. Then, compute :*

- *the gap between the chord AB and the circle,*
- *the gap between the chord AM and the circle,*
- *the gap between the parabola arc and the circle (thus, this value at the midpoint of the mid-arc, the arc related to AM).*

[1] One could conceive a method where the expansion is pushed one order higher and construct as a discretization parabola arcs lying on the curve defined by the first four terms in the expansion. In this case, it would be necessary to return to the entire discussion, as in Chapter 14.

Conclusions ? (Hints : find the value α associated with the mesh element AB, deduce the value of ε giving the gap as a ratio of ρ, repeat this for the mesh entity AM and do the explicit calculus for the arc). Observe what is obtained by changing the number of mesh entities, for example, for a number like 8 or 32.

Remark 22.1 *In terms of continuity, the P^2 discretization thus-obtained is rather good.*

Remark 22.2 *The above reasoning analyzes the behavior of the curve using an expansion between s_0 and $s_0 + \Delta s$. It is clear that a similar study can be made between s_0 and $s_0 - \Delta s$, thus leading to the same conclusion about the resulting gap. As a consequence, for the same gap, it is possible to create a mesh entity twice as large ($2\,\alpha\,\rho(s_0)$) by centering it at the point $\gamma(s_0)$. As previously seen, the accuracy remains in ε. Nevertheless, the continuity from mesh entity to mesh entity is slightly modified.*

Geometric mesh of a curve. By analogy with Definition (14.1), a P^2 geometric mesh can be defined as follows :

Definition 22.1 *A P^2 type* geometric mesh *at a given ε of a curve Γ is a parabolic piecewise discretization of this curve whose relative gap is in the order of ε at any point. If Δs is the length of an arc of the curve and if h is the length of the corresponding parabola arc, then, h tends towards Δs with Δs.*

Hence, we return to the same definition as for P^1 meshes and thus the same characterization of this property.

As seen for Definition (14.1), notice that this is a reasonable definition, *i.e.*, related to the difference in lengths or, similarly, related to the relative gap to the curve. This definition is only one possible formulation of the notion of a P^2 geometric mesh. Indeed, another definition may be, for example, to observe the value if the surface area comprised between the arc of curve the the arc of parabola which approaches this curve.

After the previous discussion and with regard to the proposed definition, constructing a geometric mesh of a smooth curve is made by computing the length of the curve using the metric of the main radii of curvature weighted by adequate coefficients α. As a corollary, meshing a curve leads to :

- finding the inflection points in this curve together with the singular points (corners),
- imposing these points as mesh vertices,
- splitting the curve into parts, each of which is bounded by to such consecutive points,
- meshing each part by using the previous principle.

Remark 22.3 *Identifying the points with a maximal curvature and prescribing them as arc midpoints enables us to have a nice regularity in these points. Nevertheless, a reasonable result is also obtained when the points of inflection are not explicitly considered.*

As a conclusion, we have briefly described a method that allows the analysis of P^2 meshes for curve meshing (in $\mathbb{R}^2$). This analysis can be followed to give idea about how to conceive a mesh generation method. Here is only one of the possible approaches and clearly other methods may be envisaged. Moreover, the actual implementation of the above principles is not necessarily obvious. Nevertheless, the remarks given in Chapter 14 may serve as a source of inspiration.

Meshing a curve. For this problem, the mesh construction of a curve with or without a metric map (not necessarily of a geometric nature), we refer the reader to Chapter 14. When no metric specifications are known, a nice variation in size for two neighboring mesh elements is a natural target. On the other hand, when a specification of a physical nature (*i.e.*, related to the physics of the problem in question) is given, it is important to make sure that the sizes specified by the physics remain compatible with those related to the geometry.

22.1.2 P^2 finite elements

For the sake of simplicity, we consider a planar case (in two dimensions) and we consider the construction of P^2 type triangles. We assume the domain to be meshed to have a curve (a series or a number of curve parts) as a boundary. The question is then to define, specifically for the triangles having at least one boundary edge, the nodes of these triangles together with their edges (other than the boundary edge(s)). In terms of mesh construction method and *a priori*, there are two ways of addressing this type of problem :

- a type P^1 mesh with step h_1 (see the remark below) is available which is then transformed in a P^2 type mesh with step h_2 where $h_2 = h_1$;
- a type P^2 mesh is constructed directly. This approach initially requires meshing the domain boundaries by means of P^2 mesh elements, as above, and then creating the P^2 triangles which form the final mesh of the domain.

The second approach implicitly implies the construction method to be P^2. The problem is no longer, for a given edge, to find the point suitable for the construction of a P^1 triangle (thus with a purely geometric nature) but to deduce from a P^2 edge, the three points and the two other edges required to define a P^2 triangle.

Keeping in mind the discussions about the main mesh generation methods, it is then clear that the corresponding algorithms (*quadtree-octree*, advancing-front or Delaunay, for example) are in essence purely of a geometric nature. Developing methods directly resulting in P^2 type elements is then probably possible but presumably not so obvious.

Following these remarks, the first approach is more reasonable in practice. Thus we ask the mesh generation method to produce a geometric mesh (thus a P^1 mesh) and then, this mesh is modified into a P^2 type mesh by means of post-processing.

This way of processing merits some comments, the first regarding the sizes of the mesh elements that are to be taken into account.

Remark 22.4 *Let h_1 be the size of a P^1 mesh element and let h_2 be that of a P^2 mesh entity, then fixing $h_2 = h_1$ implies that the number of P^2 nodes (the element vertices and the edge midpoints) is the same as the number of P^1 nodes (the element vertices) in the P^1 mesh obtained by subdividing into four the P^2 triangles. Following what we saw about the geometric approximation of a curved boundary, it is then possible to take, for a similar accuracy, a step h_2 twice as large as step h_1. Therefore, in this approach, one can take, when constructing the P^1 mesh, a relatively large step and, in particular, start from a mesh which is a relatively coarse approximation of the boundary geometries.*

Doing so, a P^1 mesh is available whose steps may be large. It is easy to see, for a boundary edge, that if the difference with the geometry is relatively small, there is no major difficulty in finding the mid-node required to transform this segment into a parabola arc. After which, in such a case, the difficulty when constructing the other edges is related to the local configurations.

A second remark must be made about this approach based on the modification of a P^1 mesh so as to complete a P^2 mesh. The local aspect of the approach implies that it is not strictly identical to a curve mesh construction method.

Remark 22.5 *Processing by means of transformation considers the boundary edges element by element while the curve meshing problem considers the entire curve. Therefore, the processing by element approach does not* a priori *see what happens in some vicinity of the element in treatment. This may result in a possible lack of smoothness of the curve thus-obtained. To overcome this trouble, it is necessary to have access to some global information such as tangents, normals, etc. Moreover, the two endpoints of the edge under examination are prescribed while these points are not necessarily optimal with regard to the meshing problem applied to the entire boundary curve.*

Despite this, up to now, it remains realistic to follow the local approach. Indeed, if the useful geometric information is known at each point (by means of queries to a geometric modeler or to a mesh assumed to give a suitable geometric definition, the *geometric support*, to some extent, it is possible to give a global aspect to the local process. The only negative feature nevertheless remains the constraint regarding the two edge endpoints that are imposed at the extremities of the arc of the curve to be constructed.

Nevertheless, in the following, we follow this principle, *i.e.*, the construction of P^2 elements by local transformation of P^1 elements. But, before going further, we make some comments on the second approach, *i.e.*, what a P^2 type mesh generation method could be.

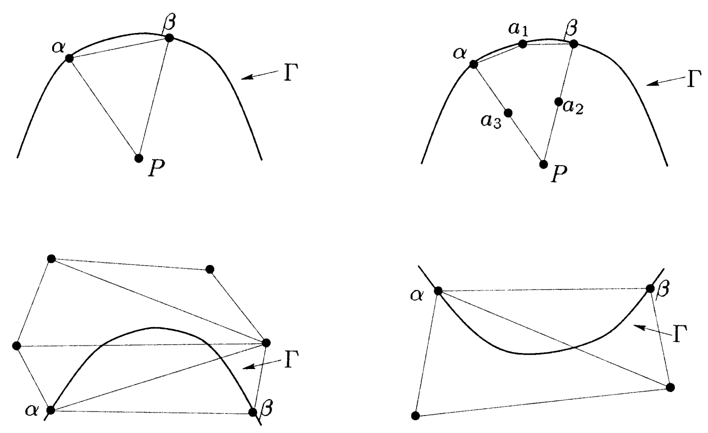

Figure 22.1: *Construction of a P^2 finite element. Top, the P^1 element as constructed by using a mesher is such that the construction of the corresponding P^2 element is "immediate". Note that the gap in P^1 is relatively poor while the gap in P^2 is rather better. Bottom, the constructed elements must be processed by a more sophisticated process if we wish to transform them into P^2 elements.*

Remarks about direct construction of a P^2 mesh. For simplicity, we focus on a triangular mesh and we assume that the boundaries of the domain to be meshed have been previously discretized using the curve meshing method described above.

Therefore, the vertices of the boundary meshing entities are assumed to be adequately located. In other words, the domain boundaries are suitably approximated.

The problem in question is then the direct construction of P^2 triangles using this discrete data input as information. The immediate question is the following :

"What is the pertinent quality criterion that must be used to govern the P^2 mesh construction ?"

For "classical" mesh generation methods (from Chapter 4 to Chapter 8), while limiting ourselves to a isotropic situation, we know that the optimality criterion is the fact that the triangles are equilateral. What becomes this criterion in the present case ? Before giving some ideas about this, let us mention that the literature is rather weak on this point. Since abstract results (about convergence and error estimates) make use of the fact that parameter h, the element size, tends towards zero, they fail to give particular indications about realistic cases where this value is not necessarily small.

It seems evident (and intuitive) that if the gap between the boundary edge (linear approximation of the boundary curve) and the boundary itself is small enough, then the reasoning used in P^1 can be followed and the optimality criterion remains unchanged. The case of interest is when this gap, for a linear approximation, is relatively large and when this gap, in P^2, is judged good enough (the geometry is suitably approximated). This implies that the optimal point related to the P^1

edge which allows the creation of an equilateral P^1 triangle is not necessarily optimal when a P^2 triangle must be defined. This depends on the curvature of the boundary in the region of interest. In other words, given a curved edge suitably meshed is not sufficient, some other conditions must be considered.

By analogy with the equilaterality criterion held in P^1, we can say that a P^2 triangle is satisfactory if the angle bounded by the two tangents in its three vertices is close to 60 degrees. If the edge is a curved edge, the tangent in question is the tangent to the curve, if the edge is a line segment, the tangent is the edge itself.

It is easy to see that this criterion results in a restriction on the length of the curved edge and makes it possible to find a reasonable position for the third vertex whatever the local concavity may be.

This being satisfied, the vertex we are seeking can be located at the intersection of the two line segments which form the above angle with the tangents to the curve. Then, the two other edges are defined using these line segments. Either the edge in construction belongs to the adequate line segment or it must be curved and this line segment is only its tangent. In such a case, it is necessary to construct a portion of a parabola using as the input data one of its endpoints, the tangent at this point and the length.

The example in Figure 22.2 shows, in comparison with a classical P^1 mesh (left-hand side) the P^2 mesh obtained by a direct construction method (see the curved boundaries).

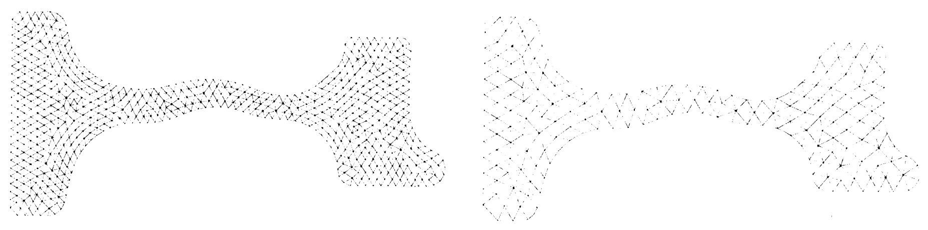

Figure 22.2: *P^1 and P^2 planar meshes. On the left, a P^1 triangular mesh with a uniform element size and, on the right, the P^2 mesh resulting from a direct construction with twice the element size.*

22.2 P-compatibility

The above discussion and the simple examples in Figure 22.1 allow us to make the notion of p-compatibility more precise. In the following and for the sake of simplicity, we only consider the case where $p = 2$.

Definition 22.2 *A geometric element (a P^1 element) is said to be* 2-compatible *if the distance between its boundary edge(s) and the boundary curve is such that there exists a point in the boundary for each of this(these) edge(s) that makes it possible to construct a valid P^2 element by a simple post-processing of the given P^1 element.*

This is a naive definition (somehow a tautology) which indicates that the simplest construction (find a point on the curve and construct a portion of parabola joining the two edge endpoints and passing through this point) results in a valid element. This means that there is no interference in the various edges (curved or straight sided). This implicitly implies that the new points (the "midpoints") remain close to the initial straight-sided segment and thus don't create any problem for the elements neighboring the element under consideration.

Note that the theory about error estimates and convergence issues (Chapter 20), deliberately considers a situation of this type. From a practical point of view, a 2-compatible (straight sided) element is easy to transform into an element of degree 2. Nevertheless, this does not mean that an element not in this case cannot be transformed into an element of degree 2 (as will be seen below).

22.2.1 P-compatibility *a priori*

Such a case is ideal. The mesh generation method completes P^1 elements which are 2-compatible. Then, it is sufficient to find the edge "midpoint" and to define the corresponding parabola arc. If the straight edge - boundary edge distance is small and if the curve is close to a parabola (or enjoys a certain symmetry with respect to the perpendicular bisector of the straight segment), the point we are seeking can be simply chosen as the projection of the edge midpoint onto the boundary.

In the case where the boundary curve does not satisfy this property regarding regularity and symmetry, the sought point *is not* the projection of the edge midpoint (return to the section about how to mesh a curve by means of parabola arcs while minimizing the gap).

Keeping this in mind, how can we ensure that a P^1 mesh satisfies *a priori* all the underlying requisites ? The simplest idea is to construct a mesh whose elements having a boundary edge(s) are such that this(these) edge(s) is(are) close to the boundary and, moreover, are such that the third vertex leads to an element which is almost equilateral. This is exactly the objective of classical mesh generation methods. Most of the elements must be equilateral. The immediate question that occurs is to know if any geometry can be approached with such triangles ? The answer is probably yes when a fine enough mesh is used, and probably no if not.

Remark 22.6 *Imposing a rather small size so as to obtain such a mesh is, as previously seen, not strictly necessary. The resulting P^2 mesh is, in general, too fine and thus dealing with such a mesh requires an unnecessarily high cost.*

Following these remarks, it is possible to replace the criterion about equilaterality by a similar criterion related to the angles formed by the tangents to the edges. Thus, given a boundary edge (a straight sided segment as we are in P^1), its angle with the next edge in the triangle is measured by taking into account the underlying boundary curve. Note that this criterion is not one of the criteria usually considered in classical mesh generation methods.

In conclusion, it is far from certain that this approach is the most suitable in all cases. In the following section, we propose another way to address this problem.

22.2.2 P-compatibility *a posteriori*

We consider here that we have an arbitrary P^1 mesh and we want to transform it into a P^2 mesh. The simplest idea consists in returning to the previous case. All edges (all situations) leading to a difficulty in the desired construction are modified so to suppress this difficulty. The possible modifications (here in two dimensions) that can be used are those described in Chapter 18. Thus they are based on an adequate combination of tools for mesh optimization such as point relocations and edge swaps.

The local configuration (an element and its neighboring elements with regard to the boundary edge under examination) to be dealt with is analyzed. Several situations may be found :

- the point in the curved edge to be added remains close enough to the corresponding straight sided edge and falls within the triangle having this edge or, again, falls outside this triangle (based on the local concavity) but has no influence with any of the neighboring elements,
- this point falls very close to one of the other edges in the triangle dealt with or again falls within a triangle other than the current triangle (one of its neighbors or, more delicate, an element that is not directly a neighbor of the triangle under examination).

In the first case, constructing a P^2 element is obvious. In the second case, we propose modifying the local neighborhood so as to return to a standard situation while observing that the present situation is related to the fact that a P^1 mesh too coarsely "violates" the geometry. Figure 22.3 depicts three examples of such situations.

As concerns how the local context must be modified so as to remove the difficulty during the construction, first, notice that in case i) and, to some extent, in case ii) an operation like an *edge swap* is not useful. In the first case, the resulting situation is similar to the initial configuration. In the second case, the new situation leads to having the node to be created very close to one of the other edges in the triangle (which results in a bad quality P^2 element or even a negative Jacobian for a very close configuration). A small change in the node position (by means of a moving point process) is then one possible way to move the new node away (by increasing the distance between this node and the edge). For the configuration shown in case iii), it is also possible to find a local strategy that suppresses the problem. First, it must be observed that using a local modification (point relocation or edge swap) does not allow us to remove the parasite point located outside the P^2 domain while being inside the P^1 domain. Thus, we propose (Figure 22.4), after [Dey-1997], :

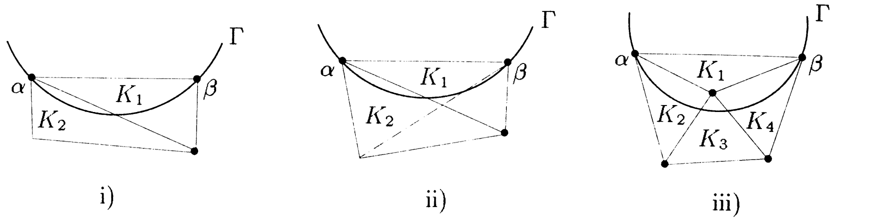

Figure 22.3: *Three local configurations where the transformation of a P^1 mesh into a P^2 mesh is not immediate because the geometrical approximation is too poor. In i) and ii), the point to be created falls outside the initial triangle, in iii) the situation is far more dramatic, the point to be defined is outside the triangle and, moreover, there are several "parasite" triangles (and even a point) which certainly impede the process.*

- constructing the desired node M,
- constraining the two edges αM and βM in the P^1 mesh,
- classifying the resulting triangles according to their positions with respect to boundary Γ and classifying the mesh entities (points and edges) in internal entities and boundary entities. This classification is made using the inheritance based on the classification of the entities in the initial mesh while preserving a coherent result,
- suppressing the exterior triangles (in this way, the parasite point disappears),
- analyzing the new context again.

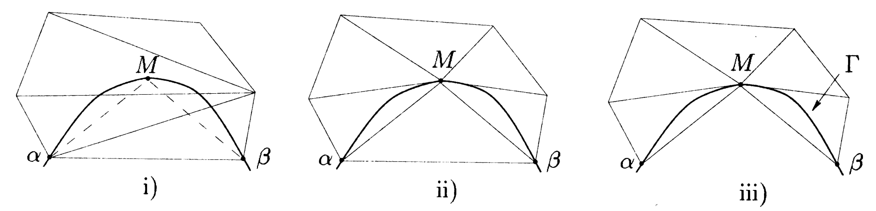

Figure 22.4: *The initial configuration and the predicted point M. This point is located in an element other than the triangle supposed to be a boundary element. The two boundary edges formed after inserting M are enforced. Consequently, the mesh is modified and the resulting elements are classified so as to find the new triangles that have a boundary edge.*

Notice that we encounter here an edge enforcement problem (as described in Chapter 7) and that the proposed method leads to what is expected while the resulting situation is not optimal. Indeed, if iterating this process, it could be necessary to subdivide the presumed boundary edges several times and, as a consequence, obtain an unnecessarily fine P^2 mesh. Note also that applying this heuristic is not trivial in three dimensions (as also seen in Chapter 7).

An example of a temporary pathology. When there are thick regions or when several boundary portions are close to one another (Figure 22.5), which are frequent situations in realistic cases, a difficulty may arise, which may be only temporary (this is not known *a priori*).

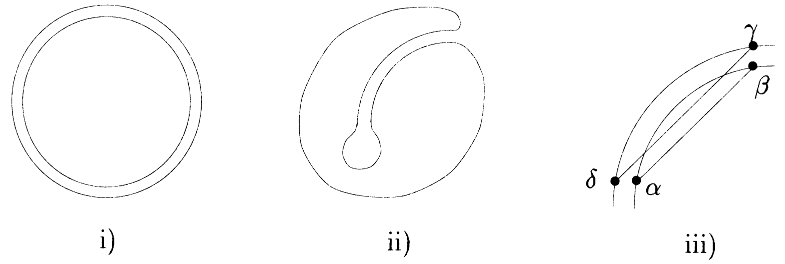

Figure 22.5: *Two local configurations where there is a thick region. The P^1 mesh is correct but, when constructing the P^2 mesh, auto-intersections at the boundary level are created, which may only be temporary. In fact, dealing with edge $\alpha\beta$ leads to an intersection with edge $\delta\gamma$ whereas, if this edge is processed first, dealing with the other edge becomes possible.*

As the boundary is assumed to be correct (not self-intersecting), there exists necessarily a suitable geometric mesh of this boundary. The problem is then to find the value of the threshold ε which ensures an accurate approximation and avoids self-intersections. The difficulty in finding the proper value of this threshold results from the fact that the points or the elements which are geometrically close to each other are not necessarily close in terms of the topology. One way to detect such a situation is to make use of a global structure like a *neighborhood space* Chapter 1) previously constructed in a proper way. Indeed, in this structure, it is demanded to store the mesh entities so as to, for a given entity, retrieve at a low cost the set of mesh entities falling in some neighborhood.

Given such a tool, the data structure is queried when constructing an element and the possible conflicts are detected. The strategy that may be used is then of a purely heuristic nature. Indeed, it is possible :

- to accept a temporary collapsing while maintaining the list of the entities concerned, prior to re-processing (*i.e.*, subdividing) these entities. As the solution exists, it will be found (at least, we hope so);
- not to construct such a situation (which means we temporarily ignore the entity concerned) in the hope that the pathology will disappear when dealing with some other entities.

As a conclusion, obtaining p-compatibility *a priori* or *a posteriori* makes sense only in cases which are already relatively close to the solutions and, in practice, the interesting cases are precisely those which are pathologic and, due to this, are bad candidates for such processing. Also, we have seen that looking for proper compatibility may lead (after some heuristics) to a valid result but one which is

probably not optimal (in terms of the number of mesh entities). Therefore, other methods must be investigated (see below).

22.3 Construction of P^2 elements

We first restrict ourselves to a planar situation.

22.3.1 Element 2-compatible

Following the definition of the notion of compatibility, the transformation of a P^1 mesh into a P^2 mesh reduces to locally modifying the initial mesh, *i.e.*, element by element.

22.3.2 Other elements

In such a case, a purely local processing, since the processing only of the boundary edge is not sufficient, does not lead to a solution. In particular, it is necessary, for "curving" the boundary edge under treatment, to curve one or several other presumably internal edges. Intuitively, this consists in "defining" the empty region necessary to obtain a suitable construction. In practice, this approach can be seen as an optimization method in which a certain propagation (a deformation between the neighboring entities) from entity to entity is necessary so as to achieve the desired result.

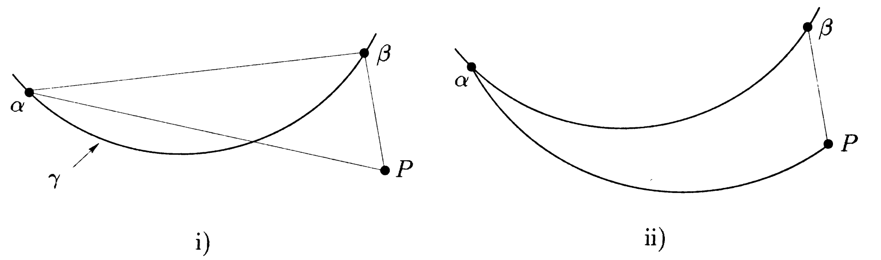

Figure 22.6: *A local configuration where it is necessary to curve an* a priori *straight-sided edge so as to have enough space for the boundary edge which is to be processed.*

This meshing problem is a problem with constraints (Figure 22.6). Edge $\alpha\beta$ (parabola arc) having been constructed, it is necessary to construct a curved edge,αP, as far as possible (in accordance with the local configuration) so as to obtain a P^2 element of reasonable quality. It should be noted that this mesh deformation may lead to processing some other edges in the neighboring elements. Here, we can also use the tangents to the relevant edges.

22.3.3 Dealing with a surface element

We turn now to the same problem for a surface mesh. The targeted application is clearly the construction of surface meshes but also the construction of the "curved" faces in three-dimensional meshes of type P^2.

Following an observation already given, meshing a surface or transforming a (planar) triangular face into a curved triangular face (belonging to the same surface) are not, *stricto sensu*, identical problems. In the first case, the surface is globally considered, in the second case, the process is a local one.

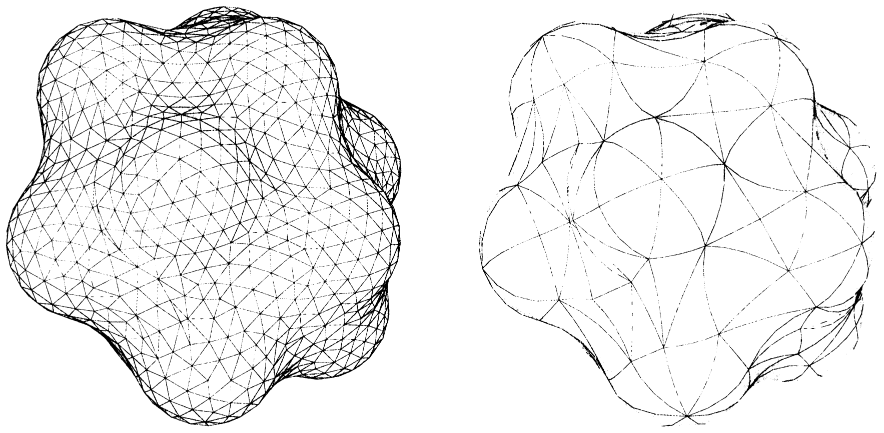

Figure 22.7: *Anisotropic meshes of a molecule of* cyclohexane. *Left-hand side, mesh with P^1 triangles controlled by an angular tolerance of 8^o, right-hand side, a P^2 mesh for a tolerance of 32^o.*

Meshing a surface by means of P^2 triangles. In this case, the principles of mesh construction described in Chapter 15 may be used. The method involves finding the (curved) edges of the triangles that ensure that the thus-created surface mesh is a suitable approximation of the real surface. We have seen that this problem does not reduce to ensuring that the curved edges are adequately close to the surface. Indeed, controlling the gap between such an edge and the surface does not automatically imply that a triangle with these edges as sides is adequately close to the surface, for the same given threshold value. In addition, the interface between two adjacent triangles must have, in some cases, a certain extent of smoothness thus implying, at least, that the edge discretization must not only take into account the gaps (in terms of distance) to the surface but must also consider a control regarding the gaps related to the variation of the tangent planes or even consider some other geometric properties (variation in the curvature, radii of curvature, etc.).

As for a planar P^2 triangle, one must control the size and the direction of the edges using the tangent plane at each vertex (the effect of the tangent in a planar case is now replaced by these various tangent planes).

An example of a P^2 surface mesh is depicted in Figure 22.7, right-hand side (software BLSURF).

Transforming a P^1 surface mesh into a P^2 mesh. In such a case, the geometry is assumed to be known (via a modeler, a mathematical description or a representative mesh) and, in addition, that a P^1 mesh is available. The question is then to transform this mesh in a P^2 mesh.

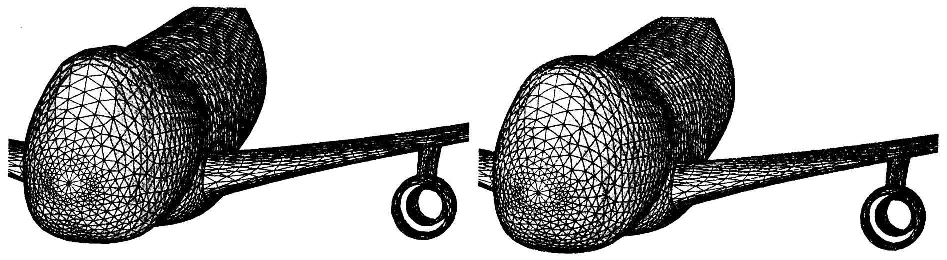

Figure 22.8: P^1 *and* P^2 *meshes of a Boeing 747. The mesh on the left results from a mesh simplification at 85% of a very fine surface mesh (original data provided by Boeing). The mesh on the right is the transformation of the* P^1 *triangles into* P^2 *triangles.*

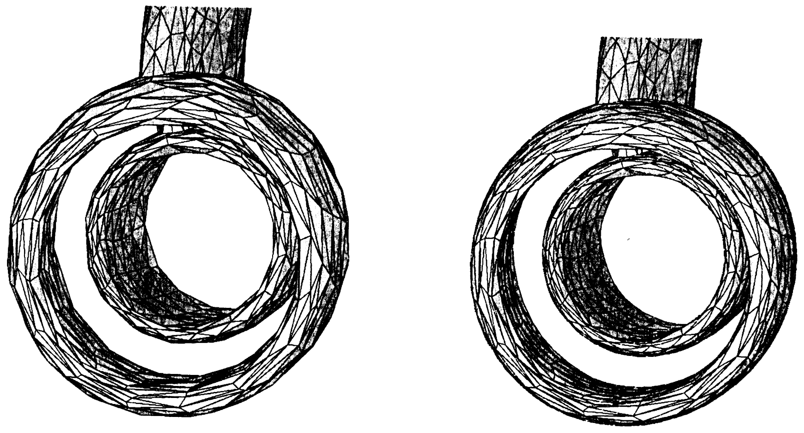

Figure 22.9: *Enlargement of part of the* P^1 *and* P^2 *meshes of the previous figure. It is easy to see the quality of the geometric approximation of the surface in the* P^2 *mesh as compared with the linear approximation seen in the* P^1 *mesh.*

We return to the above discussion. If the P^1 mesh follows the right properties (control by the tangent planes) then the transformation is of the same nature

The construction is indeed relatively easy to do. The examples in Figures 22.8 and 22.9 are courtesy of H. Borouchaki. Otherwise, it may be necessary to refine the P^1 mesh based on the local curvatures so as to obtain a mesh that is easy to transform.

22.3.4 Volumic case, example of the P^2 tetrahedron

We now give some indications on the construction of tetrahedra of degree 2. We only focus on elements having one or several entities (edge or face) lying on the boundary. Obviously, the most interesting case is when this portion of the boundary is a curved region.

Element 2-compatible. As in two dimensions, a 2-compatible tetrahedron is relatively easy to transform into a curved tetrahedron with 10 nodes (see Chapter 20). The boundary face is curved (let us assume that there is only one such face) and, for the position of the three nodes to be defined, we return to a curve meshing problem, now in three dimensions. Note that meshing these curves, edges shared by several elements, must be such that the curved face and its neighboring faces form a discretization as regular as possible of the real surface (when this latter is regular). In other words, meshing a curve in this context requires considering the underlying surface.

It must be noticed that obtaining a 2-compatible mesh for a straight sided tet mesh forming a not necessarily close approximation of the real geometry is a non trivial problem and, in particular, may lead to unnecessarily small elements. On the other hand, the technique based on edge or face enforcement, local modification of the resulting mesh and classification of the new mesh entities (vertices, edges and faces) with respect to the boundary is in this case rather delicate (Chapter 7). The alternative approach, based on the control via the various tangent planes to the faces gives better results (in specific, it is not necessary to refine too much the initial mesh and thus the number of elements remains reduced while the quality of the approximation is still within the desired range).

Other elements. An element (resulting from an automatic mesh generation method) which is a coarse approximation of the real geometry may, as in two dimensions, not be suitable for such an immediate construction. In particular, this may lead to curving some face *a priori* not in the boundary so as to obtain a region which is large enough to allow the construction of an element of adequate quality.

22.4 Elements of higher degree

The p-methods basically lead to modifying the degree p of the underlying polynomials until obtaining some values rather larger than 2. In this respect, we find some examples where the value of p could be 5, 6, or more. Therefore, for example if $p = 3, 4, ...$, it is of interest to discuss, for the curves, how to construct arcs of

cubics, quartics, ... and, for the faces, how to construct some surfaces with the corresponding degree.

Let us just mention a few remarks about the case of a P^3 mesh entity, an arc of cubic, for a curve meshing case in the plane. We start again from a limited expansion until the term of degree 3. Then, assuming adequate hypotheses about Δs, the error due to stopping this expansion at this order can be appreciated by looking at the following term (at the order 4). This being done, we can follow the method described in Chapter 14 and we construct an arc of cubic so as to approximate the curve defined by the three first terms in the expansion, namely :

$$\Delta s\,\vec{\tau} + \frac{\Delta s^2}{2\,\rho(s_0)}\,\vec{\nu} - \frac{\Delta s^3}{6\,\rho(s_0)^2}(\rho'(s_0)\,\vec{\nu} + \vec{\tau})\,.$$

For a P^3 triangle, we find a construction similar to that used for a P^2 triangle while now controlling the tangents at the endpoints of the edges of the elements. In particular, one condition that must be verified is to ensure that the arc of cubic, the P^3 edge which approximates the curve, retains the same concavity between these two endpoints (thus implying a restriction on the length in the case where this concavity changes).

As a conclusion about these ideas of methods able to construct elements of an order higher than 1, it is clear that this topic is still a topic of interest which remains to be really well addressed (apart from the case where the curves or the surfaces remain smooth enough).

22.5 About p-methods and hp-methods

In this short section, we briefly return to mesh adaptation methods based on a p or a hp approach.

22.5.1 P and hp-methods

Essentially, a p-method allows, according to a criterion (resulting from an adequate error estimate), us to predict the degree of the approximation which is appropriate with regard to the way in which the solution varies. Note that this degree changes during the computational process and also may vary from one element to another in the mesh. The literature about p-methods is extensive and, in particular, one can refer to [Babuska,Guo-1988], [Babuska *et al.* 1989], [Babuska,Suri-1990], [Dey *et al.* 1997], (among many others).

Constructing a finite element of order p is not done as in the classical case. In other words, when p varies, we don't consider the set of finite elements (their basis polynomials) with the corresponding order p but a finite element which is hierarchically defined. This being assumed, changing the degree involves adding the contributions of some shape functions whose degree is less than or equal to the chosen value and ignoring the contributions related to higher degrees.

22.5.2 An adaptation scheme

A possible (reasonable) scheme for an adaptation loop in terms of h or p comprises several steps, as for an adaptation in h only (as seen in the previous chapter). First, an initial mesh is completed using a classical method, while trying to obtain a mesh that is reasonably suitable for further use while, at this time, no precise information can be used regarding the order or the size of the elements (an error estimate *a priori* or some knowledge about the physical behavior may nevertheless be helpful in this construction). An initial solution is then computed using this mesh as a spatial support and an error estimate *a posteriori* analyzes its quality so as to estimate the sizes (h) or the degrees (p) adequate for the elements in the mesh of the next iteration step. The adaptation process then consists in constructing this mesh by taking these requests into account. Note, as for an h-method, that this adaptive process may be a local or a global process (nevertheless, the available examples are mostly local).

The new spatial support then serves to compute a new solution and a new error estimation is made and, until a given stopping criterion has been achieved, the iteration steps are continued.

The actual implementation of such a process includes two aspects which are slightly different. First, if the size parameter is affected, we return to the discussion in the chapter devoted to h-methods which describes various approaches that allow the construction (or the modification) of a mesh while following a given size map. Then, if the size remains unchanged, changing the order in the solution method results in two new problems :

- the construction of the "curved" entities in the mesh that are affected by the current step,
- the construction of the finite elements with the desired order p.

The latter aspect is out of the scope of this book, thus we advise the reader to return to the brief comment previously given and to the *ad hoc* literature (including the above references), while the first aspect leads us back to the preliminary discussion given in this chapter. Note that only the elements having a boundary entity located in a portion of the boundary which is curved are delicate to handle and that this may lead to processing some other *a priori* internal elements, in some neighborhood.

Chapter 23

Parallel computing and meshing issues

Introduction

Parallel computing is a key-issue for various categories of numerical problems. In practice, a numerical simulation may require a very fine mesh (*i.e.*, one containing a large number of elements) and/or may be costly in terms of CPU time (when the computation is done in a serial way). Parallel computing[1] is an efficient solution for large size problems (*i.e.*, with a large number of unknowns) that are impossible to carry out using classical facilities due to size requirements and/or CPU cost. In fact, parallelism consists in spreading (distributing) the computational effort and/or the memory requirements over the different computers available.

The notion of a parallel computing process can be conceived at various levels but, here, we will mainly focus on two of these levels. Obviously, the computational stage is concerned with parallelism. In such cases, a preliminary stage consists in partitioning the domain by means of sub-domains prior to dispatching these to different processors (each of them taking charge of one sub-domain). On the other hand, it could be of interest to see what degree of parallelism could be required at the mesh generation step itself.

At the solution step, a domain decomposition method requires a partition of the domain into several sub-domains. In these, the mesh that must be constructed must have certain properties. The solution method makes use of communication processes from one sub-domain to the others. At the meshing step, a parallel computational process consists in partitioning the domain into different sub-meshes in order to distribute the computation effort over several processors, each of them being responsible for computing the solution for one sub-domain. The global solution is then achieved, when all the sub-solutions have been completed, by merging all of these partial solutions.

Constructing a suitable domain partition, as well as constructing each of the corresponding sub-meshes, can be achieved in many different ways. These approaches can be broadly classified into two categories. Either *a posteriori* or *a*

[1]For the sake of clarity, we are concerned here with a multi-processor architecture using a distributed memory system.

priori partitioning methods can be considered. *A posteriori* processing starts from a mesh of the entire domain while *a priori* processing uses the domain itself and not a mesh of it. Note that *a posteriori* methods have received considerable attention and are widely used in practice whereas *a priori* methods are still a field of intensive research (particularly in order to automate the partitioning process).

When concerned with the mesh construction aspect, it is not easy to decide where precisely the parallel aspect should be included. In particular, a question could be : is it necessary to include parallel concepts in the mesh generation method or should the construction method remain serial and used in parallel ? The following remark may throw some light on this. There exist at present various mesh generation methods which are reliable, fast and able to complete several million elements within only a few minutes. Therefore, one may think that constructing[2] several million or hundred million elements can be done in an indirect way by using a classical method (for constructing, say, one million elements) included in a parallel process. In other words, the global generation process is parallel but the construction method remains serial. This leads to distributing the computational effort over different processors and, this being done, sequentially considering one distinct task in one processor. Thus, following this approach, the parallel aspect acts at the global meshing level and not at the mesh generation method level.

Nevertheless there have been a number of papers on parallel mesh generation methods. The aim here is to construct the mesh in parallel, usually using a classical meshing method which has been modified in order to incorporate some degree of parallelism. Despite the above remark, it should be noted that although this type of approach is possible, its interest seems more theoretical or academic.

★
★ ★

This chapter is subdivided into three parts. It successively discusses the different issues mentioned in the above introduction. Hence, at first, we recall the main domain partitioning methods. Several *a posteriori* methods are presented and then some *a priori* methods are described. We then turn to a parallel meshing process and, in particular, we discuss how to conceive a parallel loop of computation. To end, we give some indications about the possibility of mesh generation method parallelization.

23.1 Partition of a domain

Constructing a partition of a given domain consists in subdividing it into several disjoint sub-domains whose union forms a covering-up of the entire domain. In fact, we are not so much concerned with the domain or the above sub-domains but more with the meshes of these domains. First, we give some general indications about the question under investigation, then we discuss the different partitioning methods in greater detail.

[2]If this is strictly necessary. In fact, the concern is not to necessarily construct a "large" mesh but more probably to have such a mesh available.

23.1.1 Overview of partitioning methods

Partitioning or mesh splitting methods fall into two categories. *A posteriori* methods split a fine mesh (*i.e.*, a large size mesh) into sub-meshes of a reduced size, and *a priori* methods construct the meshes related to sub-domains that are directly obtained (*i.e.*, without having to construct a global fine mesh, but directly using the domain geometry or even a rather coarse mesh, which is easy and fast to complete and requires little memory).

Partitioning *a posteriori*. In such approaches, we are given a mesh of the entire domain and the partition method involves splitting this mesh into sub-meshes which define the partition. Several types of methods allow such a task. For details, the reader is referred to the *ad-hoc* literature (see [Simon-1991] or [Farhat,Lesoinne-1993], among many others) and, in what follows, we simply outline some of these methods.

It is obvious that the weak point in such a direct way of addressing the problem is the problem of memory requirement. In fact, the required memory size is about the sum of the resources needed to store the complete mesh plus the resources required to store at least one of the sub-meshes. Moreover, classical problems inherent to the partition process must be considered (as they will be present whatever the chosen method). Among these, we find :

- the issue of *regularity* or *smoothness*. This mainly concerns the shape of the sub-meshes and thus includes the shape of the sub-mesh interfaces. A certain degree of smoothness is indeed required to guarantee a relative numerical accuracy (convergence) for the partial solutions with regard to the global solution. Indeed, some methods, such those using the Schur complement, lead to the solution of local problems coupled with a problem at the interface level. These methods require well conditioned local problems in order to guarantee the proper global convergence. Connexity or not of the sub-domains is therefore an important factor for this local convergence,

- the question of *interface size*. The size of the interface between two sub-domains directly influences the amount of data that must be exchanged and thus on the degree of saturation (bottleneck) in the network used to carry these messages. This size also affects the number of potentially redundant operations,

- concern about the number of connections from sub-domain to sub-domain,

- concern of balancing between the sub-meshes. The size of these meshes must be distributed in such a way as to balance the effort of each processor at the solution step,

- concern about the interface sizes (the number of nodes on these interfaces), etc.

Also, we do not consider the memory resources necessary to construct the initial mesh and those needed to store it (not to mention the CPU time required in the i/o).

Different classes of partition method are encountered. Among these, some are based on graph partitions and some are purely geometric methods directly based on mesh partitions. All these methods apply to finite element type meshes since a vicinity graph can be constructed based on the connections between the elements in a given mesh.

Before presenting some of the most frequently used algorithms, it is useful to consider the type of entities (vertices (nodes) or elements) on which the decomposition is applied. This point is of importance since it is related to the numerical software or the solution method that will use this decomposition. This choice may also have some effect on the splitting algorithms. Nevertheless, we do not go into more detail about the positive features or the weaknesses of these two approaches.

Element based decomposition. This is the most frequent case. The mesh is partitioned by distributing the elements among the sub-domains. In other words, one element is logically associated with one and only one sub-domain. One can see an example of such a decomposition in Figure 23.1, left-hand side. This partition has an interface consisting of vertices (nodes), edges and faces.

Node based decomposition. In this case, the mesh is partitioned by distributing its vertices among the sub-domains. In other words, one vertex is logically associated with one and only one sub-domain. One can see an example of such a decomposition in Figure 23.1, right-hand side. This partition has an interface consisting now of elements (nodes, faces and segments), which are shared by the sub-domains while maintaining the initial connections.

Before describing these methods in greater detail, it may be noticed that this approach is probably the worst way to go about mesh parallelism. Indeed, for a large-scale problem, the construction of an initial mesh of the entire domain[3] may even be impossible. Nevertheless, this type of method, when suitable, gives nice results and is probably the most pragmatic (in fact, simply the only possible) way to address the parallel problem. To end, note that the creation of the initial mesh does not take benefit from the assumed advantages of parallelism.

Partitioning *a priori*. To avoid this memory requirement problem, it is natural to examine the *a priori* approaches, when they exist and can actually be implemented. In this case, it is not necessary to construct a large mesh thus limiting the amount of memory available and, moreover, the parallel aspect directly appears in the construction, thus avoiding the main weakness of the previous methods.

Thus we first create a partition of the domain. This step may start from a reasonable mesh (*i.e.*, a relatively coarse mesh) of the entire domain. Several sub-domains are then identified and meshed, each on one processor. Thus parallelism

[3]In our experience, at this time, we are limited to 10 million tets constructed by a Delaunay type method, mainly due to the memory resources available in a classical workstation.

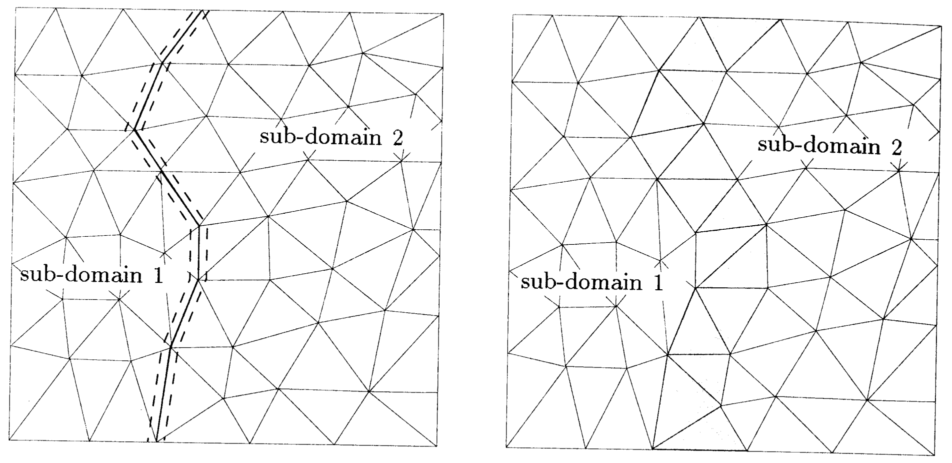

Figure 23.1: *Element based decomposition (EOD, element oriented decomposition), left-hand side and node based decomposition (VOD, vertex oriented decomposition), right-hand side.*

plays a part from the mesh generation step. Another strategy consists in extracting the various sub-domains (their boundaries) starting from a mesh of the boundary (the surface) of the entire domain. Then, the same method applies.

Actually, the balancing between the sub-domains together with the smoothness of the interfaces may be considered by using the information now available (the coarse mesh in the first case or the surface mesh in the other approach).

In the first strategy, there are two ways of obtaining a coarse mesh : either an empty mesh (*i.e.*, without any internal vertices) as defined in a Delaunay based method (Chapter 7) or a mesh with a limited number of internal vertices constructed by one or the other of the possible methods.

In the second strategy, the meshes are constructed using the meshes of the surfaces that constitute the interfaces between the sub-domains extracted from the given boundary mesh.

23.1.2 Partitioning *a posteriori*

In this section, we describe some *a posteriori* partitioning methods. Specifically, there are four types of methods and some approaches based on the combination, to some degree, of two or more of these methods.

Greedy method. This approach makes use of a graph partition algorithm and is both very simple and quite intuitive. Its principle is similar to those of the node renumbering methods developed to limit the bandwidth of a finite element matrix (such as the Cuthill-McKee [Cuthill,McKee-1969] method, for instance, Chapter 17).

Given a neighborhood graph G, a starting entity $n1$, a number of expected sub-domains *ndsd* together with a targeted distribution (the number of expected entities in each sub-domain), the algorithm takes the following form :

```
set the parent entity to $n1$,
WHILE the number of sub-domain is less than $ndsd$, DO
   (A) initialize the front line to be one element, the parent entity
   FOR ALL entities in the current front line
      FOR ALL neighbors of the entity
         If the entity has not been previously marked, THEN
            put it into the current sub-domain
            add it to the list of the entities in the new front line
            under construction
            increment the number
         END IF
      END FOR
   END FOR
   If no entity is detected
      search for a new parent
      go to (A)
   IF the targeted number is that of the sub-domain
      IF the sub-domain is the last, END
      ELSE
         construct a new number 1
         return to WHILE
   END IF
   the current line is the so-constructed line
   return to FOR ALL
END WHILE
```

This algorithm is very fast but its main drawback is the generation of non-connected sub-domains. Various variations exist, for instance based on a recursive bisection, with adequate parents. This algorithm may also serve to construct a pre-partition used at a later time as the entry point of another partitioning method.

Partition by a spectral method. This graph partition method, described for example in [Simon-1991], [Gervasio *et al.* 1997] or in [Natarajan-1997] also referred to as the *Recursive Spectral Bisection* (or R.S.B.) is based on the properties of the eigenvalues and eigenvectors of a positive definite symmetric matrix.

For instance, let us consider the Laplace matrix associated with the graph dual of the mesh, then the matrix coefficients $a_{i,j}$ are defined as :

$$a_{i,j} = \begin{cases} -1 & \text{if} \quad (n_i, n_j) \quad \text{are connected} \\ deg(n_i) & \text{if} \quad i = j \\ 0 & \text{otherwise} \end{cases}$$

This matrix is useful when computing a separator between the nodes, as explained in [Donath,Hoffman-1972], [Donath-1973] or [Pothen *et al.* 1990]. One can prove that the eigenvector related to the second smaller eigenvalue of A (the so-called Fielder vector [Fielder-1975]) corresponds to minimizing a constraint about the number of connections.

If the vector components are partitioned in increasing order and if the graph is partitioned in this order, one obtains two sub-graphs with a close size in which the number of connections from one to the other is minimal. The corresponding algorithm may be written as follows :

```
$j = 1$
WHILE the number of sub-domains $j$ is less than the target
   construct the Laplace matrix of the dual graph
   compute the second smallest eigenvalue and the corresponding
   eigenvector (the Fielder vector)
   sort the graph according to the components of the Fielder vector
   distribute the entities between the two graphs
   $j = 2j$
END WHILE
```

This method is time consuming. Indeed it requires the solution of numerous eigenvalue problem for sparse matrices. Iterative methods are generally chosen as being better adapted to such problems (for instance the Lanczos method [Cullum,Willoughby-1985], [Chatelin-1993]).

Nevertheless, this algorithm has two drawbacks related to memory requirements and the necessary CPU cost. A number of variations can be developed, for instance, based on a multi-level approach that allows an improvement in costs.

Recursive partition using the inertia axis. The approach for partitioning is here related to the domain geometry. It is based on an analogy of the initial mesh with a mechanical system consisting of discrete points (*i.e.*, the vertices) with which mass are associated. This mechanical device has various inertia components including a main component that corresponds to minimizing the rotation energies.

Intuitively, the main component gives the axis along which the set is the thickest. This idea, as applied in a partitioning problem, simply leads to trying to split the set of entities into two parts along this axis. Indeed, this is the place where the interface will be minimal. This algorithm is more efficient if the mesh is homogeneous (in terms of element sizes) (and, *a contrario*, is less efficient when there is large variation in these sizes).

Computing the main inertia component involves finding the eigenvector associated with the largest eigenvalue of matrix $\mathcal{I} = {}^t\mathcal{A}\mathcal{A}$ where $\mathcal{A}$ has as its coefficients the $a_{i,j}$'s[4] by :

$$a_{i,j} = x_j(i) - g_j \,,$$

with $x_j(i)$ the coordinate of index j of the node of index i and g_j the j-coordinate of the centroid G defined by :

$$g_j = \frac{1}{n} \sum_{1=1,n} x_j(i) \,,$$

n being the number of entities under consideration (the nodes, for instance). Then $\mathcal{I}$ is a $d \times d$ matrix where d is the spatial dimension. This matrix, in three dimensions, takes the form :

$$\mathcal{I} = \begin{vmatrix} \mathcal{I}_{x_1x_1} & \mathcal{I}_{x_1x_2} & \mathcal{I}_{x_1x_3} \\ \mathcal{I}_{x_2x_1} & \mathcal{I}_{x_2x_2} & \mathcal{I}_{x_2x_3} \\ \mathcal{I}_{x_3x_1} & \mathcal{I}_{x_3x_2} & \mathcal{I}_{x_3x_3} \end{vmatrix} \qquad (23.1)$$

with :

$$\mathcal{I}_{x_j,x_k} = \sum_i (x_j(i) - g_j)\,(x_k(i) - g_k) \,,$$

[4]To establish what the coefficients are, one turns to a minimization problem. If $X(i)$ stands for vector $P(i) - G$, where G is the centroid of the $P(i)$'s and if V is an unknown unity vector, one considers the problem of minimizing the square of the distances between the $P(i)$'s and G to a line segment aligned with V passing through G. This corresponds to the following problem :

$$\max_V \sum_i \langle X(i), V \rangle^2$$

under the constraint :

$$\langle V, V \rangle - 1 = 0 \,.$$

The solution corresponds to the case where the gradients of these two expressions are aligned. Then, one looks for a coefficient λ such that :

$$\sum_i X(i)\,\langle X(i), V \rangle = \lambda\, V$$

In terms of the coordinates $x_j(i)$'s of the $X(i)$'s and v_j's of V, this leads to :

$$\left(\sum_i x_1(i)^2 - \lambda\right) v_1 + \sum_i x_1(i)x_2(i)\, v_2 + \sum_i x_1(i)x_3(i)\, v_3 = 0 \,,$$

$$\sum_i x_1(i)x_2(i)\, v_1 + \left(\sum_i x_2(i)^2 - \lambda\right) v_2 + \sum_i x_2(i)x_3(i)\, v_3 = 0 \,,$$

$$\sum_i x_1(i)x_3(i)\, v_1 + \sum_i x_2(i)x_3(i)\, v_2 + \left(\sum_i x_3(i)^2 - \lambda\right) v_3 = 0 \,,$$

which, in a matrix form, is :

$$\begin{pmatrix} \sum_i x_1(i)^2 - \lambda & \sum_i x_1(i)x_2(i) & \sum_i x_1(i)x_3(i) \\ \sum_i x_2(i)x_1(I) & \sum_i x_2(i)^2 - \lambda & \sum_i x_2(i)x_3(i) \\ \sum_i x_3(i)x_1(i) & \sum_i x_3(i)x_2(i) & \sum_i x_3(i)^2 - \lambda \end{pmatrix} \begin{pmatrix} v_x \\ v_y \\ v_z \end{pmatrix} = 0 \,.$$

This system has a non null solution if its determinant is null. Thus, λ is an eigenvalue of the previous matrix $\mathcal{I}$.

thus, in particular, we have as diagonal coefficients :

$$\mathcal{I}_{x_j,x_j} = \sum_i (x_j(i) - g_j)^2 \,.$$

In three dimensions, the problem reduces to computing the largest eigenvalue of a 3×3 matrix and then computing the corresponding eigenvector. This task is easy for instance by means of the power method. The partitioning algorithm then takes the form :

```
set $j = 1$ (initialization of the number of members in the partition)
WHILE $j$ is less than the target
   FOR $i = 1, j$ where $i$ denotes a sub-domain
      compute the centroid $G$ of sub-domain $i$
      compute the main inertia component of sub-domain $i$.
      (This involves computing $\mathcal{I}$ for the entities in $i$ and then
      picking the largest eigenvalue and eigenvector)
      project the entities in $i$ following this eigenvector
      sort the projected entities
      partition into two the sub-domain $i$ according to the
      sorted projected entities
   END FOR $i$
   $j = 2j$
END WHILE
```

This algorithm can be easily adapted to obtain an arbitrary number of sub-domains. To do so, it is sufficient to modify the way in which the partition into two sub-domains is made by partitioning not in a homogeneous way but following a distribution which is proportional to the number of sub-domains expected when dealing with the current sub-domain, in one side or the other of the partition.

Remark about the inertia matrix. The above method makes use of the matrix defined in Relationship (23.1) which is not the inertia matrix widely used in mechanical engineering. The latter is written as follows :

$$\mathcal{I} = \begin{vmatrix} \mathcal{I}_{x_1x_1} & \mathcal{I}_{x_1x_2} & \mathcal{I}_{x_1x_3} \\ \mathcal{I}_{x_2x_1} & \mathcal{I}_{x_2x_2} & \mathcal{I}_{x_2x_3} \\ \mathcal{I}_{x_3x_1} & \mathcal{I}_{x_3x_2} & \mathcal{I}_{x_3x_3} \end{vmatrix} \qquad (23.2)$$

but, now, with :

$$\mathcal{I}_{x_j,x_k} = -\sum_i (x_j(i) - g_j)\,(x_k(i) - g_k) \,,$$

and, for the diagonal coefficients, expressions of the form :

$$\mathcal{I}_{x_j,x_j} = \sum_i \left((x_{j+1}(i) - g_{j+1})^2 + (x_{j+2}(i) - g_{j+2})^2\right) \,,$$

with obvious notations about the indices $j+1$ and $j+2$, for a given j.

Clearly, in two dimensions, the eigen elements of matrices (23.1) and (23.2) are the same. The same is true in three dimensions (or for higher dimensions). To be certain of this, just remark that the sum of these two matrices (in three dimensions) is :

$$\sum_i \left(\sum_j (x_j(i) - g_j)^2 \right) \mathcal{I}_3 ,$$

in other words a diagonal matrix. A simple examination of the eigen elements of the two matrices (while using this relationship) allows for the proof.

K-means method. Initially developed in a different context (classification problems, [Celeux *et al.* 1989]), the aim is still the same, we try to balance in a certain number of classes the entities in a given set, here a mesh.

Let us consider $\mathcal{S}$ a cloud of points (the mesh points, in a vertex based algorithm or the element centroids, in an element based algorithm). An outline of the K-means method is as follows :

- choosing, using one method or another (random choice, inertia axis, octree type structure, etc.) k seeds (or kernels) where k is the number of classes to be constructed,
- (A) - associating the points in $\mathcal{S}$ with a seed based on a given criterion (for instance, a proximity criterion). This points-seeds relationship then defines k classes,
- computing, for every class, a new seed (for example, its centroid), then return to (A).

The interesting feature in this method is, after some adequate assumptions, its convergence. Indeed, it builds a set of k stable classes. In addition, these classes satisfy (maximize) a criterion meaning that the classes and their seeds are in adequation and that the classes are well aggregated and well separated from one another.

Convergence results from observing that a series associated with the process is a decreasing series and thus the algorithm implies that a criterion decreases at each step until stability which, in turn, ensures that the solution is complete.

A few examples of partitioning output. We now give (Figures 23.2 to 23.5) some application examples resulting from the above methods. More precisely, Figure 23.2 (i-iv) shows the partitioning of two dimensional domains as obtained by a greedy method with VOD or EOD choices.

Figure 23.3 depicts two partitions of a domain, in two dimensions, using *Recursive Spectral Bisection* methods, one vertex based (VOD), the other element based (EOD). We can see in this example that the (greedy) bisection method produces a rather similar result.

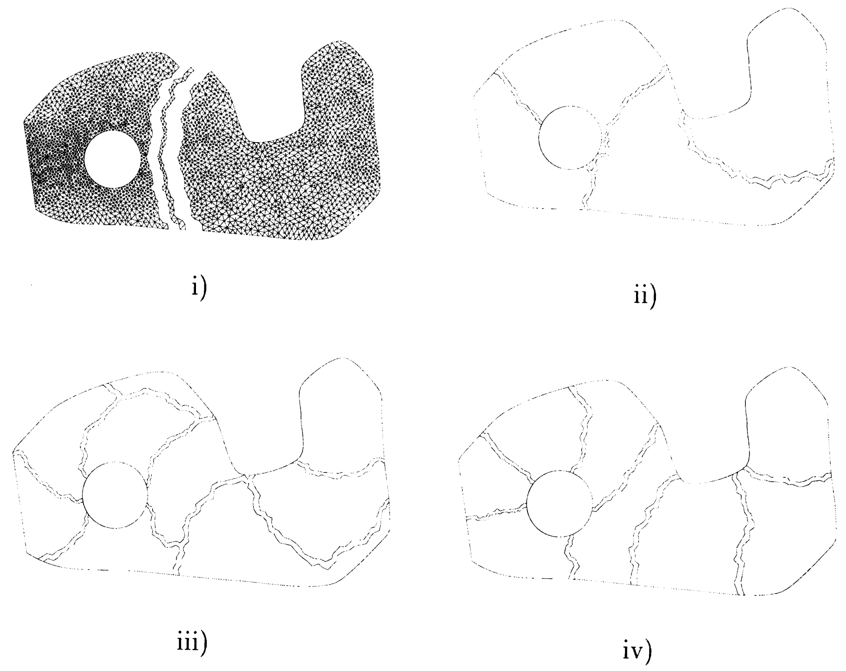

Figure 23.2: *Partition of a sub-domain in two dimensions using a VOD greedy method into, respectively, two i), four ii) and eight iii) sub-domains. Partition by a multigrid-greedy EOD method iv).*

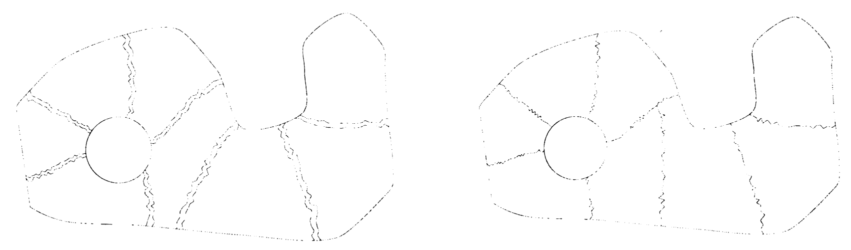

Figure 23.3: *Partition of a sub-domain into eight sub-domains, using a VOD RSB type method, left-hand side and a EOD RSB method, right-hand side.*

Figure 23.4 (i-iv) shows, for the same case, some partitions into two, four and eight sub-domains as completed by an inertia axis method.

Then, the case of a three dimensional domain is depicted in Figure 23.5. The domain has been split into eight sub-domains using a VOD spectral method and an inertia axis EOD method.

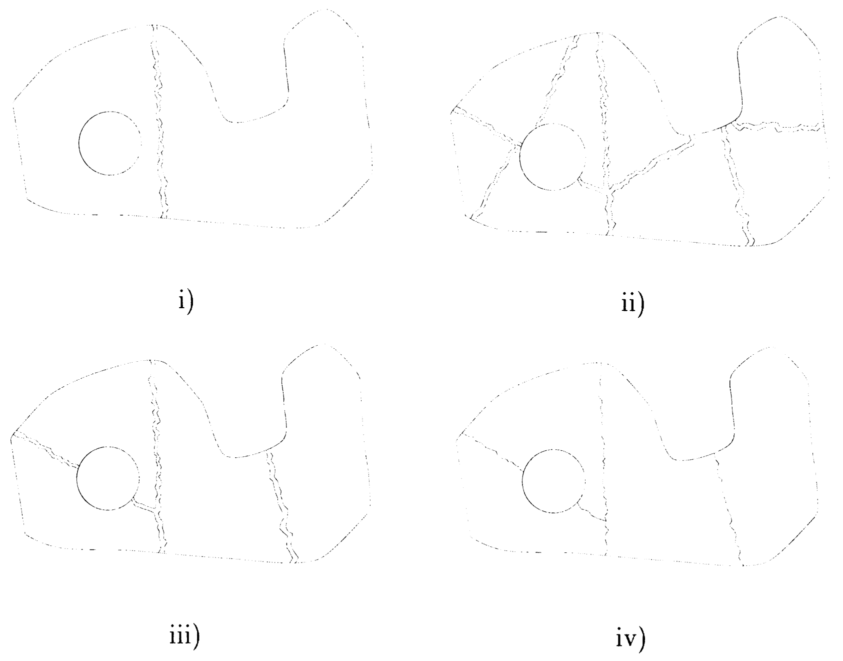

Figure 23.4: *Partition of a sub-domain by an* inertia axis *method (in the VOD case) into two i), four iii) and eight ii) sub-domains and in the EOD case in the case where four sub-domains are desired, iv).*

23.1.3 Partitioning *a priori*

In what follows we discuss two particular *a priori* partitioning methods. Afterwards, some remarks about other methods will be given.

Method 1. In this approach the entry point is a relatively (or a really) coarse mesh of the entire domain. The technique may be classified as an *a priori* method in the sense that it is not necessary to mesh the domain with the fine mesh that will be the support of the envisaged computation in order to define the partition.

The coarse mesh, the basis of the partition, may be easily obtained using a Delaunay type method (Chapter 7), based on the discretization of the boundary of the entire domain. The meshing process is then stopped when the domain is covered, *i.e.,* once the boundary points have been inserted and the boundary integrity step has been completed. In other words, no (or a few) internal vertices exist inside the domain.

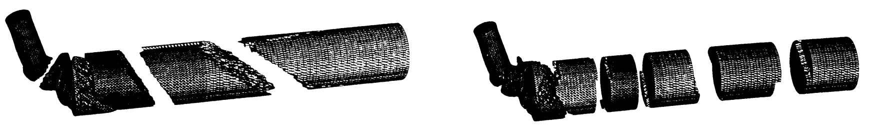

Figure 23.5: *Partition into eight sub-domains by a* Recursive Spectral Bisection *VOD type method, left-hand side and by an* inertia axis *EOD method, right-hand side (data courtesy of Renault).*

One uses a splitting algorithm based on this mesh, for instance, by a *greedy* type method (see above). This results in a partition of the domain into several sub-domains. However, it is tedious to balance this partition. In particular, two questions must be carefully addressed that concern the well-balancing aspect and, on the other hand, the shape and the smoothness of the interfaces.

A simple idea to obtain a nice balancing (say an equidistribution of the elements) is to introduce a weight to the vertices such as the average of the volumes (surfaces in two dimensions) of the elements sharing these vertices. In this way, we implicitly assume that the number of elements in the final mesh is proportional to the size of the initial elements. This choice is a reasonable idea when a uniform density is expected, but less reasonable in other situations. Thus, in such cases, it is necessary to adjust the weights with information of a metric nature. In other words, two initial elements of identical size do not necessarily lead to the same number of elements after meshing.

For a domain in two dimensions, the interface between sub-domains is basically formed by line segments which separate the domain from one side to the other. In three dimensions, the interfaces are composed of a series of triangular facets and the angles between two such facets may be arbitrary : thus the geometry is likely to be badly-shaped (disturbed) resulting in a rather peculiar aspect at the interface levels.

Then, the method includes the following :

- interface meshing,
- sub-domain definitions and balancing these over the different processors,
- serial meshing for each sub-domain (one sub-domain in one processor).

Relationships between the different meshes are established at the end of the process in order to allow the transfer of data at a later stage.

Method 2. Following this approach, a mesh of the domain is not required and only a mesh of the domain boundary is used. The idea is then to directly build one or several separators (lines or curves, planes or surfaces) so as to define several sub-domains based on these boundaries.

The method primarily proposed by [Galtier,George-1996], [Galtier-1997] is based on the inductive Delaunay triangulation notion (termed projective Delaunay triangulation in the above references). To introduce this notion, let us recall that :

$$V_i = \{P \quad \text{such that} \quad d(P,P_i) \leq d(P,P_j), \quad \forall j \neq i\}$$

allows us to define the Voronoï cell associated with point P_i, a member of a given cloud of points (Chapter 7). The dual of these cells (that constitute the so-called Voronoï diagram) is the Delaunay triangulation of the convex hull of the points in the cloud. This triangulation satisfies the empty ball criterion.

By analogy, we now consider a plane Π and we define the set :

$$V_i^{(\Pi)} = \{P \in \Pi \ \text{such that} \quad d(P,P_i) \leq d(P,P_j), \quad \forall j \neq i\}, \qquad (23.3)$$

which is a polygon in $\mathbb{R}^3$. The dual of the different cells defined in this way is a set of triangular facets in $\mathbb{R}^3$ obtained by joining the points whose cells share a common edge. This set of facets (triangles) satisfies a criterion similar to that of the empty ball, referred to as the *inductive* criterion hereafter. This set of facets is denoted by $\mathcal{F}^\Pi$.

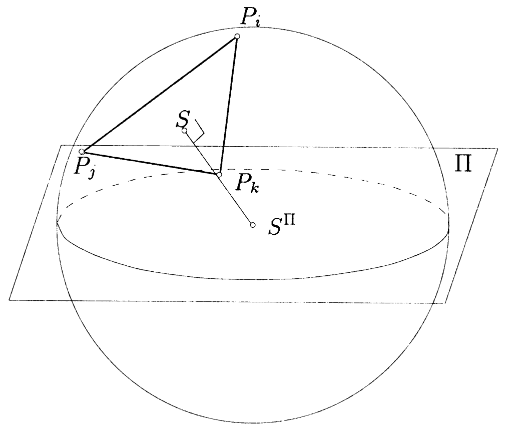

Figure 23.6: *The inductive empty ball criterion. The ball centered in the plane* Π *passing through the three vertices of facet* $P_iP_jP_k$ *is empty.*

In [Galtier-1997] it is proved that the facets constructed in this way form a separator in the domain (here, the convex hull of the cloud). Thanks to this result, it is possible to split this particular domain into two parts. The principle of this partitioning is to use this idea and to apply it to a domain in $\mathbb{R}^3$ (and not only to the convex hull of a set of points).

The proof is based on some assumptions that simplify the problem (for instance about the point positions). First, one shows that the set of facets $\mathcal{F}^\Pi$ is a planar graph. Then one notices that a triangle $P_iP_jP_k$ is a facet in $\mathcal{F}^\Pi$ if and only if the

three Voronoï cells, after Definition (23.3), are not empty and share a vertex. Then, one establishes that these faces are Delaunay-admissible (following the definition in Chapter 9) and therefore will be formed in every Delaunay triangulation. One deduces in addition that the facets in $\mathcal{F}^\Pi$ define a separator in the convex hull of the domain.

The principle of the partitioning method is then to apply these issues to an arbitrary domain in $\mathbb{R}^3$ (and not only in a convex hull problem) and, in particular, to remove the above assumptions, which were made for the sake of simplification (the points are now assumed to be in an arbitrary position and not in a general position). The main steps of the splitting method are as follows (in the case where the separator entity is constructed from a plane) :

- a polygonal line (a set of edges among the triangle edges in the mesh of the domain surface) is found such that this set of segments is Delaunay admissible for the inductive empty ball criterion,
- the points on this line are inserted using a Delaunay type algorithm. Thus providing a mesh of a surface whose boundary is this line.

In this way, an initial mesh of a surface that separates the domain into two parts is obtained. Then :

- some points are created on this surface (for instance, as in Chapter 7, by using the edges as a spatial support),
- these points are inserted.

On completion, we have a mesh of the separator surface. Thus, it is possible to define two sub-domains (through their boundary meshes) whose boundaries are composed of the union of the separator mesh with the adequate parts of the surface mesh of the entire domain.

By repeating this process, it is possible, step by step, to cut the initial domain into several sub-domains. As the interface between two sub-domains is defined in a unique way, a classical mesh construction algorithm (*i.e.*, such that boundary integrity is maintained) can be suitably applied in each sub-domain[5].

Notice, and here is our concern, that the serial mesher is used in parallel meaning that the global meshing effort has been distributed over several processors (one per sub-domain).

A few remarks may be given. In specific, questions arise about the way in which plane Π mentioned above must be chosen and, this being done (a separator surface being available) about how to discretize this surface so as to obtain a mesh whose density reflects the desired element sizes. Also the recurrent problem of well-balancing the task must be addressed.

As concerns this last question, notice that if it is possible to evaluate, even in a rough way, the volume of the sub-domains (with regard to the density of the points in their boundaries, which is the only data available at this time), then it

[5]Thus, due to the required properties, it seems natural to use a Delaunay type meshing method.

is possible to predict the number of elements (its order of magnitude) that will be constructed within each sub-domain. For more details about this method, we refer the reader to the already mentioned references.

Other methods. Multi-block type methods (Chapter 4) as well as methods based on a space decomposition (Chapter 5) are also essentially decomposition methods. Other methods can also be considered, for example, at the level of the CAD system (thus, before any meshing concerns).

The most favorable and automated method is most probably the *quadtree-octree* type method which is based on a recursive decomposition of a box enclosing the domain of interest. The decomposition here is directly connected with the underlying tree structure.

This approach will be discussed below, seen as a parallel meshing method, in the sense where the same principle applies even if some modifications are made.

23.2 Parallel meshing process

Throughout this section, we will consider two cases. First, we are concerned with the design of a static process of parallel meshing, *i.e.*, used once. Then, we will see the case of a computation loop. The computational process is then a dynamic task where at each iteration step of the loop, the issue of parallelism must be examined.

23.2.1 One-step process

The load balancing between the members in the partition must be ensured at the beginning (*i.e.*, when constructing the partition) or, this having been constructed as well as possible, before using the resulting meshes (*i.e.*, before any actual computation). Thus, we can distinguish between two types of load balancing approaches, *a priori* load balancing and *a posteriori* load balancing.

***A priori* load balancing.** When the processors used at a later stage for the computation have comparable speed-up, the effort balancing can be done *a priori* by distributing identical numbers of vertices (elements) to each processor. In contrast, when the speed-up varies (from processor to processor) or when some processors are busy with other tasks, an equidistribution is not the optimal solution. It is then necessary to re-balance the loads during the computation.

As a general rule, such a load balancing is tedious to obtain. To realize this, one only has to keep in mind the different partitioning approaches previously discussed. Hence, for example, the key-idea to ensure a good load balancing is to introduce some weights during the partitioning stage. This idea implicitly assumes that the number of elements in the final mesh is proportional to the size of the initial mesh. When the element density is not uniform (which, in practice, is frequent), such an assumption is obviously wrong. Then, the weights have to be adjusted to take into account metric specifications, which makes the load balancing stage more delicate.

***A posteriori* load balancing.** The load balancing per processor is basically obtained by *migration*. The migration step consists in moving one element or a set of elements from one processor to another (or to several processors), if the load from one processor to the others is rather different [Coupez-1996]. Moreover, the migration step may serve some other purposes such as obtaining a nice smoothness at the sub-domain interfaces and can also be used as a way to deal with non connected sub-domains.

The goal of the partitioning algorithms is to minimize data exchange between the processors and to balance the computational effort in each processor. The data migration actually consists in collecting and updating the links between the mesh entities and the processors when some entities are assigned to one (or several) different processors. This is generally done in three steps :

- a *source* processor sends some data items to a *target* processor,
- the source and target processors update the information related to the transferred data items,
- the source and target processors indicate this change to all the processors with access to the data items that are exchanged (see below).

The efficiency of the migration procedure then greatly depends on the volume of the data items exchanged as well as on the data structure used in this task.

Data structures. (Adaptive) mesh generation methods for unstructured mesh construction when used in distributed memory architecture require specific data structures suitable for efficiency queries and data item exchanges from processor to processor. In particular, information about adjacency relationships must be considered.

At a given mesh entity level, it is necessary to know all the relationships (links) between this entity and the different processors. In principle, only a single processor really *possesses* this entity but several other processors may have some links to it (thus, this information is not duplicated in each processor). The edges and faces of the sub-domain boundaries are then shared by several processors.

It is clear that in the first step of the migration process the amount of data item exchange is proportional to the number of entities exchanged (thus, to np the number of vertices). On the other hand, the two other steps depend on the number of entities in the sub-domain boundaries (thus typically in $\mathcal{O}(np^{\frac{1}{2}})$).

23.2.2 Parallel adaptation loop

Here, load balancing is a *dynamic* task that must be adapted at each new iteration step (or, at least, when the sizes of the members in the partition become rather different). This load balancing is a fundamental issue since a physical problem with a solution that has a large variation may, during the iteration in an adaptive computational process, lead to a large variation in the number of elements in a given region (thus in a given processor). However, in contrast to the previous case, it is possible to collect some information from the current partition and the behavior of the solution related to it in order to deduce the next partition in the iterative process in progress.

Without mentioning here the probabilist style methods, some algorithms may use a *defect* function that reflects the difference in the load balancing between neighboring elements [Löhner,Ramamurti-1993], in order to govern data exchanges between processors. One may also use some variations of the partitioning algorithms previously described that operate on distributed data. Typically, the partitioning algorithms are either of a geometrical nature (inertia axis based method for instance) or of a topological nature (using the adjacency relationships of a tree structure, see below).

23.3 Parallel meshing techniques

First, it could be observed that the parallel aspect can be present at the input data level, the data being distributed between the processors, or at the level of the tasks (which are then distributed) or, again, in a combination of these two levels.

Distributing the data (without task exchange) involves making the task in each processor independent of the others. Distributing the tasks leads to sending some requests from processor to processor when dealing with an entity (located in one processor), and implies some processing from the proprietary processor but also from some neighboring processors.

Various classical techniques for mesh generation may be performed in parallel. Among these, the Delaunay method is a potentially interesting candidate. Indeed, this method widely uses a *proximity criterion*, namely the empty ball criterion. Given a set of points, one may consider separating these points into several disjoint sub-sets and thus make the point insertion procedure uncoupled. In other words, the mesh can be completed in parallel.

In practice, this point separation could either be decided *a priori*, which is in general a tedious issue, or prescribed by means of a constraint about edges (resp. faces) in a coarse triangulation used as a separator. Once this separator has been defined, the problem turns to meshing with consistency the sub-domains identified in this way, while their interface (*i.e.*, the separator) could be meshed :

- before constructing the mesh of the sub-domains,
- at the same time as the sub-domains,
- after constructing the mesh of the sub-domains,

which, in fact, leads to three classes of parallel meshing methods.

Another class of methods for parallel use is based on the spatial decomposition of a box enclosing the computational domain. Methods like *quadtree-octree* (Chapter 5) are then natural candidates since they make use of a hierarchical tree structure and thus could be used for domain partitioning purposes.

Are any other meshing methods candidates for parallelism ? The answer is not so trivial. In particular, it is of interest to examine whether an advancing front type method (Chapter 6) may include some degree of parallelism.

Within this section, we give some indications about these various approaches and briefly examine their ability to include some parallel aspect.

Remark 23.1 *Note that the concern here is to construct a mesh in parallel and not to define a parallel mesh construction or computational process. In the present context, the wish is to have a complete mesh available (after merging together the sub-meshes) while in the other case, it is not strictly necessary (see the previous sections) or we do not want to have, at a given time, the complete mesh of the domain.*

23.3.1 Delaunay-type method

Let us recall that the key-point of a Delaunay style method is the point insertion process, the *Delaunay kernel* as introduced in Chapter 7. This process is a local one in the sense that it involves the cavity of the point to be inserted. Depending on whether the elements in this cavity all belong to the same processor or are common to two or more processors, it is easy to imagine that the degree of parallelism could be different. Simply remark, in addition, that finding this locality could be a non trivial problem (or, at least, a time consuming task).

Various ways to implement the Delaunay kernel have been proposed based on the various possible configurations (in this respect, [Okusanya,Peraire-1996], [Chrisochoides,Sukupp-1996] or [Chew *et al.* 1997] can be consulted, for example).

General principle. For the sake of simplicity, let us consider a domain Ω in two dimensions, provided through a discretization $\Gamma(\Omega)$ of its boundary. Using this sole data, an initial mesh is constructed by inserting, one at a time, the points in $\Gamma(\Omega)$, and then by enforcing the boundary entities that are missing, if any[6].

In this way, an *empty* mesh is obtained (*i.e.*, without any internal points) whose internal edges join one domain side to another. We pick one of these edges, say AB, which separates Ω into two sub-domains, Ω_1 and Ω_2 of approximately the same size. Each of these two domains is then assigned to one processor. Now, let us look at a condition to ensure the independence of a point with respect to the sub-domains :

- every point P inside Ω_1 is independent of Ω_2 if the cavity $\mathcal{C}_P$ is entirely included in Ω_1 : $\mathcal{C}_P \subset \Omega_1$ and $\mathcal{C}_P \not\subset \Omega_2$,
- on the other hand, if $\mathcal{C}_P \cap \Omega_2 \neq \emptyset$, point P depends on Ω_1 and Ω_2 and its insertion affects the mesh both in Ω_1 and in Ω_2.

Note that a (sufficient) condition for independence is that the circle of diameter AB is empty (Chapter 9). After this assumption, the mesh is carried out in parallel without any exchange of tasks.

[6]Thus, the case of a *constrained* Delaunay mesh (Chapter 7).

Meshing the separator. It is easy to see that, in general, the output completed using the previous algorithm will be rather poor because the constraint implies that there is no point in some vicinity of AB (*i.e.*, its open ball) and only some post-processing may be able to fill up this region (and thus to split AB).

Using a constrained Delaunay algorithm by prescribing AB as a constraint leads to the same issue and the global process remains "processor-independent". The presence of points in some vicinity of AB necessarily gives a bad quality mesh and some post-processing is again required in order to enhance this result (and, in particular, to split AB).

At last, a "classical" meshing is possible, while using data and task exchanges. This can be done in two ways. On the one hand, we maintain AB as a constraint but we may even split this segment (by means of point insertion) in such a way as to preserve the global quality of the mesh. In such a situation, the processor responsible for this point creation indicates this request to the other processor in order to make the mesh construction synchronous at the interface level (then a delay must be expected for the simultaneous updating of the two meshes because the interface must remain coherent).

The other way to proceed is more tedious. Edge AB (or any interface edge) is no longer a constraint and may disappear. The interface is then modified during the processing which impedes the global process in terms of data and task organization.

After this discussion about the principles, one may observe that there are only a few examples of implementations of these ideas in two dimensions, for architectures with a limited number of processors (actually, from 2 to 16). The extension of these approaches to three dimensions remains a relatively open problem. In particular, constraining a mesh (Chapter 7) is a tedious problem which, in practice, is widely based on heuristics. Moreover, identifying a separator is not obvious because even a face joining two "opposite" sides of the domain does not in general separate this domain into two parts.

23.3.2 *Quadtree-octree*-type method

Here two approaches will be discussed. The first, mainly of academic interest, assumes the tree structure to be known and, using this information, simply seeks to balance the terminal cells (the leafs) in the available processors. The second one, which is more interesting from a practical point of view, considers the tree structure construction in parallel.

Distribution of the nodes in a given tree. For the sake of simplicity, we assume the tree to be balanced by means of the [2:1] rule (Chapter 5), thus every cell edge is shared by at most three cells, in two dimensions, while a cell facet is shared by five cells at most in three dimensions. Following on from this, one could observe the natural link between the number of terminal cells and the number of elements in the resulting mesh[7].

[7]In fact, we saw in Chapter 5 that the cell sizes are locally compatible with the element size distribution function.

Therefore obtaining a nice load balancing in the processors involves partitioning the tree in such a way that the number of leaves assigned to each processor leads to the same number of elements. Meshing each sub-domain is then performed in parallel, with no communication between the processors. Since a quadrant boundary edge (resp. face) may belong to several processors, special care is necessary regarding the meshing algorithm used to mesh these interfaces (as the resulting meshes must be conformal). On the other hand, if the leaves in the interface are inside the domain, predefined patterns (or *templates*) can be used. The adjacency relationships between the tree cells must be enriched with information about the processor to which the cell belongs [Saxena,Perucchio-1992].

Following these remarks, the general scheme of a tree partitioning method simply consists in the following :

```
ne ← 0
FOR each leaf c in the tree
   compute the number of elements that must be created $ne(c)$
   $ne \leftarrow ne + ne(c)$, (increment the total number of elements)
END FOR
compute the load per processor :  $cpp = \frac{ne}{nbproc}$
traverse the tree
$nep \leftarrow 0$
$p \leftarrow 1$
FOR each leaf c
   IF $nep < cpp$
      $nep \leftarrow nep + ne(c)$
      assign the $ne(c)$ elements to processor p
   OTHERWISE
      $p \leftarrow p + 1$
   END IF
END FOR
```

To take advantage of the adjacency relationships in the tree (and thus obtain sub-domains that are as connected as possible), one may traverse the tree at first (using a pre-order as seen in Chapter 2).

Distributed construction of the tree. A more interesting problem is the design in parallel of the whole meshing process, *i.e.*, the tree construction and its use to form the mesh elements.

The method proposed in [deCougny *et al.* 1996] constructs an *octree* which is balanced in parallel. The entry data in this algorithm is a discretization of the domain surface. With each vertex in this triangulation is associated a size (that corresponds to the average of the lengths of the incident edges), thus defining a discrete size map. This scalar value h is then modified in an integer value related to a level l in the tree (*i.e.*, the level of the cell within which the vertex falls), by means of the formula (Chapter 5) :

$$l = log_2\left(\frac{b}{h}\right) ,$$

where b is the length of a side of the enclosing box.

Conceptually, the tree construction includes two stages, the construction of local sub-trees and the refinement of these sub-trees. The four (resp. eight) initial cells are assigned to four (eight) different processors. Then, all the terminal cells are iteratively subdivided once or more (if necessary) and the cells resulting from the decomposition are assigned to the different processors according to the load they imply. This effort approximatively corresponds to the number of vertices to be inserted (*i.e.*, this is the stopping criterion of the process, Chapter 5) inside the volume related to one cell. Notice that the non-terminal cells are then replaced by the leaves. In this way, on completion of the construction (once all the processors have been assigned an equivalent load), each terminal cell corresponds to the root of a sub-tree and exists in only one processor. The sub-tree construction is in this way balanced in a natural way , each processor having approximately the same number of vertices to be inserted.

Then, the sub-trees are constructed independently in each processor by subdividing their terminal cells until a level related to the desired local size is achieved.

The meshes of the sub-trees are then completed in parallel (with no communication between the processors) by using in each sub-tree a technique similar to that described in Chapter 5.

Remark 23.2 *In three dimensions, a tree re-balancing step may be made after the mesh of the terminal cells inside the sub-domains has been completed. This is due to the fact that an advancing front type algorithm is often used to mesh the boundary leaves (the number of elements in these cells being tedious to evaluate* a priori*).*

Complexity. The complexity in time of the construction process is of the order of $\mathcal{O}(np/nproc \log(nproc))$, where $np/nproc$ is the load per processor and where the term in $\log(np)$ corresponds to the number of iteration steps in the algorithm (thus to the number of refinement levels in the tree). While the sub-tree refinement is of a complexity $\mathcal{O}\left(\frac{np}{nproc}\log\left(\frac{np}{nproc}\right)\right)$.

23.3.3 Other methods ?

The third category of automated mesh generation methods not yet discussed is that of the advancing front type methods. At present, we have no knowledge of developments or specific issues for this type of approach. We may just mention that the advancing front strategy may be used for a parallel mesh construction of sub-domains while the underlying decomposition is completed using a tree structure or a Delaunay type empty mesh (with no internal point), see, for instance [Shostko,Löhner-1992] or [Löhner-1998]).

A short conclusion

Parallelism appears to be a practical solution to handle systems with several million (or dozens of millions of) unknowns[8].

The impact on meshing technologies seems to be, as we have seen, more about the parallelization of the meshing processes (partitioning, load balancing and communication) than the parallelization of the meshing methods themselves. It is likely that such parallel methods will be subjected to important developments in the coming future and, as a consequence, significant advances are expected. Thus this subject appears to be both open and very promising.

[8]Such systems are nowadays used in computational fluid dynamics where the order of magnitude of the physics varies greatly.

Bibliography

[Aftomis *et al.* 1995] M. Aftomis, J. Melton and M. Berger (1995), Adaptation and surface modeling for Cartesian mesh methods, *AIAA paper 95-1725*, 12th AIAA CFD Conf., San Diego, CA.

[Aggarwal *et al.* 1989] A. Aggarwal, L.J. Guibas, J. Saxe and P.W. Shor (1989), A linear-time algorithm for computing the Voronoï diagram of a convex polygon, *Discrete Comput. Geom.*, **4**(6), 591-604.

[Aho *et al.* 1983] A. Aho, J. Hopcroft and J. Ullman (1983), *Data structures and algorithms*, Addison-Wesley, Reading, Mass.

[Allgower,Schmidt-1985] E.L. Allgower and P.H. Schmidt (1985), An algorithm for piecewise-linear approximation of an implicitly defined manifold, *SIAM J. Numer. Anal.*, **22**, 322-346.

[Allgower,Gnutzmann-1987] E.L. Allgower and S. Gnutzmann (1987), An algorithm for piecewise linear approximation of implicitly defined two-dimensional surfaces, *SIAM J. Numer. Anal.*, **24**(2), 452-469.

[Allgower,Georg-1990] E.L. Allgower and K. Georg (1990), Numerical continuation methods : an introduction, *Springer-Verlag.*

[Allgower,Gnutzmann-1991] E.L. Allgower and S. Gnutzmann (1991), Simplicial pivoting for mesh generation of implicitly defined surfaces, *CAGD*, **8**, 305-325.

[Amenta *et al.* 1997] N. Amenta, M. Bern, and D. Eppstein (1997), Optimal point placement for mesh smoothing, *8th ACM-SIAM Symp. Discrete Algorithms*, New Orleans, 528-537.

[Amezua *et al.* 1995] E. Amezua, M.V. Hormaza, A. Hernandez and M.B.G. Ajuria (1995), A method for the improvement of 3d solid finite-element meshes, *Advances in Engineering Software*, **22**, 45-53.

[Anglada *et al.* 1999] M.V.Anglada, N.P. Garcia and P.B. Crosa (1999), Directional adaptive surface triangulation, *Computer Aided Geometric Design*, **16**, 107-126.

[Armstrong *et al.* 1993] C.G. Armstrong, D.J. Robinson, R.M. McKeag, D. Sheehy, C. Toh and T.S. Li (1993), Abstraction and meshing of 3d stress analysis models, *Proc. ACME Research Conf.*, Sheffield Univ., England.

[Armstrong *et al.* 1995] C.G. ARMSTRONG, D.J. ROBINSON, R.M. MCKEAG, T.S. LI, S.J. BRIDGETT AND R.J. DONAGHY (1995), Applications of the medial axis transform in analysis modelling, *Proc. Fifth International Conference on Reliability of Finite Element Methods for Engineering Applications*, Nafems, 415-426.

[Armstrong *et al.* 1995] C.G. ARMSTRONG, D.J. ROBINSON, R.M. MC KEAG, T.S. LI, S.J. BRIDGETT, R.J. DONAGHY AND C.A. MC GLEENAN, Medials for meshing and more, *proc. 4th Int. Meshing Roundtable*, Albuquerque, NM, 277-288.

[Attili-1997] B.S. ATTILI (1997), Tracing implicitly defined curves and the use of singular value decomposition, *Applied Num. Math.*, **25**, 1-11.

[Aurenhammer-1987] F. AURENHAMMER (1987), Power diagrams properties, algorithms and applications, *SIAM J. Comput.*, **16**, 78-96.

[Aurenhammer-1990] F. AURENHAMMER (1990), A new duality result concerning Voronoï diagrams, *Discrete Comput. Geom.*, **5**, 243-254.

[Aurenhammer-1991] F. AURENHAMMER (1991), Voronoïdiagrams: a survey of a fundamental geometric data structure, *ACM Comput. Surv.*, **23**, 345-405.

[Avis,Bhattacharya-1983] D. AVIS AND B.K. BHATTACHARYA (1983), Algorithms for computing *d*-dimensional Voronoï diagrams and their duals, in F.P. Preparata eds. *Advances in Computing Research*, **1**, 159-188.

[Avis,ElGindy-1987] D. AVIS AND H. ELGINDY (1987), Triangulating point sets in space, *Discrete Comput. Geom.*, **2**, 99-111.

[Azevedo,Simpson-1989] E.F. D'AZEVEDO AND B. SIMPSON (1989), On optimal interpolation triangle incidences, *Siam J. Sci. Stat. Comput.*, **10**, 1063-1075.

[Azevedo-1991] E.F. D'AZEVEDO (1991), Optimal triangular mesh generation by coordinate transformation, *Siam J. Sci. Stat. Comput.*, **12**, 755-786.

[Azevedo,Simpson-1991] E.F. D'AZEVEDO AND B. SIMPSON (1991), On optimal triangular meshes for minimizing the gradient error, *Numerische Mathematik*, **59**(4), 321-348.

[Avnaim *et al.* 1994] F. AVNAIM, J.D. BOISSONNAT, O. DEVILLERS, F.P. PREPARATA AND M. YVINEC (1994), Evaluating signs of determinants using single-precision arithmetic, *RR INRIA* **2306**.

[Babuska,Aziz-1976] I. BABUŠKA AND A. AZIZ (1976), On the angle condition in the finite element method, *SIAM J. Numer. Analysis*, **13**, 214-227.

[Babuska,Rheinboldt-1978] I. BABUŠKA AND W.C. RHEINBOLDT (1978), A posteriori error estimates for the finite element method, *Int. j. numer. methods eng.*, **12**, 1597-1615.

[Babuska,Guo-1988] I. BABUŠKA AND B.Q. GUO (1988), The hp version of the finite element method for domains with curved boudaries, *SIAM j. numer. anal.*, **25**(4), 837-861.

[Babuska *et al.* 1989] I. BABUŠKA, M. GRIEBEL AND J. PITKARANTA (1989), The problem of selecting the shape functions for p-type finite elements, *Int. j. numer. methods eng.*, **28**, 1891-1908.

[Babuska,Suri-1990] I. BABUŠKA AND M. SURI (1990), The p end the hp versions of the finite element method : An overview, *Comp. Meth. Appl. Mech. Engng.*, **x**, 5-26.

[Babuska *et al.* 1994] I. BABUŠKA, T. STROUBOULIS, C.S. UPADHYAY, S.K. GANGARAJ AND K. COPPS (1994), Validation of a posteriori error estimation by numerical approach, *Int. j. numer. methods eng.*, **37**, 1073-1123.

[Babuska,Guo-1996] I. BABUŠKA AND B.Q. GUO (1996), Approximation properties of the hp version of the finite element method, *Comp. Meth. Appl. Mech. Engng.*, **133**, 319-346.

[Baehmann *et al.* 1987] P.L. BAEHMANN, S.L. WITTCHEN, M.S. SHEPHARD, K.R. GRICE AND M.A. YERRY (1987), Robust geometrically-based automatic two-dimensional mesh generation, *Int. j. numer. methods eng.*, **24**, 1043-1078.

[Bai,Brandt-1987] D. BAI AND A. BRANDT (1987), Local mesh refinement multilevel techniques, *Siam J. Sci. Stat. Comput.*, **8**(2), 109-134.

[Bajaj *et al.* 1993] C. BAJAJ, I. IHM AND J. WARREN, Higher-order interpolation and least-squares approximation using implicit algebraic surfaces, *ACM Trans. Graph.*, **12**, 327-347.

[Bajaj *et al.* 1997] C. BAJAJ, J. BLINN, M.P. GASCUEL, A. ROCKWOOD, B. WYVILL AND G. WYVILL, Introduction to Implicit Surfaces, J. Bloomenthal ed., The Morgan Kaufmann Series in Computer Graphics and Geometric Modeling, Brian A. Barsky, Series Ed.

[Baker-1986] T.J. BAKER (1986), Three dimensional mesh generation by triangulation of arbitrary point sets, *Proc. AIAA 8th Comp. Fluid Dynamics Conf.* Honolulu, HI, **87**-1124.

[Baker *et al.* 1988] B.S. BAKER, E. GROSS AND C. RAFFERTY (1988), Nonobtuse triangulation of polygons, *Discrete and Comp. Geometry*, **3**, 147-168.

[Baker-1988] T.J. BAKER (1988), Generation of tetrahedral meshes around complete aircraft, *Numerical grid generation in computational fluid mechanics '88*, Miami, FL.

[Baker-1989a] T.J. BAKER (1989), Developments and trends in three-dimensional mesh generation, *Appl. Num. Math.*, **5**, 275-304.

[Baker-1989b] T.J. BAKER (1989), Element quality in tetrahedral meshes, *Proc. 7th Int. Conf. on Finite Element Methods in Flow Problems*, Huntsville, AL.

[Baker-1989c] T.J. BAKER (1989), Automatic Mesh Generation for Complex Three-Dimensional Regions Using a Constrained Delaunay Triangulation, *Eng. Comp.*, **5**, 161-175.

[Baker-1989d] T.J. BAKER (1989), Mesh generation by a sequence of transformations, *Appl. Numer. Math.*, **2**(6), 515-528.

[Baker-1991a] T.J. BAKER (1991), Shape Reconstruction and Volume Meshing for Complex Solids, *Int. j. numer. methods eng.*, **32**, 665-675.

[Baker-1991b] T.J. BAKER (1991), Single block mesh generation for a fuselage plus two lifting surfaces, *Proc. Third Int. Conf. Numerical Grid Generation*, Barcelona, Spain, 261-272.

[Baker-1992] T.J. BAKER (1992), Mesh Generation for the Computation of Flowfields over Complex Aerodynamic Shapes, *Comp. Math. Applic.*, **24**(5-6), 103-127.

[Baker-1997] T.J. BAKER (1997), mesh adaptation strategies for problems in fluid dynamics, *Finite Elements in Analysis and Design*, **25**(3-4), 243-273.

[Bakhvalov-1973] N. BAKHVALOV (1973), *Méthodes numériques*, Eds. Mir, Moscou.

[Bank *et al.* 1983] R.E. BANK, A.H. SHERMAN, A. WEISER (1983), Refinement algorithms and data structure for regular local mesh refinement, Scientific Computing, R. Stepleman et al. (eds), IMACS, North Holland.

[Bank,Chan-1986] R. E. BANK AND T.F. CHAN (1986), PLTMGC: a multigrid continuation program for parametrized nonlinear elliptic system, *Siam J. Sci. Stat. Comput.*, **7**(2), 540-559.

[Bank-1990] R. E. BANK (1990), *PLTMG : a software package for solving elliptic partial differential equations*, Frontiers in applied mathematics, Siam.

[Bank-1997] R. E. BANK (1997), Mesh smoothing using *a posteriori* estimates, *Siam J. numer. anal.*, **34**(3), 979-997.

[Bänsch-1991] E. BÄNSCH (1991), Local Mesh Refinement in 2 and 3 Dimensions, *Impact of Comp. in Sci. and Eng.*, **3**, 181-191.

[Barequet *et al.* 1996a] , G. BAREQUET, M. DICKERSON AND D. EPPSTEIN (1996), On triangulating three-dimensional polygons, *Computational Geometry'96*, 38-47, Philadelphia, PA, USA.

[Barequet *et al.* 1996b] , G. BAREQUET, B. CHAZELLE, L.J. GUIBAS, J. MITCHELL AND A. TAL (1996), Boxtree: a hierarchical representation for surfaces in 3D, *Eurographics'96*, **15**(3), 387-396.

[Barequet *et al.* 1998] G. BAREQUET, C.A. DUNCAN AND S. KUMAR (1998), RSVP: A geometric toolkit for controlled repair of solid models, *IEEE Trans. on Visualization and Computer Graphics*, **4**(2), 162-177.

[Barfield-1970] W.D. BARFIELD (1970), An optimal mesh generator for Lagrangian hydrodynamic calculations in two space dimensions, *J. Comput. Physics*, **6**, 417-429.

[Bartels *et al.* 1987] R. H. BARTELS, J.C. BEATY AND B.A. BARSKY (1987), *An Introduction to Splines for Use in Computer Graphics and Geometric Modeling*, Morgan Kaufmann.

[Bartels *et al.* 1988] R. H. BARTELS, J.C. BEATY AND B.A. BARSKY (1988), *B-splines, Mathématiques et CAO*, **6**, Hermès, Paris.

[Barth-1994] T.J. BARTH (1994), Aspects of unstructured grids and finite-volume solvers for the Euler and Navier-Stokes equations, *Technical report*, von Karman Inst. for Fluids Dynamics, Lecture Series 1994-05.

[Bathe,Chae-1989] K.J. BATHE, S.W. CHAE (1989), On automatic mesh construction and mesh refinement in Finite Element analysis, *Computers & Structures*, **32**(3/4), 911-936.

[Baumgart-1974] B.G. BAUMGART (1974), Geometric Modeling for Computer Vision, *AIM-249, CS-TR-74-463*, Stanford Artificial Intelligence Lab., Stanford University.

[Baumgart-1975] B.G. BAUMGART (1975), A Polyhedron Representation for Computer Vision., in *National Computer Conference*, Montvale, N.J.: AFIPS Press, 589-596.

[Beall,Shephard-1997] M.W. BEALL AND M.S. SHEPHARD (1997), A general topology-based mesh data structure, *Int. j. numer. methods eng.*, **40**(8), 1573-1596.

[Benzley *et al.* 1995] S.E. BENZLEY, E. PERRY, M. MERKLEY, B. CLARK AND G. SJAARDEMA, A comparison of all-hexahedral and all-tetrahedral finite element meshes for elastic and elasto-plastic analysis, *Proc 4th Int. Meshing Roundtable*, 179-191.

[deBerg *et al.* 1997] M. DE BERG, M. VAN KREVELD, M. OVERMARS AND O. SCHWAZKOPF (1997), Computational geometry: algorithms and applications, *Springer-Verlag*, Berlin Heidelberg.

[Berger-1978] M. BERGER (1978), *Géométrie tome 3 : convexes et polytopes, polyèdres réguliers, aires et volumes*, Fernand Nathan, Paris.

[Berger,Colella-1989] M.J. BERGER AND P. COLELLA (1989), Local adaptive mesh refinement for shock hydrodynamics, *J. Comput. Physics*, **82**, 64-84.

[Berger,Gostiaux-1987] M. BERGER ET B. GOSTIAUX (1987), *Géométrie différentielle : variétés, courbes et surfaces*, PUF, Paris.

[Berger,Jameson-1985] M.J. BERGER AND A. JAMESON (1985), Automatic adaptive grid refinement for Euler equations, *AIAA J.* , **23**(4), 561-568.

[Berger,Oliger-1984] M.J. BERGER AND J. OLIGER (1984), Adaptive mesh refinement for hyperbolic partial differential equations, *J. Comput. Physics*, **53**, 484-512.

[Bern *et al.* 1990] M. BERN, D. EPPSTEIN AND J. GILBERT (1994), Provably good mesh generation, *J. Comp. Sys. Sci.*, **48**, 384-409.

[Bern *et al.* 1991] M. BERN, D. EPPSTEIN AND F. YAO. (1991), The expected extremes in a Delaunay triangulation, *Int. J. Comp. Geom. & Appl.*, **1**, 79-92.

[Bern *et al.* 1991] M. BERN, H. EDELSBRUNNER, D. EPPSTEIN, S. MITCHELL AND T.S. TAN (1991), Edge insertion for optimal triangulations, *Disc. & Comp. Geom.*, **10**, 47-65.

[Bern-1995] M. BERN (1995), Compatible tetrahedralizations, *Fundamenta Informaticae*, **22**, 371-384.

[Bern,Eppstein-1995] M. BERN AND D. EPPSTEIN (1995), Mesh generation and optimal triangulation, *Computing in Euclidean Geometry*, 2nd ed., D.-Z. Du and F.K. Hwang, eds., World Scientific, 47-123.

[Bern,Eppstein-1997] M. BERN AND D. EPPSTEIN (1997), Quadrilateral meshing by circle packing, *Proc. 6th Int. Meshing Roundtable*, Park City, Utah, 7-19.

[Bern,Plassmann-1998] M. BERN AND P. PLASSMANN (1998), *Mesh generation*, Joerg Sack ed., Elsevier.

[Bernardi-1996] C. BERNARDI (1996) Estimations *a posteriori* : une mode ou un outil de base, *Matapli*, **48**, 19-30.

[Berzins-1999] M. BERZINS (1999), Mesh quality : a function of geometry, error estimates or both ?, *Engineering with Computers*, **15**, 236-247.

[Bernadou *et al.* 1988] M. BERNADOU, P.L. GEORGE, A. HASSIM, P. JOLY, P. LAUG, B. MULLER, A. PERRONNET, E. SALTEL, D. STEER, G. VANDERBORCK ET M. VIDRASCU (1988), Modulef: une bibliothèque modulaire d'éléments finis, INRIA (Eds.).

[Bézier-1986] P. BÉZIER (1986), *Courbes et surfaces, Mathématiques et CAO*, **4**, Hermès, Paris.

[Blacker,Stephenson-1991] T.D. BLACKER AND M.R. STEPHENSON (1991), Paving: a new approach to automated quadrilateral mesh generation, *Int. j. numer. methods eng.*, **32**, 811-847.

[Blacker,Meyers-1993] T.D. BLACKER AND R.J. MEYERS (1993), Seams and wedges in plastering: a 3d hexahedral mesh generation algorithm, *Engr. with computers*, **9**, 83-93.

[Blinn-1982] J.F. BLINN(1982), A generalization of algebraic surface drawing, *ACM Trans. on Graphics*, **1**(3), 235-256.

[Bloomenthal-1988] J. BLOOMENTHAL, Polygonization of implicit surfaces, *CAGD*, **5**, 341-355.

[Bloomenthal *et al.* 1997] J. BLOOMENTHAL, C. BAJAJ, J. BLINN, M.P. CANI-GASCUEL, A. ROCKWOOD, B. WYWILL, G. WYVILL (1997), Introduction to implicit surfaces, Morgan Kaufmann Publishers, San Francisco.

[Blum-1967] H. BLUM, A transformation for extracting new descriptors of shape, in *Models for the perception of speech and visual form*, W. Whaten Dunn eds., MIT Press, 362-380.

[Boissonnat-1984] J.D. BOISSONNAT (1984), Geometric structures for 3-dimensional shape representation, *ACM Transactions on Graphics*, **3**(4), 266-286.

[Boissonnat,Teillaud-1986] J.D. BOISSONNAT AND M. TEILLAUD (1986), A hierarchical representation of objects: the Delaunay Tree, *Second ACM Symp. on Comput. Geom.*, Yorktown Heights, NY.

[Boissonnat *et al.* 1992] J.D. BOISSONNAT, O. DEVILLERS, R. SCHOTT, M. TEILLAUD AND M. YVINEC (1992), Applications of Random Sampling to On-Line Algorithms in Computational Geometry, *Discrete Comput. Geom.*, **8**, 51-71.

[Boissonnat,Yvinec-1995] J.D. BOISSONNAT ET M. YVINEC (1995), *Géométrie Algorithmique*, Ediscience, Paris. Also as *Algorithmic Geometry*, Cambridge University Press, 1997.

[Boissonnat *et al.* 1999] J.D. BOISSONNAT, F. CAZALS AND S. NULLANS (1999), Smooth shape reconstruction, *Proc. 15th European Workshop on Computational Geometry*, 63-68.

[Bonet,Peraire-1991] J. BONET AND J. PERAIRE (1991), An alternate digital tree for geometric searching and interaction problems, *Int. j. numer. methods eng.*, **31**, 1-17.

[Borgers-1990] C. BORGERS (1990), Generalized Delaunay triangulations of non convex domains, *Comput. Math. Applic.*, **20**(7), 45-49.

[Borouchaki *et al.* 1995] H. BOROUCHAKI, F. HECHT, E. SALTEL AND P.L. GEORGE (1995), Reasonably efficient Delaunay based mesh generator in 3 dimensions, *Proc. 4th Int. Meshing Roundtable, Albuquerque, New Mexico*, 3-14.

[Borouchaki,George-1996a] H. BOROUCHAKI ET P.L. GEORGE (1996), Triangulation de Delaunay et métrique riemannienne. Applications aux maillages éléments finis, *Revue européenne des éléments finis*, **5**(3), 323-340.

[Borouchaki,George-1996b] H. BOROUCHAKI ET P.L. GEORGE (1996), Mailleur de Delaunay gouverné par une carte, *C. R. Acad. Sci. Paris*, t. 323, Série I, 1141-1146.

[Borouchaki *et al.* 1996] H. BOROUCHAKI, P.L. GEORGE AND S.H. LO (1996), Optimal Delaunay point insertion, *Int. j. numer. methods eng.*, **39**(20), 3407-3438.

[Borouchaki,George-1997a] H. BOROUCHAKI AND P.L. GEORGE (1997), Aspects of 2D Delaunay mesh generation, *Int. j. numer. methods eng.*, **40**, 1957-1975.

[Borouchaki,George-1997b] H. BOROUCHAKI ET P.L. GEORGE (1997), Maillage des surfaces paramétriques. Partie I: Aspects théoriques, *C.R. Acad. Sci. Paris*, t 324, Serie I, 833-837.

[Borouchaki,Laug-1996] H. BOROUCHAKI ET P. LAUG (1996), Le mailleur adaptatif bidimensionnel BL2D: manuel d'utilisation et documentation, *RT INRIA* **0185**.

[Borouchaki *et al.* 1997a] H. BOROUCHAKI, P.L. GEORGE, F. HECHT, P. LAUG AND E. SALTEL (1997), Delaunay mesh generation governed by metric specifications. Part I: Algorithms, *Finite Elements in Analysis and Design*, **25**(1-2), 61-83.

[Borouchaki *et al.* 1997b] H. BOROUCHAKI, P.L. GEORGE AND B. MOHAMMADI (1996), Delaunay mesh generation governed by metric specifications. Part II: Application examples, *Finite Elements in Analysis and Design*, **25**(1-2), 85-109.

[Borouchaki,Frey-1998] H. BOROUCHAKI AND P.J. FREY (1998), Adaptive triangular-quadrilateral mesh generation, *Int. j. numer. methods eng.*, **41**, 915-934.

[Borouchaki *et al.* 1998] H. BOROUCHAKI, F. HECHT AND P.J. FREY (1997), Mesh gradation control, *Int. j. numer. methods eng.*, **43**(6), 1143-1165.

[Borouchaki *et al.* 1999] H. BOROUCHAKI, P. LAUG AND P.L. GEORGE (1999), About parametric surface meshing, 2nd *Symposium on Trends in Unstructured Mesh Generation, USNCCM'99*, University of Colorado, Boulder, CO, USA.

[Bowyer-1981] A. BOWYER (1981), Computing Dirichlet tesselations, *The Comp. J.*, **24**(2), 162-167.

[Brière,George-1995] E. BRIÈRE DE L'ISLE AND P.L. GEORGE (1995), Optimization of tetrahedral meshes, *IMA Volumes in Mathematics and its Applications*, I. Babuska, W.D. Henshaw, J.E. Oliger, J.E. Flaherty, J.E. Hopcroft and T. Tezduyar (Eds.), **75**, 97-128.

[Bristeau,Periaux-1986] M.O. BRISTEAU AND J. PERIAUX (1986), Finite element methods for the calculation of compressible viscous flows using self-adaptive refinement, *VKI lecture notes on CFD.*

[Brown-1979] K.Q. BROWN (1979), Voronoi diagrams from convex hull, *Inform. Process. Lett.*, **9**, 223-228.

[Buratynski-1990] E.K. BURATYNSKI (1990), A fully automatic three-dimensional mesh generator for complex geometries, *Int. j. numer. methods eng.*, **30**, 931-952.

[Bykat-1976] A. BYKAT (1976), Automatic generation of triangular grid-subdivision of a general polygon into convex subregions. Part II: triangulation of convex polygons, *Int. j. numer. methods eng.*, **10**, 1329-1342.

[Canann et al.1993] S. CANANN, M. STEPHENSON AND T. BLACKER (1993), Optismoothing: An optimization-driven approach to mesh smoothing, *Finite Elements in Analysis and Design*, **13**, 185-190.

[Canann et al.1996] S. CANANN, S.N. MUTHUKRISHNAN AND R.K. PHILIPS (1996), Topological refinement procedures for triangular finite element meshes, *Engineering with Computers*, **12**, 243-255.

[Carey-1997] G.F. CAREY (1997), Computational grids : generation, adaptation and solution strategies, *Taylor and Francis.*

[Carey,Oden-1984] G.F. CAREY AND J.T. ODEN (1984), Finite Elements: computational aspects, *Prentice-Hall.*

[doCarmo-1976] M.P. DO CARMO (1976), Differential geometry of curves and surfaces, *Prentice Hall*, New Jersey.

[Carnet-1978] J. CARNET (1978), Une méthode heuristique de maillage dans le plan pour la mise en œuvre des éléments finis, Thèse, Paris.

[Cass et al.1996] R.J. CASS, S.E. BENTLEY, R. MEYERS AND T.J. BAKER (1996), Generalized 3D paving: an automated quadrilateral surface mesh generation algorithm, *Int. j. numer. methods eng.*, **39**, 1475-1489.

[Castro-1994] M.J. CASTRO-DIAZ (1994), Mesh refinement over triangulated surfaces, *RR INRIA* **2462**.

[Castro,Hecht-1995] M.J. CASTRO-DIAZ AND F. HECHT (1995), Anisotropic surface mesh generation, *RR INRIA* **2672**.

[Castro *et al.* 1995] M.J. CASTRO-DIAZ, F. HECHT AND B. MOHAMMADI (1995), New Progress in Anisotropic Mesh Adaption for Inviside and Viscous Flow Simulations, *RR INRIA* **2671**.

[Catmull-1974] E. CATMULL (1974), A subdivision Algorithm for Computer Design of curved Surfaces, Univ. Utah Comp. Sci. Dept., UTEC-CSC,74-133.

[Cavendish-1974] J.C. CAVENDISH (1974), Automatic Triangulation of Arbitrary Planar Domains for the Finite Element Method, *Int. j. numer. methods eng.*, **8**, 679-696.

[Cavendish-1975] J.C. CAVENDISH (1975), Local mesh refinement using rectangular blended finite elements, *J. of Comp. Phys.*, **19**, 211-228.

[Cavendish *et al.* 1985] J.C. CAVENDISH, D.A. FIELD AND W.H. FREY (1985), An approach to automatic three-dimensional finite element mesh generation, *Int. j. numer. methods eng.*, **21**, 329-347.

[Cazals,Puech-1997] F. CAZALS, C. PUECH (1997), Bucket-like space partitioning data structures with applications to ray-tracing, *13th ACM Synposium on Computational Geometry*, Nice.

[Celeux *et al.* 1989] G. CELEUX, E. DIDAY, G. GOVAERT, Y LECHEVALLIER ET H. RALAMBONDRAINY (1989), *Classification automatique des données*, Dunod, Paris.

[Cendes *et al.* 1985] Z.J. CENDES, D.N. SHENTON AND H. SHAHNASSER (1985), Magnetic field computations using Delaunay triangulations and complementary finite element methods, *IEEE Trans. Magnetics*, **21**.

[Cendes,Shenton-1985a] Z.J. CENDES, D.N. SHENTON (1985), Adaptive mesh refinement in the finite element computation of magnetic fields, *IEEE Trans. Magnetics*, **21**.

[Cendes,Shenton-1985b] Z.J. CENDES, D.N. SHENTON (1985), Complementary error bounds for foolproof finite element mesh generation, *Math. and Comp. in Simulation*, **27**, 295-305, North Holland.

[Cendes,Shenton-1985c] Z.J. CENDES, D.N. SHENTON (1985), Three-dimensional finite element mesh generation using Delaunay tesselation, *IEEE Trans. Magnetics*, **21**, 2535-2538.

[Chan,Anatasiou-1997] C.T. CHAN AND K. ANATASIOU (1997), An automatic tetrahedral mesh generation scheme by the advancing-front method, *Commun. numer. methods eng.*, **13**, 33-46.

[Chatelin-1993] F. CHATELIN(1993), *Eigenvalues of matrices*, Wiley.

[Chazelle-1984] B. CHAZELLE (1984), Convex partitions of polyhedra: a lower bound and worst-case optimal algorithm, *SIAM J. Comput*, **13**, 488-507.

[Chazelle,Palios-1990] B. CHAZELLE AND L. PALIOS (1990), Triangulating a nonconvex polytope, *Discrete Comput. Geom.*, **5**, 505-526.

[Chazelle-1991] B. CHAZELLE (1991), Triangulating a simple polygon in linear time, *Discrete Comput. Geom.*, **6**, 485-524.

[Chen,Schmitt-1992] X. CHEN AND F. SCHMITT (1992), Intrinsic surface properties from surface triangulation, *Rapport de Recherche*, ENST-92-D-004.

[Cherfils,Hermeline-1990] C. CHERFILS AND F. HERMELINE (1990), Diagonal swap procedures and characterizations of 2D-Delaunay triangulations, *Rairo, Math. Mod. and Num. Anal.*, **24**(5), 613-626.

[Chew,Drysdale-1985] L.P. CHEW AND R.L. DRYSDALE (1985), Voronoï diagrams based on convex distance functions, *ACM 0-89791-163-6*, 235-244.

[Chew-1989a] L.P. CHEW (1989), Constrained Delaunay Triangulations, *Algorithmica*, **4**, 97-108.

[Chew-1989b] L.P. CHEW (1989), Guaranteed-Quality Triangular Meshes, *Dept. of Computer Science Tech Report 89-983*, Cornell University.

[Chew-1993] L.P. CHEW (1993), Guaranteed-Quality Mesh Generation for Curved Surfaces, *Proc. 9th ACM Computational Geometry*, 274-280.

[Chew *et al.* 1997] L.P. CHEW, N. CHRISOCHOIDES AND F. SUKUPP (1997), Parallel constrained Delaunay meshing, *Joint ASME/ASCE/SES Summer Meeting McNu'97*, Northwestern Univ., USA, June 29-July 2, 89-96.

[Chrisochoides,Sukupp-1996] N. CHRISOCHOIDES AND F. SUKUPP (1996), Task parallel implementation of the Bowyer-Watson algorithm, *Proc. 5th Int. Conf. Numerical grid Generation in Computational Field Simulations*, Mississippi State Univ., 773-782.

[Ciarlet-1978] P.G. CIARLET (1978), *The Finite Element Method*, North Holland.

[Ciarlet-1986] P.G. CIARLET (1986), *Elasticité Tridimensionnelle*, RMA 1, Masson, Paris.

[Ciarlet-1988] P.G. CIARLET (1988), *Mathematical Elasticity, Vol 1: Three-Dimensional Elasticity*, North-Holland.

[Ciarlet-1991] P.G. CIARLET (1991), Basic Error Estimates for Elliptic Problems, in Handbook of Numerical Analysis, vol II, Finite Element methods (Part 1), P.G. Ciarlet and J.L. Lions Eds, North Holland, 17-352.

[Ciarlet,Raviart-1973] P.G. CIARLET AND P.A. RAVIART (1973), Maximum principle and uniform convergence for the finite element method, *Computer Methods in Appl. Mechanics and Engineering*, **2**, 17-31.

[Cignoni *et al.* 1996] P. CIGNONI, C. MONTANI, E. PUPPO AND R. SCOPIGNO (1996), Optimal isosurface extraction from irregular volume data, *IEEE/ACM Symp. on Volume Visualization*, San Francisco, CA.

[Clarkson-1987] K. CLARKSON (1987), New applications of random sampling in computational geometry, *Discrete Comput. Geom.*, **2**, 195-222.

[Clarkson,Shor-1989] K. CLARKSON AND P.W. SHOR (1989), Applications of random sampling in computational geometry, *Discrete Comput. Geom.*, **4**, 387-421.

[Clarkson *et al.* 1989] K. CLARKSON, R.E. TARJAN AND C.J. VAN WYK (1989), A fast Las Vegas algorithm for triangulating a simple polygon, *Discrete Comput. Geom.*, **4**, 423-432.

[Cook-1974] W.A. COOK (1974), Body oriented coordinates for generating 3-dimensional meshes, *Int. j. numer. methods eng.*, **8**, 27-43.

[Coorevits *et al.* 1995] P. COOREVITS, P. LADEVÈZE AND J.P. PELLE (1995), Au automatic procedure for finite element analysis in 2d elasticity, *Comput. Methods Appl. Mech. Engrg.*, **121**, 91-120.

[Cormen *et al.* 1990] T. CORMEN, C. LEISERSON AND R. RIVEST (1990), Introduction to Algorithms, *MIT Press.*

[deCougny *et al.* 1990] H. DECOUGNY, M.S. SHEPHARD AND M.K. GEORGES (1990), Explicit Node Point smoothing within Octree, *Scorec Report* **10**, RPI, Troy, NY.

[deCougny,Shephard 1996] H.L. DECOUGNY AND M.S. SHEPHARD (1996), Surface Meshing Using Vertex Insertion, *Proc. 5th Int. Meshing Roundtable*, October 10-11, Pittsburgh, PA, 243-256.

[deCougny *et al.* 1996] H. DECOUGNY, M.S. SHEPHARD AND C. ÖZTURAN (1996), Parallel three-dimensional mesh generation on distributed memory MIMD computers, *Engineering with Computers*, **12**, 94-106.

[deCougny-1997] H.L. DECOUGNY (1997), Distributed parallel mesh generation, *PhD thesis*, Scorec, Rensselaer Polytechnic Inst., Troy, NY.

[deCougny-1998] H.L. DECOUGNY (1998), Refinement and coarsening of surface meshes, *Engineering with Computers*, **14**, 214-222.

[deCougny,Shephard-1999] H.L. DECOUGNY AND M.S. SHEPHARD (1999), Parallel refinement and coarsening of tetrahedral meshes, *Int. j. numer. methods eng.*, **46**, 1101-1125.

[Coulomb-1987] J.L. COULOMB (1987), Maillage 2D et 3D. Experimentation de la triangulation de Delaunay, *Conf. on Automated mesh generation and adaptation*, Grenoble.

[Coupez-1991] T. COUPEZ (1991), Grandes transformations et remaillage automatique, Thèse ENSMP, CEMEF, Sophia Antipolis.

[Coupez-1996] T. COUPEZ (1996), Parallel Adaptive Remeshing in 3D Moving Mesh Finite Element, *5th Int. Conf. on Grid Generation in Comp. Field Simulations*, MS, USA, 783-792.

[Coupez-1997] T. COUPEZ (1997), Mesh generation and adaptive remeshing by a local optimization principle, *Proc. Nafems World Congress'97*, Stuttgart, 9-11 april.

[Coxeter *et al.* 1959] H.S.M. COXETER, L. FEW AND C.A. ROGERS (1959), Covering space with equal spheres, *Mathematika*, **6**, 147-151.

[Coxeter-1963] H.S.M. COXETER (1963), Regular polytopes, *Macmillan*, New York.

[Cullum,Willoughby-1985] J. CULLUM AND R.A. WILLOUGHBY (1985), Lanczos algorithms for large symmetric eigenvalue computations, in *Progress in Scientific Computing*, **3-4**, Biekhauser, London.

[Cuthill-1972] E. CUTHILL (1972), Several strategies for reducing the bandwidth of sparse symmetric matrices, in *Sparse matrices and their applications*, Plenum Press, New York.

[Cuthill,McKee-1969] E. CUTHILL, J. MCKEE (1969), Reducing the bandwidth of sparse symmetric matrices, *Proc. 24th Nat. Conf. Assoc. Comput. Mach.*, 157-172.

[Dahlquist,Bjorck-1974] G. DAHLQUIST AND A. BJORCK (1974), Numerical Methods, *Prentice-Hall*, Englewood Cliffs, N.J.

[Dannelongue,Tanguy-1990] H. DANNELONGUE AND P.A. TANGUY (1990), Efficient data structures for adaptive remeshing with the FEM, *J. Comput. Physics*, **91**, 94-109.

[Dannelongue,Tanguy-1991] H. DANNELONGUE AND P.A. TANGUY (1991), Three-dimensional adaptive finite element computations and applications to non-Newtonian flows, *Int. j. numer. methods fluids*, **13**, 145-165.

[Delaunay-1934] B. DELAUNAY (1934), Sur la sphère vide, *Bul. Acad. Sci. URSS, Class. Sci. Nat.*, 793-800.

[Demengel,Pouget-1998] G. DEMENGEL, J.P. POUGET (1998), Modèles de Bézier, des B-Splines et des NURBS, *Ellipses*, Paris.

[Deriche-1987] R. DERICHE (1987), Using Canny's criteria to derive a recursively implemented optimal edge detector, *Int. J. Comput. Vision*, 167-187.

[Désidéri-1998] J.A DÉSIDÉRI (1998), *Modèles discrets et schémas itératifs*, Hermès.

[Devillers-1992] O. DEVILLERS (1992), Randomization yields simple $O(nlog^*n)$ algorithms for difficult $\Omega(n)$ problems, *Internat. J. Comput. Geom. Appl.*, **2**(1), 97-111.

[Devillers-1997] O. DEVILLERS (1997), Improved incremental randomized Delaunay triangulation, *RR INRIA*, **3298**.

[Devillers *et al.* 1992] O. DEVILLERS, S. MEISER AND M. TEILLAUD (1992), Fully dynamic Delaunay triangulation in logarithmic expected time per operation, *Comput. Theory Appl.* **2**(2), 55-80.

[Devillers-Preparata 1998] O. DEVILLERS, AND F. PREPARATA (1998), A probalistic analysis of the power of arithmetic filters, *Discrete and Computational geometry.*, to appear.

[Devroye-1986] L. DEVROYE (1986), Lecture Notes on Bucket Algorithms, *Birkauser.*

[Dey-1997] S. DEY (1997), Geometry-Based Three-Dimensional *hp*-Finite Element Modelling And Computations, *PhD thesis*, Scorec, Rensselaer Polytechnic Inst., Troy, NY.

[Dey *et al.* 1997] S. DEY, M.S. SHEPHARD AND J.E. FLAHERTY (1997), Geometry representation issues associated with p-version finite lement computations, *Comp. Meth. Appl. Mech. and Eng.*, **150**(1-4), 39-56.

[Dirichlet-1850] G.L. DIRICHLET (1850), Über die reduction der positiven quadratischen formen mit drei understimmten ganzen zahlen, *Z. Angew Math. Mech.*, **40**(3), 209-227.

[Doi,Koide-1991] A. DOI AND A. KOIDE (1991), An efficient method of triangulating equi-valued surfaces by using tetrahedral cells, *IEICE Trans.*, **E74**(1), 214-224.

[Dolenc,Makela-1990] A. DOLENC AND I. MÄKELÄ (1990), Optimized triangulation of parametric surfaces, *Mathematics of Surfaces*, **IV**.

[Donaghy *et al.* 1996] R.J. DONAGHY, W.M. CUNE, S.J. BRIDGETT, C.G. ARMSTRONG, D.J. ROBINSON AND R.M. KEAG (1996), Dimensional reduction of analysis models, *Proc. 5th Int. meshing Roundtable*, Pittsburgh, PA, 307-320.

[Donath,Hoffman-1972] W.E. DONATH AND A.J. HOFFMAN (1972), Algorithms for Partitioning of Graphs and Computer Logic based on Eigenvectors of Connection matrices, *IBM Technical Disclosure Bulletin*, **15**, 938-944.

[Donath-1973] W.E. DONATH (1973), Lower Bounds for the partitioning of Graphs, *IBM J. Res. Develop.*, **17**, 420-425.

[Dürst-1988] M.J. DÜRST (1988), Letters: additional reference to "Marching Cubes", *ACM Comp. Graphics*, **22**(4), 72-73.

[Dyn,Goren-1993] N. DYN AND I. GOREN (1993), Transforming triangulations in polygonal domains, *Computer Aided Geometric Design*, **10**, 531-536.

[Dwyer-1987] R.A. DWYER (1987), A faster Divide-and-Conquer algorithm for constructing Delaunay triangulations, *Algorithmica*, **2**, 137-151.

[Dwyer-1991] R.A. DWYER (1991), Higher-Dimensional Voronoï diagrams in linear expected time, *Discrete Comput. Geom.*, **6**, 342-367.

[Ecer et al. 1985] A. ECER, J.T. SPYROPOULOS, J.D. MAUL (1985), A three-dimensional block-structured Finite Element Grid generation scheme, *AIAA J.*, **23**, 1483-1490.

[Edelsbrunner-1987] H. EDELSBRUNNER (1987), *Algorithms in Combinatorial Geometry*, **10**, EATCS Monographs on Theoretical Computer Science, Springler-Verlag.

[Edelsbrunner *et al.* 1990] H. EDELSBRUNNER, F.P. PREPARATA AND D.B. WEST (1990), Tetrahedrizing point sets in three dimensions, *J. Symbolic Computation*, **10**, 335-347.

[Eiseman-1979] P.R. EISEMAN (1979), A multi-surface method of coordinate generation, *J. Comput. Phys.*, **33**.

[Eiseman-1985] P.R. EISEMAN (1985), Alternating direction adaptive grid generation, *AIAA J.*, **23**.

[Eiseman,Erlebacher-1987] P.R. EISEMAN AND G. ERLEBACHER, Grid generation for the solution of partial differential equations, *ICASE report 87-57*.

[Eppstein-1992] D. EPPSTEIN (1992), The farthest point Delaunay triangulation minimizes angles, *Comp. Geom. Theory & Applications*, **1**, 143-148.

[Eriksson-1982] L.E. ERIKSSON (1982), Generation of boundary-conforming grids around wing-body configurations using transfinite interpolation, *AIAA J.*, **20**.

[Eriksson-1983] L.E. ERIKSSON (1983), Practical three-dimensional mesh generation using transfinite interpolation, Von Karman Inst. for Fluids Dynamics, Lecture Series Notes.

[Eriksson-1984] L.E. ERIKSSON (1984), Transfinite mesh generation and Computer-Aided-Analysis of mesh effects, *PhD thesis*, Universitetshuset, Uppsala, Sweden.

[Faddeev *et al.* 1992] D.K. FADDEEV, N.P. DOLBILIN, S.S. RYSHKOV AND M.I. SHTOGRIN (1992), B.N. Delone (on his life and creative work), *Proc. the Steklov Inst. of Math.*, **4**, 1-9.

[Farestam,Simpson-1993] S. FARESTAM AND R.B. SIMPSON (1993), On correctness and efficiency for advancing-front techniques of finite element mesh generation, *RR*, Univ. Waterloo, Canada.

[Farhat,Lesoinne-1993] C. FARHAT AND M. LESOINNE (1993), Automatic partitioning of unstructured meshes for the parallel solution of problems in computational mechanics, *Int. j. numer. methods eng.*, **36**, 745-764.

[Farin-1983] G. FARIN (1983), Smooth interpolation to scattered 3D data, in R.E. Barnhill and W. Boehm (Eds.) *Surfaces in Computer Aided Geometric Design*, North-Holland, 43-63.

[Farin-1985] G. FARIN (1985), A modified Clough-Tocher interpolant, *Computer Aided Geometric Design*, **2**, 19-27.

[Farin-1986] G. FARIN (1986), Triangular Bernstein-Bézier patches, *Computer Aided Geometric Design*, **3**(2), 83-127.

[Farin-1987] G. FARIN (1987), Geometric Modeling : Algorithms and new trends, SIAM.

[Farin-1997] G. FARIN (1997), Curves and surfaces for CAGD. A practical guide. Academic Press.

[Faroukin,Rajan-1987] R.T. FAROUKI AND V.T. RAJAN (1987), On the numerical condition of algebraic curves and surfaces. 1 : implicit equations, *IBM Research Report*, **RC 13263**.

[Fezoui *et al.* 1991] L. FEZOUI, F. LORIOT, M. LORIOT AND J. RÉGÈRE J. (1991), A 2D Finite volume/Finite element Euler Solver on a MIMD parallel machine, *Proc. 2nd Symposium on High Performance Computing*, Montpellier, France.

[Field-1988] D.A. FIELD (1988), Laplacian smoothing and Delaunay triangulations, *Commun. numer. methods eng.*, **4**, 709-712.

[Field,Smith-1991] D.A. FIELD AND W.D. SMITH (1991), Graded tetrahedral finite element meshes, *Int. j. numer. methods eng.*, **31**, 413-425.

[Field,Yarnall-1989] D.A. FIELD AND K. YARNALL (1989), Three dimensional Delaunay triangulations on a Cray X-MP, in Supercomputing 88, vol 2, Science et Applications, *IEEE C.S. and ACM Sigarch.*

[Field-1995] D.A. FIELD (1995), The legacy of automatic mesh generation from solid modeling, *Computer Aided Geometric Design*, **12**, 651-673.

[Field-2000] D.A. FIELD (2000), Quality measures for initial meshes, *Int. j. numer. methods eng.*, **47**, 887-906.

[Fielder-1975] M. FIELDER (1975), A Property of Eigenvectors of Nonnegative Symmetric Matrices and its Application to Graph Theory, *Czech. Math.*, **25**, 619-633.

[deFigueiredo *et al.* 1992] L.H. DEFIGUEIREDO, J.DEMIRANDA GOMEZ, D. TERZOPOULOS AND L. VELHO (1992), Physically-based methods for polygonization of implicit surfaces, in *proc. Graphics Interfaces'92*, 250-257.

[deFigueiredo-1992] L.H. DEFIGUEIREDO (1992), Computational morphology of implicit curves, *PhD thesis*, IMPA, Brazil.

[Filip *et al.* 1986] D. FILIP, R. MAGEDSON AND R. MARKOT (1986), Surface algorithm using bounds on derivatives, *Computer Aided Geometric Design*, **3**, 295-311.

[Filipiak-1996] M. Filipiak (1986), Mesh generation, Edinburgh Parallel Comp. Center, University of Edinburgh.

[Fiorot,Jeannin-1989] J.C. Fiorot, P. Jeannin(1989), *Courbes et Surfaces Rationnelles: Applications à la CAO*, RMA 12, Masson, Paris.

[Folwell,Mitchell-1998] N.T. Folwell and S.A. Mitchell (1998), Reliable Whisker Weaving via curve contraction, *Proc. 7th International Meshing Roundtable*, 365-378, Detroit, MI.

[Fortune-1987] S.J. Fortune (1987), A sweepline algorithm for Voronoi diagrams, *Algorithmica*, **2**(2), 153-174.

[Fortune-1992] S.J. Fortune (1992), Voronoï diagrams and Delaunay triangulations. In D.Z. Du and F.K. Hwand, eds., *Computing in Euclidean Geometry*, vol 1 of *Lecture Notes Series on Computing*, World Scientific, Singapore, 193-233.

[Fortune,Van Wyk-1993] S.J. Fortune and C.J. Van Wyk (1993), Efficient exact arithmetic for computational geometry, *Proc. 9th ACM Sympos. Comput. Geom.*, 163-172.

[Freitag,Gooch-1996] L. Freitag and C.O. Gooch (1996), A comparison of tetrahedral mesh improvement techniques, *Proc. 5th International Meshing Roundtable*, Pittsburgh, PA, 87-100.

[Freitag,Gooch-1997] L. Freitag and C.O. Gooch (1996), The effect of mesh quality on solution efficiency, *Proc. 6th International Meshing Roundtable*, Park City, UT, 249.

[Freudenthal-1942] H. Freudenthal (1942), Simplizialzerlegungen von Beschränker Flachheit, *Annals of Mathematics*, **43**, 580-582.

[Frey-1987] W.H. Frey (1987), Selective refinement procedure: a new strategy for automatic node placement in graded triangular meshes, *Int. j. numer. methods eng.*, **24**, 2183-2200.

[Frey,Field-1991] W.H. Frey and D.A. Field (1991), Mesh relaxation: a new technique for improving triangulations, *Int. j. numer. methods eng.*, **31**, 1121-1131.

[Frey-1993] P.J. Frey (1993), Génération automatique de maillages 3D dans des ensembles discrets. Application biomédicale aux méthodes d'éléments finis. Thèse, Université de Strasbourg I.

[Frey *et al.* 1994] P.J. Frey, B. Sarter and M. Gautherie (1994), Fully automatic mesh generation for 3-D domains based upon voxel sets, *Int. j. numer. methods eng.*, **37**, 2735-2753.

[Frey *et al.* 1996] P.J. Frey, H. Borouchaki and P.L. George (1996), Delaunay tetrahedralization using an advancing-front approach, *Proc. 5th International Meshing Roundtable*, Pittsburgh, PA, 31-43.

[Frey *et al.* 1998] P.J. Frey, H. Borouchaki and P.L. George (1998), 3D Delaunay mesh generation coupled with an advancing-front approach, *Comput. Methods Appl. Mech. Engrg.*, **157**, 115-131.

[Frey,Borouchaki-1996] P.J. Frey et H. Borouchaki (1996), Triangulation des surfaces implicites, *C.R. Acad. Sci. Paris*, t. 325, Serie I, 101-106.

[Frey,Borouchaki-1996] P.J. Frey et H. Borouchaki (1996), Texel: triangulation des surfaces implicites. Partie I: aspects théoriques, *RR INRIA*, **3066**.

[Frey,Borouchaki-1997] P.J. Frey and H. Borouchaki (1997), Unit surface mesh simplification, *Joint ASME/ASCE/SES Summer Meeting McNu'97*, Northwestern Univ., USA, June 29-July 2.

[Frey,Borouchaki-1998] P.J. Frey and H. Borouchaki (1998), Geometric evaluation of finite element surface meshes, *Finite Element in Analysis and Design*, **31**, 33-53.

[Frey,Borouchaki-1998] P.J.Frey and H. Borouchaki (1998), Geometric surface mesh optimization, *Computing and Visualization in Science*, **1**, 113-121.

[Frey,Maréchal-1998] P.J. Frey and L. Maréchal (1998), Fast Adaptive Quadtree Mesh Generation, *Proc. 7th International Meshing Roundtable*, 211-224, Detroit, MI.

[Frey,Borouchaki-1999] P.J. Frey and H. Borouchaki(1999), Surface mesh quality evaluation, *Int. J. Numer. Methods Engng.*, **45**.

[Frykestig-1994] J. Frykestig (1994), Advancing-front mesh generation techniques with application to the finite element method, *Research Report* **94-10**, Chalmers Univ. of Technology, Göteborg, Sweden.

[Galtier-1997] J. Galtier (1997), Structures de données irrégulières et architectures haute performance. Une étude du calcul numérique intensif par le partitionnement de grappes, Thèse, Université de Versailles Saint-Quentin.

[Galtier,George-1996] J. Galtier and P.L. George (1996), Prepartitioning as a way to mesh subdomains in parallel, *Proc. 5th International Meshing Roundtable*, Pittsburg, PA, 107-121.

[Garey *et al.* 1978a] M.R. Garey, D.S. Johnson, F.P. Preparata and R.E. Tarjan (1978), Triangulating a simple polygon, *Inform. Proc. Letters*, **7**(4), 175-180.

[Gargantini-1982] I. Gargantini (1982) Linear octtrees for fast processing of three-dimensional objects, *CVGIP*, **20**, 365-374.

[Garimella,Shephard-1998] R.V. Garimella and M.S. Shephard (1998), Boundary layer meshing for viscous flows in complex domains, *Proc. 7th International Meshing Roundtable*, Dearborn, MI, 107-118.

[Gaudel *et al.* 1987] M.C. Gaudel, M. Soria et C. Froidevaux (1987), *Types de Données et Algorithmes : Recherche, Tri, Algorithmes sur les Graphes*, collection didactique INRIA, **4**(2).

[vanGelder,Wilhelms-1992] A. vanGelder and J. Wilhelms (1992), Octrees for faster isosurface generation, *Trans. on Graphics*, **11**(3), 201-227.

[A.George-1971] J.A. GEORGE (1971), Computer implementation of the finite element method, *PhD thesis*, Dept. of Computer Science, Stanford Univ.

[A.George-1973] J.A. GEORGE (1973), Nested dissection of a regular finite element mesh, *SIAM J. Num. Anal.*, **10**, 1345-367.

[A.George,Liu-1978] J.A. GEORGE, J.W. LIU (1978), An automatic nested dissection algorithm for irregular finite element problems, *SIAM J. Num. Anal.*, **15**, 1053-1069.

[A.George,Liu-1979] J.A. GEORGE, J.W. LIU (1979), An implementation of a pseudoperipheral node finder, *ACM Tras. Math. Software*, **5**(3), 284-295.

[A.George,Liu-1981] J.A. GEORGE, J.W. LIU (1981), *Computer solution of large sparse positive definite systems*, Prentice Hall.

[George-1991] P.L. GEORGE (1991), *Génération automatique de maillages. Applications aux méthodes d'éléments finis*, RMA 16, Masson, Paris. Also as P.L. GEORGE (1991), *Automatic mesh generation. Applications to finite element methods*, Wiley.

[George-1993] P.L. GEORGE (1993), Construction et Modification de Maillages, *Guide Modulef* **3**.

[George-1996a] P.L. GEORGE (1996), Automatic Mesh Generation and Finite Element Computation, in Handbook of Numerical Analysis, vol IV, Finite Element methods (Part 2), Numerical Methods for Solids (Part 2), P.G. Ciarlet and J.L. Lions Eds, North Holland, 69-190.

[George-1997] P.L. GEORGE (1997), Improvement on Delaunay based 3D automatic mesh generator, *Finite Elements in Analysis and Design*, **25**(3-4), 297-317.

[George *et al.* 1990] P.L. GEORGE, F. HECHT AND E. SALTEL (1990), Fully automatic mesh generator for 3d domains of any shape, *Impact of Comp. in Sci. and Eng.*, **2**, 187-218.

[George *et al.* 1991a] P.L. GEORGE, F. HECHT AND E. SALTEL (1991), Automatic mesh generator with specified boundary, *Comput. Methods Appl. Mech. Engrg.*, **92**, 269-288.

[George *et al.* 1991b] P.L. GEORGE, F. HECHT AND M. G. VALLET (1991), Création of internal points in Voronoi's type method, Control and adaptation, *Adv. in Eng. Soft.*, **13**(5/6), 303-313.

[George,Hermeline-1992] P.L. GEORGE AND F. HERMELINE (1992), Delaunay's mesh of a convex polyhedron in dimension d. Application to arbitrary polyhedra, *Int. j. numer. methods eng.*, **33**, 975-995.

[George,Seveno-1994] P.L. GEORGE AND E. SÉVENO (1994), The advancing-front mesh generation method revisited, *Int. j. numer. methods eng.*, **37**, 3605-3619.

[George,Borouchaki-1997] P.L. GEORGE ET H. BOROUCHAKI (1997), *Triangulation de Delaunay et maillage. Applications aux éléments finis*, Hermès, Paris. Also as
P.L. GEORGE AND H. BOROUCHAKI (1998), *Delaunay triangulation and meshing. Applications to Finite Elements*, Hermès, Paris.

[George,Borouchaki-1998] P.L. GEORGE ET H. BOROUCHAKI (1998), Génération automatique de maillages tridimensionnels respectant une carte de taille, *Revue européenne des éléments finis* **7**(4), 339-363.

[Gervasio *et al.* 1997] P. GERVASIO, E. OVTCHINNIKOV AND A. QUARTERONI (1997), The Spectral Projection Decomposition Method for Elliptic Equations in Two Dimensions, *SIAM Journal on Numerical Analysis*, **34**(4), 1616-1639.

[Gibbs *et al.* 1976a] N.E. GIBBS, W.G. POOLE, P.K. STOCKMEYER (1976), An algorithm for reducing the bandwidth and profile of a sparse matrix, *SIAM J. Num. Anal.*, **13**(2), 236-250.

[Gibbs *et al.* 1976b] N.E. GIBBS, W.G. POOLE, P.K. STOCKMEYER (1976), A comparison of several bandwidth and profile reduction algorithm, *ACM Trans. on Math. Software*, **2**, 322-330.

[Glowinski-1973] R. GLOWINSKI (1973), Approximations externes par éléments finis de lagrange d'ordre un et deux du problème de Dirichlet pour l'opérateur biharmonique. Méthode itérative de résolution des problèmes approchés, *Topics in numerical analysis*, J.J.H. Miller ed., Academic Press, 123-171.

[Goldberg-1991] D. GOLDBERG (1991), What every computer scientist should know about floating-point arithmetic, *ACM Computing surveys*, **23**(1).

[Golgolab-1989] A. GOLGOLAB (1989), Mailleur tridimensionnel automatique pour des géométries complexes, *RR INRIA* **1004**.

[Golub,VanLoan-1983] G.H. GOLUB AND C.F. VAN LOAN (1983), *Matrix computations*, John Hopkins Univ. Press.

[Gonnet *et al.* 1991] G. GONNET AND R. BAEZA-YATES (1991), *Handbook of Algorithms and Data-Structures*, Addison-Wesley.

[Gordon,Hall-1973] W.J. GORDON, C.A. HALL (1973), Construction of curvilinear coordinate systems and applications to mesh generation, *Int. j. numer. Methods eng.*, **7**, 461-477.

[Green,Sibson-1978] P. GREEN AND R. SIBSON (1978), Computing Dirichlet tesselations in the plane, *Comput. J.*, **21**, 168-173.

[Gregory-1974] J.A. GREGORY (1974), Smooth interpolation without twist constraints, in R.E. Barnhill and R.F. Riesenfed (Eds.), *Computer Aided Geometric Design*, Academic Press, 71-87.

[Grice *et al.* 1988] K.R. GRICE, M.S. SHEPHARD, C.M. GRAICHEN (1988), Automatic, topologically correct, three-dimensional mesh generation by the finite octree technique, *Technical Report*, RPI Center for Interactive Computer Graphics.

[Grooms-1972] H.R. GROOMS (1972), Algorithm for matrix bandwidth reduction, *J. of Structural division*, **98**, no. ST1, 203-214.

[Grunbaum-1967] B. GRUNBAUM (1967), *Convex Polytopes*, Wiley, New York.

[Guéziec,Hummel-1995] A. GUÉZIEC AND R. HUMMEL 1995), Exploiting triangulated surface extraction using tetrahedral decomposition, *IEEE Trans. on Visualization and Comput. Graphics*, **1**(4), 328-342.

[Guibas *et al.* 1989] L.J. GUIBAS, D. SALESIN AND J. STOLFI (1989), Epsilon geometry: building robust algorithm from imprecise computations, *Proc. 5th ACM Sympos. Comput. Geom.*, 208-217.

[Guibas *et al.* 1992] L.J. GUIBAS, D.E. KNUTH AND M. SHARIR (1992), Randomized incremental construction of Delaunay and Voronoï diagrams, *Algorithmica*, **7**, 381-413.

[Hackbush,Trottenberg-1982] W. HACKBUSCH, U. TROTTENBERG (1982), Multigrid methods, *Lect. notes in Mathematics* **960**, Springer Verlag.

[Hacon,Tomei-1989] D. HACON, C. TOMEI (1989), Tetrahedral decompositions of hexahedral meshes, *Europ. J. Combinatorics*, **10**, 435-443.

[Hall-1976] C.A. HALL (1976), Transfinite interpolation and applications to engineering problems, *Law and Sahney, Theory of approximation*, Academic Press, 308-331.

[Hamann-1993] B. HAMANN (1993), Curvature approximation for triangulated surfaces, in *Geometric Modelling*, Computing Suppl. 8, Farin, Hagen, Noltmeier and Knodel eds., Springer, NY.

[Hansbo-1995] P. HANSBO (1995), Generalized Laplacian smoothing of unstructured grids, *Comm. numer. methods eng.*, **11**, 455-464.

[Hart-1993] J. HART (1993), Sphere tracking: a geometric method for the antialiased ray tracing of implicit surfaces, *Research Report*, EECS-93-015, Washington State Univ.

[Hartmann-1990] E. HARTMANN (1990), Blending of implicit surfaces with functional splines, *Comput. Aided Design*, **22**, 500-506.

[Hartmann-1998a] E. HARTMANN (1998), A marching method for the triangulation of surfaces, *The Visual Computer*, **14**, 95-108.

[Hartmann-1998b] E. HARTMANN (1998), Numerical implicitization for interpolation and G^n-continuous blending of surfaces, *Computer Aided Geometric Design*, **15**, 377-397.

[Hassan *et al.* 1996] O. HASSAN, K. MORGAN, E.J. PROBERT AND J. PERAIRE (1996), Unstructured tetrahedral mesh generation for three-dimensional viscous flows, *Int. j. numer. methods eng.*, **39**, 549-567.

[Hauser,Taylor-1986] J. HAUSER, C. TAYLOR (1986), *Numerical grid generation in computational fluid dynamics*, Pinebridge Press.

[Hecht,Saltel-1989] F. HECHT ET E. SALTEL (1990), Emc2 : Un logiciel d'édition de maillages et de contours bidimensionnels, *RT INRIA* **118**.

[Hermeline-1980] F. HERMELINE (1980), Une méthode automatique de maillage en dimension n, Thèse, Université Paris VI, Paris.

[Hermeline-1982] F. HERMELINE (1982), Triangulation automatique d'un polyèdre en dimension N, *Rairo, Analyse numérique*, **16**(3), 211-242.

[Ho Le-1988] K. HO LE (1988), Finite element mesh generation methods: a review and classification, *Comp. Aided Design*, **20**, 27-38.

[Hoffmann-1993] C. M. HOFFMANN, Implicit curves and surfaces in CAGD, *IEEE Comp. Graphics Appl.*, **13**, 79-88.

[Hoit,Garcelon-1989] M. HOIT AND J.H. GARCELON (1989), Updated profile front minimization algorithm, *Computers & Structures*, **33**(3), 903-914.

[Holmes,Snyder-1988] D.G. HOLMES AND D.D. SNYDER (1988), The generation of unstructured triangular meshes using Delaunay triangulation, *Numerical grid generation in computational fluid mechanics'88*, Miami, 643-652.

[Hoppe *et al.* 1991] H. HOPPE, T. DEROSE, T. DUCHAMP, J. MCDONALD AND W. STUETZLE (1991), Surface reconstruction from unorganized points, *technical report 91-12-03*, Dept. of Comp. Science, University of Washington.

[Hoppe-1994] H. HOPPE (1994), Surface reconstruction from unorganized points, *PhD thesis*, Dept. of Computer Science and Engineering, University of Washington.

[Hosaka-1992] M. HOSAKA (1992), *Modeling of Curves and Surfaces in CAD/CAM*, Springer-Verlag.

[Hughes-1988] T.J.R. HUGHES (1988), *The Finite Element Method: linear static and dynamic finite element analysis*, Prentice-Hall Inc, NJ.

[Ibaroudene,Acharya-1991] D. IBAROUDENE AND R. ACHARYA (1991), Coordinate relationships between vertices of linear octree nodes and corners of the universe, *Comput. and Graphics*, **15**(3), 375-381, 1991.

[Jacquotte-1992] O.P. JACQUOTTE AND G. COUSSEMENT (1992), Structured mesh adaptation: space accuracy and interpolation methods, *Comput. Methods Appl. Mech. Engrg.*, **101**, 397-432.

[Jameson *et al.* 1986] A. JAMESON, T.J. BAKER AND N.P. WEATHERILL (1986), Calculation of inviscid transonic flow over a complete aircraft, *Proc AIAA 24th Aerospace meeting*, Reno.

[Jin,Tanner-1993] H. Jin and R.I. Tanner (1993), Generation of unstructured tetrahedral meshes by advancing-front technique, *Int. j. numer. methods eng.*, **36**, 1805-1823.

[Joe-1989] B. JOE (1989), Three-dimensionnal triangulations from local transformations, *SIAM J. Sci. Stat. Comput.*, **10**(4), 718-741.

[Joe-1991] B. JOE (1991), Construction of three-dimensionnal Delaunay triangulations using local transformations, *Comput. Aided Geom. Design*, **8**, 123-142.

[Joe-1991a] B. JOE (1991), Delaunay versus max-min solid angle triangulations for three-dimensionnal mesh generation, *Int. j. numer. methods eng.*, **31**(5), 987-997.

[Joe-1995] B. JOE (1995), Quadrilateral mesh generation in polygonal regions, *Computer Aided Design*, **27**(3), 209-222.

[Johnson-1995] A. A. JOHNSON (1995), Mesh Generation and Update Strategies for Parallel Computation of Flow Problems with Moving Boundaries and Interfaces, Thesis lecture, U. of Minnesota at Minneapolis.

[Johnson,Hansbo-1992] C. JOHNSON AND P. HANSBO (1992), Adaptive finite element methods in computational mechanics, *Comput. Methods Appl. Mech. Engrg.*, **101**, 143-181.

[Johnston *at al.* 1991] B.P. JOHNSTON, J.M. SULLIVAN AND A. KWASNIK (1991), Automatic conversion of triangular finite element meshes to quadrilateral elements, *Int. j. numer. methods eng.*, **31**, 67-84.

[Johnston,Sullivan-1993] B.P. JOHNSTON AND J.M. SULLIVAN (1993), A normal offsetting technique for automatic mesh generation in three dimensions, *Int. j. numer. methods eng.*, **36**, 1717-1734.

[Jones,Plassmann-1997] M.T. JONES AND P.E. PLASSMANN (1997), Adaptive refinement of unstructured finite-element meshes, *Finite element in Analysis and Design*, **25**(1-2) 41-60.

[Jung,Lee-1991] Y.H. JUNG AND K. LEE(1991), Tetrahedron-based octree encoding for automatic mesh generation, *Technical Report*, Seoul National Univ.

[Kahan-1996] W. KAHAN (1996), Lecture notes on the status of IEEE standard 754 for binary floating-point arithmetic, University of Berkeley, 1996.

[Kallinderis *et al.* 1995] Y. KALLINDERIS, A. KHAWAJA AND H. MCMORRIS (1995), Hybri prismatic/tetrahedral grid generation for complex geometries, *AIAA paper* 95-0211.

[Kalvin,Taylor-1996] A.D. KALVIN AND R.H. TAYLOR (1996), Superfaces: polygonal mesh simplification with bounded error, *IEEE Comp. Graphics and App.*, 64-77.

[Karron-1992] D. KARRON(1992), The "SpiderWeb" surface construction algorithm for building triangle mesh surfaces in noisy volume data, *Proc. 14th Annual Int. Conf. of the IEEE Engineering in Medicine and Biology Soc.*, Paris.

[Kaufmann,Nielson-1993] A.E. KAUFMANN AND G.M. NIELSON (1993), Tutorial on volume visualization techniques and applications, *Computer Graphics Int. 93*, École Polytechnique fédérale de Lausanne.

[Kela *et al.* 1986] A. KELA, R. PERUCCHIO AND H. VOELCKER (1986), Toward automatic finite element analysis, *Comp. Mech. Eng.*, 57-71.

[Kela-1989] A. KELA (1989), Hierarchical octree approximations for boundary representation-based geometric models, *Comp. Aided Des.*, **21**, 355-362.

[Kettner-1998] L. KETTNER (1998), Designing a data structure for polyhedral surfaces. *ACM Symposium on Computational Geometry*, Minneapolis.

[Khawaja *et al.* 1995] A. KHAWAJA, H. MCMORRIS AND Y. KALLINDERIS (1995), Hybrid grids for viscous flows around complex 3D geometries including multiple bodies, *Proc. 12th AIAA CFD Conf.*, AIAA paper-1685.

[Klee-1966] V. KLEE (1966), Convexe polytopes and linear programming, *Proc. IBM Sci. Comput. Symp: Combinatorial Problems*, 123-158.

[Klee-1980] V. KLEE (1980), On the complexity of d-dimensional Voronoï diagrams, *Archiv der Mathematik* **34**, 75-80.

[Klein-1989] R. KLEIN (1989), Concrete and Abstract Voronoï diagrams, *Lecture Notes in Computer Science*, **400**, Springer Verlag.

[Klein *et al.* 1993] R. KLEIN, K. MEHLHORN AND S. MEISER (1993), Randomized incremental construction of abstract Voronoi diagrams, *Comput. Geom. Theory Appl.*, **3**(3), 157-184.

[Knupp,Steinberg-1993] P. KNUPPE AND S. STEINBERG (1993), *The fundamentals of grid generation*, CRC press.

[Knuth-1975] D.E. KNUTH (1975), The Art of Computer Programming, 2nd ed., *Addison-Wesley*, Reading, Mass.

[Knuth-1998a] D.E. KNUTH (1998), The Art of Computer Programming, Vol I: Fundamental algorithms, *Addison-Wesley*, Reading, Mass.

[Knuth-1998b] D.E. KNUTH (1998), The Art of Computer Programming, Vol III: Sorting and Searching, *Addison-Wesley*, Reading, Mass.

[Kohli,Carey-1993] H.S. KOHLI AND G.F. GAREY (1993), Shape optimization using adaptive shape refinement, *Int. j. numer. methods eng.*, **36**, 2435-2451.

[Krause,Rank-1996] R. KRAUSE AND E. RANK (1996), A Fast Algorithm for Point-Location in a Finite Element Mesh, *Computing*, **57**, 49-62.

[Kreiner,Kröplin-1994] R. KREINER AND B. KRÖPLIN (1994), Unstructured quadrilateral mesh generation on surfaces from CAD, *Proc. Fourth Int. Conf. Numerical Grid Generation*, Swansea, UK.

[Kriegman,Ponce-1990] D.J. KRIEGMAN AND J. PONCE (1990), On recognizing and positioning curved 3D objects from image contours, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **12**, 1127-1137.

[Krysl-1996] P. KRYSL (1996), Computational Complexity of the Advancing Front Triangulation, *Engineering with Computers*, **12**, 16-22.

[Kwok *et al.* 1995] W. KWOK, K. HAGHIGHI AND E. KANG, An efficient data structure for the advancing-front triangular mesh generation technique, *Comm. numer. methods eng.*, **11**, 465-473.

[Ladevèze *et al.* 1991] P. LADEVÈZE, J.P. PELLE AND P. ROUGEOT (1991), Error estimates and mesh optimization for finite element computation, *Eng. comput.*, **8**, 69-80.

[Lai-1998] Y.C. LAI (1998), A three-step renumbering procedure for high-order finite element analysis, *Int. j. numer. methods eng.*, **41**, 127-135.

[Lantéri,Loriot-1996] S. LANTERI ET M. LORIOT, Large-scale solutions of Three-dimensional compressible flows using the parallel N3S-MUSCL solver, *Concurrency: Pract. Exp.*, **8**(10), 769-798.

[Laug, Borouchaki-1994] P. LAUG AND H. BOROUCHAKI (1996), The BL2D Mesh Generator, Beginner's Guide, User's and Programmer's Manual, *RT INRIA* **0194**.

[Laug *et al.* 1996] P. LAUG, H. BOROUCHAKI ET P.L. GEORGE (1996), Maillage de courbes gouverné par une carte de métriques, *RR INRIA* **2818**.

[Laug, Borouchaki-1999] P. LAUG AND H. BOROUCHAKI (1999), BLSURF, Mailleur de surfaces composées de carreaux paramétrés. Manuel d'utilisation, *RT INRIA* **0232**.

[Lawson-1972] C.L. LAWSON (1972), Generation of a triangular grid with application to contour plotting, California institute of Technology, JPL, 299.

[Lawson-1972] C.L. LAWSON (1972), Transforming triangulations, *Discrete Mathematics*, **3**, 365-372.

[Lawson-1977] C.L. LAWSON (1977), Software for C^1 surface interpolation, Math. Soft., 3, J. Rice ed., Academic Press, New York.

[Lawson-1986] C.L. LAWSON (1986), Properties of n-dimensional triangulations, *Computer Aided Geometric Design*, **3** , 231-246.

[Lee-1980] D.T. LEE (1980), Two-dimensional Voronoï diagrams in Lp-Metric, *J. of the ACM*, **27**(4), 604-618.

[Lee,Schachter-1980] D.T. LEE AND B.J. SCHACHTER (1980), Two algorithms for constructing a Delaunay Triangulation, *Int. J. Comp. Inf. Sci.*, **9**(3), 219-242.

[Lee,Lo-1994] C.K. LEE AND S.H. LO (1994), A new scheme for the generation of a graded quadrilateral mesh, *Comp. Struct.*, **52**, 847-857.

[Lee,Wong-1980] D.T. LEE AND C.K. WONG (1980), Voronoi diagrams in L_1 (L_∞) metrics with 2-dimensional storage applications, *SIAM J. Comput.*, **9**, 200-211.

[Lee,Lin-1988] D.T. LEE AND A.K. LIN (1988), Generalized Delaunay triangulation for planar graphs, *Discrete Comput. Geom.*, **1**, 201-217.

[Lelong-Ferrand,Arnaudies-1977] J. LELONG-FERRAND AND J.M. ARNAUDIÈS (1977), *Cours de Mathématiques*, Tome 3, *Géométrie et cinématique*, Dunod Université, Bordas.

[Lennes-1911] N.J. LENNES (1911), Theorems on the simple finite polygon and polyhedron, *Am. J. Math.*, **33**, 37-62.

[Léon-1991] J.C. LÉON (1991), *Modélisation et construction des surfaces pour la CFAO*, Éditions Hermès, Paris.

[Lewis-1982] J.G. LEWIS (1982), Implementation of the Gibss-Poole-Stockmeyer and Gibbs-King algorithms, *ACM Trans. Math. Software*, **8**(2), 180-189.

[Lewis *et al.* 1996] R.W. LEWIS, Y. ZHENG AND D.T. GETHIN (1996), Three-dimensional unstructured mesh generation: Part 3. Volume meshes, *Comput. Methods Appl. Mech. Engrg.*, **134**, 285-310.

[Lewis *et al.* 1995] R.W. LEWIS, Y. ZHENG AND A.S. USMANI (1995), Aspects of adaptive mesh generation based on domain decomposition and Delaunay triangulation., *Finite Elements in Anal. and Design.*, **20**, 47-70.

[Li *et al.* 1997] T.S. LI, C.G. ARMSTRONG AND R.M. MCKEAG (1997), Quad mesh generation for k-sided faces and hex mesh generation for trivalent polyhedra, *Finite Elements in Analysis and Design*, **26**(4), 279-301.

[Liu,Joe-1994] A. LIU AND B. JOE (1994), On the shape of tetrahedra from bisection, *Mathematics of Computation*, **63**(207), 141-154.

[Lo-1985] S.H. LO (1985), A new mesh generation scheme for arbitrary planar domains, *Int. j. numer. methods eng.*, **21**, 1403-1426.

[Lo-1989] S.H. LO (1989), Delaunay triangulation of non-convex planar domains, *Int. j. numer. methods eng.*, **28**, 2695-2707.

[Lo-1989] S.H. LO (1989), Generating quadrilateral elements on plane and over curved surfaces, *Comp. Struct.*, **31**, 421-426.

[Löhner-1988] R. LÖHNER (1988), Some useful Data Structures for the Generation of unstructured grids, *Commun. numer. methods eng.*, **4**, 123-135.

[Löhner,Parikh-1988] R. LÖHNER AND P. PARIKH (1988), Three-Dimensional Grid Generation by the Advancing Front Method, *Int. j. numer. methods fluids*, **8**, 1135-1149.

[Löhner-1989] R. LÖHNER (1989), Adaptive remeshing for transient problems, *Comput. Methods Appl. Mech. Engrg.*, **75**, 195-214.

[Löhner et al.1992] R. LÖHNER, J. CAMBEROS AND M. MERRIAM (1992), Parallel Unstructured Grid Generation, *Comput. Methods Appl. Mech. Engrg.*, **95**, 343-357.

[Löhner-1993] R. LÖHNER (1993), Matching semi-structured and unstructured grids for Navier-Stokes calculations, *AIAA paper 93-3348*, july.

[Löhner *et al.* 1992] R. LÖHNER, J. CAMBEROS AND M. MERRIAM (1992), Parallel Unstructured Grid Generation, *Comput. Methods Appl. Mech. Engrg.*, **95**, 343-357.

[Löhner,Ramamurti-1993] R. LÖHNER AND R. RAMAMURTI (1993), A parallelizable load balancing algorithm, *Proc. of the AIAA 31st Aerospace Sciences Meeting and Exhibit.*

[Löhner-1995] R. LÖHNER (1995), Surface gridding from discrete data, *Proc. 4th International Meshing Roundtable*, Albuquerque, NM, 29-45.

[Löhner-1996a] R. LÖHNER (1996), Regridding Surface Triangulations, *Jour. of Comput. Phys.*, **126**, 1-10.

[Löhner-1996b] R. LÖHNER (1996), Extensions and improvements of the advancing-front grid generation technique, *Commun. numer. methods eng.*, **12**, 683-702.

[Löhner-1996c] R. LÖHNER (1996), Progress in grid generation via the advancing-front technique, *Engineering with Computers*, **12**, 186-210.

[Löhner-1997] R. LÖHNER (1997), Automatic Unstructured Grid Generators, *Finite Elements in Analysis and Design*, **25**(3-4), 111-134.

[Löhner-1998] R. LÖHNER (1998), Renumbering strategies for unstructured-grid solvers operating on shared memory, cache-based parallel machines, *Comput. Methods Appl. Mech. Engrg.*, **163**(1-4), 95-110.

[Lorensen,Cline-1987] W.E. LORENSEN AND H.E. CLINE (1987), Marching cubes: a high resolution 3D surface construction algorithm, *Comput. Graphics*, **21**(4), 163-169.

[Mackerle-1993] J. MACKERLE (1993), Mesh generation and refinement for FEM and BEM - a bibliography, *Finite Elem. Anal. Design*, 177-188.

[McMorris,Kallinderis-1997] H. MACMORRIS AND Y. KALLINDERIS (1997), Octree-advancing front method for generation of unstructured surface and volume meshes, *AIAA journal*, **35**(6), 976-984.

[Mahmoud-1992] H. MAHMOUD (1992), *Evolution of random search trees*, Wiley, New York.

[Mäntylä-1988] M. MÄNTYLÄ (1988), An Introduction to Solid Modeling. Rockville, Md.: Computer Science Press.

[Marchant et al.1997] M.J. MARCHANT, N.P. WEATHERILL AND O. HASSAN (1997), The adaptation of unstructured grids for transonic viscous flow simulation, *Finite Elements in Analysis and Design*, **25**(3-4), 199-217.

[Marcum-1996] D.L. MARCUM (1996), Unstructured grid generation components for complete systems, *5th Int. Conf. on Grid Generation in Comp. Field Simulations*, MS, USA, 1-5 April.

[Marcum,Weatherill-1995] D.L. MARCUM AND N.P. WEATHERILL (1995), Unstructured grid generation using iterative point insertion and local reconnection, *AIAA Journal.*, **33**(9), 1619-1625.

[Matveyev-1994] S.V. MATVEYEV (1994), Approximation of isosurface in the Marching Cube: ambiguity problem, *Visualization'94*, Conf. Proc., IEEE Computer Society Press, 288-292.

[Mavriplis-1990] D.J. MAVRIPLIS (1990), Adaptive mesh generation for viscous flows using Delaunay triangulation, *J. Comput. Physics*, **90**(2), 271-291.

[Mavriplis-1992] D.J. MAVRIPLIS (1992), An advancing front Delaunay triangulation algorithm designed for robustness, *ICASE report 92-49.*

[Mavriplis-1995] D.J. MAVRIPLIS (1995), Unstructured mesh generation and adaptivity, *ICASE report 95-26*

[Meagher-1982] D. MEAGHER (1982), Geometric modeling using octree encoding,*Comput. Graphics and Image Proc.*, **19**, 129-147.

[Mehlhorn *et al.* 1991] D. MEHLHORN, S. MEISER AND C. O'DUNLAING (1991), On the construction of abstract Voronoi diagrams, *Discrete Comput. Geom.*, **6**, 211-224.

[Melhem-1987] R. MELHEM (1987), Towards efficient implementation of preconditioned conjugate gradient methods on vector supercomputers, *Int. Jour. of Supercomp. Appl.*, **1**.

[Merriam-1991] M.L. MERRIAM (1991), An efficient advancing front algorithm for Delaunay triangulation., *AIAA 91-0792.*

[Miller *et al.* 1991] J.V. MILLER, D.E. BREN, W.E. LORENSEN, R.B. O'BARA AND M.J. WOZNY (1991), Geometrically deformed models: a method for extracting closed geometric models from volume data, *Computer Graphics*, **25**(4), 217-225.

[Miller *et al.* 1997] G.L. MILLER, D. TALMOR AND S.H. TENG (1997), Optimal coarsening of unstructured meshes, *Proc. 8th ACM-SIAM Symp. Disc. Algorithms.*

[Mitchell,Vavasis-1992] S. MITCHELL AND S. VAVASIS (1992), Quality Mesh Generation in Higher Dimensions, *Proc. 8th ACM Symp. Comp. Geometry*, 212-221.

[Mitty *et al.* 1993] T.J. MITTY, A. JAMESON AND T. J. BAKER (1993), Solution of three-dimensionnal supersonic flowfields via adapting unstructured meshes, *Computer Fluids*, **22**(2/3), 271-283.

[Mohammadi-1994] B. MOHAMMADI (1994), Fluid dynamics computation with NSC2KE - an User-Guide, Release 1.0, *RT INRIA* **164**.

[Mohammadi,Pironneau-1994] B. MOHAMMADI AND O. PIRONNEAU (1994), *Analysis of the K-Epsilon Turbulence Model*, Wiley and Masson (Eds.).

[Möller,Hansbo-1995] P. MÖLLER AND P. HANSBO (1995), On advancing-front mesh generation in three dimensions, *Int. j. numer. methods eng.*, **38**, 3551-3569.

[Monga,Benayoun-1995] O. MONGA AND S. BENAYOUN (1995), Using partial derivatives of 3D images to extract typical surface features, *Comput. Vision and Image Understanding*, **61**(2), 171-189.

[Montani *et al.* 1994] C. MONTANI, R. SCATENI AND R. SCOPIGNO (1994), A modified look-up table for implicit disambiguation of Marching Cubes, *The Visual Computer*, **10**.

[Moreton-1992] H.P. MORETON (1992), Minimum curvature variation curves, networks and surfaces for fair free-form shape design, *PhD thesis*, Dept. of Computer science, Univeristy of California, Berkeley.

[Mortenson-1985] M.E. MORTENSON (1985), *Geometric modeling*, Wiley.

[Muller *et al.* 1992] J.D. MULLER, P.L. ROE AND H. DECONINCK (1992), A frontal approach for node generation in Delaunay triangulations., *Unstructured grid methods for advection dominated flows.*, VKI Lecture notes, 91-97, AGARD Publication R-787.

[Muller,Stark-1993] H. MULLER AND M. STARK (1993), Adaptive generation of surfaces in volume data, *The Visual Computer*, **9**(4), 182-199.

[Mulmuley-1993] K. MULMULEY (1993), *Computational Geometry: An Introduction Through Randomized Algorithms*, Prentice Hall, New York.

[Murdoch,Benzley-1995] P. MURDOCH AND S. BENZLEY (1995), The Spatial Twist Continuum, *Proc. 4th International Meshing Roundtable*, 243-251, Albuquerque, NM.

[Nakahashi,Sharov-1995] K. NAKAHASHI AND D. SHAROV (1995), Direct surface triangulation using the advancing-front method, *AIAA-95-1686-CP*, 442-451.

[Natarajan-1997] R. NATARAJAN (1997), Domain Decomposition using Spectral Expansions of Steklov-Poincaré Operators II: A Matrix Formulation, *SIAM Journal on Scientific Computing*, **18**(4), 1187-1199.

[Nielson,Hamann-1991] G.M. NIELSON AND B. HAMANN (1991), The asymptotic decider: resolving the ambiguity in Marching Cubes, *Visualization'91*, Conf. Proc., IEEE Computer Society Press, 83-90.

[Ning,Bloomenthal-1993] P. NING AND J. BLOOMENTHAL (1993), An evaluation of implicit surface tilers, *IEEE Computer Graphics & Applications*, **13**(6), 3341.

[Oden *et al.* 1995] J.T. ODEN, W. WU AND M. AINSWORTH (1995), Three-step hp adaptive strategy for the incompressible Navier-Stockes equations, *IMA Volumes in Mathematics and its Applications*, I. Babuska, W.D. Henshaw, J.E. Oliger, J.E. Flaherty, J.E. Hopcroft and T. Tezduyar (Eds.), **75**, 347-366.

[Okusanya,Peraire-1996] T. OKUSANYA AND J. PERAIRE (1996), Parallel unstructured mesh generation, *Proc. 5th Int. Conf. Numerical Grid Generation in Computational Field Simulations*, Mississippi State Univ., 719-729.

[deOliveira *et al.* 1997] M. DE OLIVEIRA, A.C. SALGADO, R.A. FEIJÓO, M.J. VÉNERE AND E.A. DARI, An object oriented Tool for automatic surface mesh generation using the Advancing Front technique, *Latin American Applied Research*, **27**(1/2), 39-49.

[O'Rourke-1994] J. O'ROURKE (1994), *Computational geometry in C*, Cambridge University Press.

[Owen *et al.* 1998] S.J. OWEN, M.L. STATEN, S.A. CANANN AND S. SAIGAL (1998), Q-Morph : an indirect approach to advancing-front quad meshing, *Int. j. numer. meth. eng.*, **44**(9), 1317-1340.

[Owen-1999a] S.J. OWEN (1999), Non-Simplicial Unstructured Mesh Generation, *PhD thesis*, Carnegie Mellon Univ., Pittsburg, PA.

[Owen-1999b] S.J. OWEN (1999), Constrained triangulation : application to hex-dominant mesh generation, *Proc. 8th Meshing Roundtable*, Lake Tahoe, CA.

[Owen,Saigal-2000] S.J. OWEN AND S. SAIGAL (2000), Surface mesh sizing control, *Int. j. numer. meth. eng.*, **46**, 497-511.

[Palmerio-1994] B. PALMERIO (1994), An attraction-repulsion mesh adaption model for flow solution on unstructured grid, *Comp. and Fluids*, **23**(3), 487-506.

[Parthasarathy *et al.* 1993] V.N. PARTHASARATHY, C.M. GRAICHEN AND A.F. HATHAWAY (1993), A comparison of tetrahedron quality measures, *Finite Elements in Analysis and Design*, **15**, 255-261.

[Pasko *et al.* 1995] A. PASKO, V. ADZHIEV, A. SOURIN AND V. SAVCHENKO, Function representation in geometric modeling: concepts, implementation and applications, *The Visual Computer*, **11**, 429-446.

[Pebay-1998a] P. PEBAY (1998), Construction d'une triangulation surfacique Delaunay admissible, *RR INRIA* **3369**.

[Pebay-1998b] P. PEBAY (1998), Construction d'une contrainte Delaunay admissible en dimension 2, *RR INRIA* **3492**.

[Peraire *et al.* 1987] J. PERAIRE, M. VAHDATI, K. MORGAN AND O.C. ZIENKIEWICZ (1987), Adaptive remeshing for compressible flow computations, *Jour. of Comput. Phys.*, **72**, 449-466.

[Peraire *et al.* 1988] J. PERAIRE, J. PEIRO, L. FORMAGGIA, K. MORGAN, O.C. ZIENKIEWICZ (1988), Finite element Euler computations in three dimensions, *Int. j. numer. methods eng.*, **26**, 2135-2159.

[Peraire *et al.* 1992] J. PERAIRE, J. PEIRO, K. MORGAN (1992), Adaptive remeshing for three-dimensional compressible flow computations, *Jour. of Comput. Phys.*, **103**, 269-285.

[Peraire,Morgan-1997] J. PERAIRE AND K. MORGAN (1997), Unstructured mesh generation including directional refinement for aerodynamic flow simulation, *Finite Elements in Analysis and Design*, **25**(3-4), 343-355.

[Perronnet-1984] A. PERRONNET (1984), Logical and physical representation of an object, modularity for the programming of finite element methods, *PDE Software, Interfaces and Systems*, Elsevier, IFIP.

[Perronnet-1988a] A. PERRONNET (1988), Tétraèdrisation d'un objet multimatériaux ou de l'extérieur d'un objet, Laboratoire d'analyse numérique 189, Université Paris 6.

[Perronnet-1988b] A. PERRONNET (1988), A generator of tetrahedral finite elements for multi-material object and fluids, Numerical grid generation in computational fluid mechanics'88, Miami, FL.

[Perronnet-1992] A. PERRONNET (1992), Triangulation par arbre-4 de triangles équilatéraux et maximisation de la qualité, *Rapport de Recherche*, R-92015, Univ. Paris 6.

[Perronnet-1993] A. PERRONNET (1993), Tetrahedrization by the 5-14-OTT-tree technique and the Delaunay's criterion, *Proc. 8th Conf. on Finite Elements in Fluids*, Barcelona, Spain.

[Perronnet-1998] A. PERRONNET (1998), Interpolation transfinie sur le triangle, le tetraèdre et le pentaèdre. Application à la création de maillages et à la condition de Dirichlet. *C.R. Acad. Sci. Paris*, t 326, Serie I, 117-122.

[Perucchio *et al.* 1989] R. PERUCCHIO, M. SAXENA AND A. KELA (1989), Automatic mesh generation from solid models based on recursive spatial decompositions, *Int. j. numer. meth. eng.*, **28**, 2469-2501.

[Piegl,Richard-1995] L.A. PIEGL AND A.M. RICHARD (1995), Tesselations of trimmed NURBS surfaces, *Comput. Aided Des.*, **7**(1), 12-26.

[Piper-1987] B.R. PIPER (1987), Visually smooth interpolation with triangular Bézier patches, in *Geometric modeling: algorithms and new trends*, G. Farin Ed., SIAM, 221-233.

[Pirzadeh-1994] S. PIRZADEH (1994), Viscous unstructured three-dimensional grids by the advancing-layers method, *AIAA-94-0417*.

[Pothen *et al.* 1990] A. POTHEN, H. D. SIMON AND K.-P. LIOU (1990), Partitioning Sparse Matrices with eigenvectors of graphs, *SIAM Journal Matrix Analysis Application*, **11**(3), 430-252.

[Preparata,Hong-1977] F.P. PREPARATA AND S.J. HONG (1977), Convex hull of finite sets of points in two and three dimension, *Com. of the ACM*, **2**(20), 87-93.

[Price *et al.* 1995] M.A. PRICE, C.G. ARMSTRONG AND M.A. SABIN (1995), Hexahedral mesh generation by medial surface subdivision; Part I. Solids with convex edges, *Int. j. numer. methods eng.*, **38**, 3335-3359.

[Price,Armstrong-1997] M.A. PRICE AND C.G. ARMSTRONG (1997), Hexahedral mesh generation by medial surface subdivision; Part II. Solids with flat and concave edges, *Int. j. numer. methods eng.*, **40**, 111-136.

[Rajan-1994] V.T. RAJAN (1994), Optimality of the Delaunay Triangulation in $\mathbb{R}^d$, *Discrete Comput. Geom.*, **12** , 189-202.

[Raviart,Thomas-1983] P.A. RAVIART ET J.M. THOMAS (1983), *Introduction à l'analyse numérique des équations aux dérivées partielles*, Masson, Paris.

[Ravichandran,Shephard-1995] R. RAVICHANDRAN AND M.S. SHEPHARD (1995), Mesh searching structures for adaptive finite element analysis, *Scorec Report* **26**, RPI, Troy, NY.

[Rank *et al.* 1993] E. RANK, M. SCHWEINGRUBER AND M. SOMMER (1993), Adaptive mesh generation and transformation of triangular to quadrilateral meshes, *Commun. numer. methods eng.*, **9**(2), 121-129.

[Rassineux-1995] A. RASSINEUX (1995), Maillage automatique tridimensionnel par une méthode frontale pour la méthode des éléments finis, Thèse, Nancy I.

[Rassineux-1997] A. RASSINEUX (1997), Maillage automatique tridimensionnel par une technique frontale et respect d'une carte de taille, *Revue européenne des éléments finis* **6**(1), 43-70.

[Rebay-1993] S. REBAY (1993), Efficient unstructured mesh generation by means of Delaunay triangulation and Bowyer/Watson algorithm, *J. Comput. Physics*, **106**, 125-138.

[Reddy,Turkiyyah-1995] J.M. REDDY AND G.M. TURKIYYAH (1995), Computatin of 3D skeletons using a generalized Delaunay triangulation technique, *Computer Aided Design*, **27**(9), 677-694.

[vanRens *et al.* 1998] B.J.E. VANRENS, D. BROKKEN, W.A.M. BREKELMANS AND F.P.T. BAAIJENS (1998), A two-dimensional paving mesh generator for triangles with controllable aspect ratio and quadrilaterals with high-quality, *Engineering with Computers*, **14**, 248-259.

[Requicha-1980] A.A.G. REQUICHA (1980), Representations for rigid solids: theory, methods and systems, *Comput. Surveys*, **12**, 437-464.

[Ricci-1972] A. RICCI (1972), A constructive geometry for computer graphics, *The Computer Journal*, **16**, 157-160.

[Rippa-1990] S. RIPPA (1990), Minimal roughness property of the Delaunay triangulation, *Comput. Aided Geom. Design.*, **7**, 489-497.

[Rippa-1992] S. RIPPA (1992), Long and thin triangles can be good for linear interpolation, *SIAM J. Numer. Analysis*, **29**, 257-270.

[Risler-1991] J.J. RISLER (1991), *Méthodes mathématiques pour la C.A.O.*, RMA 18, Masson, Paris.

[Rivara-1984a] M.C. RIVARA (1984), Mesh refinement processes based on the generalised bisection of simplices, *SIAM J of Num. Ana.*, **21**, 604-613.

[Rivara-1984b] M.C. RIVARA (1984), Algorithms for refining triangular grids suitable for adaptive and multigrid techniques, *Int. j. numer. methods eng.*, **21**, 745-756.

[Rivara-1986] M.C. RIVARA (1986), Adaptive finite element refinement and fully irregular and conforming triangulations, in *Accuracy Estimates and Adaptive Refinements in Finite Element Computations*, I. Babuska *et al.*, John Wiley and Sons.

[Rivara-1990] M.C. RIVARA (1990), Selective refinement/derefinement algorithms for sequences of nested triangulations, *Int. j. numer. methods eng.*, **28**(12), 2889-2906.

[Rivara-1991] M.C. RIVARA (1991), Local modification of meshes for adaptive and/or multigrid finite element methods, *J. Comp. and Appl. Math.*, **36**, 79-89.

[Rivara,Levin-1992] M.C. RIVARA AND C. LEVIN (1992), A 3D refinement algorithm suitable for adaptive and multigrid techniques, *J. Comp. and Appl. Math.*, **8**, 281-290.

[Rivara-1997] M.C. RIVARA (1997), New longest-edge algorithms for the refinement and/or improvement of unstructured triangulations, *Int. j. numer. methods eng.*, **40**, 3313-3324.

[Robinson-1987] J. ROBINSON (1987), Some new distorsion measures for quadrilaterals, *Finite Elements in Analysis and Design*, **3**, 183-197.

[Robinson-1988] J. ROBINSON (1988), Distorsion measures for quadrilaterals with curved boundaries, *Finite Elements in Analysis and Design*, **4**, 115-131.

[Rogers,Adams-1989] D.F. ROGERS AND J.A. ADAMS (1989), *Mathematical Elements for Computer Graphics*, McGraw-Hill.

[Ruppert,Seidel-1992] J. RUPPERT AND R. SEIDEL (1992), On the difficulty of triangulating three-dimensional nonconvex polyhedra, *Discrete Comput. Geom.*, **7**, 227-253.

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

[Rypl,Krysl-1994] D. RYPL AND P. KRYSL (1994), Triangulation of 3-D surfaces, *Tech. Report*, Czech Tech. Univ., Prague.

[Rypl,Krysl-1997] D. RYPL AND P. KRYSL (1997), Triangulation of 3-D surfaces, *Engineering with Computers*, **13**, 87-98.

[Rypl-1998] D. RYPL (1998), Sequential and parallel generation of unstructured 3D meshes, *PhD thesis*, Faculty of Civil Engineering, Czech Tech. Univ., Prague.

[Rvachev-1963] V.L. RVACHEV (1963), On the analytical description of some geometric objects, *Report of the Ukrainian Academy of Sciences*, **153**, 765-767.

[Salmon,1885] G. SALMON (1885), *Modern Higher Algebra*, Longmans, Greens and Co., London.

[Samareh-Abolhassani,Stewart-1994] J. SAMAREH-ABOLHASSANI AND J.E. STEWART (1994), Surface grid generation in parametric space, *J. Comput. Physics*, **113**, 112-121.

[Samet-1984] H. SAMET (1984), The Quadtree and Related Hierarchical Data Structures, *ACM Computing Surveys*, **16**(2), 187-260.

[Samet-1989] H. SAMET (1989), Neighbor finding in images represented by octrees, *CVGIP*, **46**, 367-386.

[Samet-1990] H. SAMET (1990),*Design and analysis of spatial data structures*, Addison Wesley.

[Sapidis,Perucchio-1991] N. SAPIDIS AND R. PERUCCHIO (1991), Delaunay triangulation of arbitrarily shaped planar domains, *Comput. Aided Geom. Des.*, **8**, 421-437.

[Sapidis,Perucchio-1993] N. SAPIDIS AND R. PERUCCHIO (1993), Combining recursive spatial decompositions and domain Delaunay tetrahedrizations for meshing arbitrarily shaped curved solid models, *Comput. Methods Appl. Mech. Engrg.*, **108**, 281-302.

[Savchenko *et al.* 1995] V.V. SAVCHENKO, A.A. PASKO, O.G. OKUNEV AND T.L. KUNII (1995), Function representation of solids reconstructed from scattered surface points and contours, *Computer Graphics Forum*, **14**(4), 181-188.

[Saxena,Perucchio-1992] M. SAXENA AND R. PERUCCHIO (1992), Parallel FEM algorithms based on recursive spatial decomposition. I: automatic mesh generation, *Computers & Structures*, **45**(5-6), 817-831.

[Saxena *et al.* 1995] M. SAXENA, P.M. FINNIGAN, C.M. GRAICHEN, A.F. HATHAWAY AND V.N. PARTHASARATHY (1995), Octree-based automatic mesh generation for non-manifold domains, *Engineering with Computers*, **11**, 11-14.

[Schaeffer-1979] H. G. SCHAEFFER(1979), MSC/NASTRAN Primer. Static and Normal Modes Analysis, *Schaeffer Analysis*, Mont Vernon.

[Schmidt-1993] M.F.W. SCHMIDT (1993), Cutting cubes - visualizing implicit surfaces by adaptive polygonization, *The Visual Computer*, **10**, 101-115.

[Schneiders *et al.* 1996] R. SCHNEIDERS, R. SCHINDLER AND F. WEILER (1996), Octree-based generation of hexahedral element meshes, in *Proc. 5th International Meshing Roundtable*, Pittsburgh, PA, 205-215.

[Schneiders-1996a] R. SCHNEIDERS (1996), A grid-based algorithm for generation of hexahedral element meshes, *Engineering with Computers*, **12**, 168-177.

[Schneiders-1996b] R. SCHNEIDERS (1996), Refining quadrilateral and hexahedral element meshes, *Proc. 5th Int. Conf. on Numerical Grid Generation in Computational Field Simulations*, 679-688.

[Schöberl-1997] J. SCHÖBERL (1997), NETGEN: An advancing-front 2D/3D-mesh generator based on abstract rule, *Comput Visual Sci.*, **1** 41-52.

[Schönhardt-1928] E. SCHÖNHARDT (1928), Über die Zerlegung von Dreieckspolyedern, *Mathematish Annalen*, **98**, 309-312.

[Schroeder,Shephard-1988] W.J. SCHROEDER AND M.S. SHEPHARD (1988), Geometry-based fully automatic mesh generation and the delaunay triangulation *Int. j. numer. meth. eng.*, **26**, 2503-2515.

[Schroeder,Shephard-1989] W.J. SCHROEDER AND M.S. SHEPHARD (1989), An $O(N)$ algorithm to automatically generate geometric triangulations satisfying the Delaunay circumsphere criteria, *Engrg. with Computers*, **5**, 177-193.

[Schroeder,Shephard-1990] W.J. SCHROEDER AND M.S. SHEPHARD (1990), A combined octree/Delaunay method for fully automatic automatic 3D mesh generation, *Int. j. numer. meth. eng.*, **29**, 37-55.

[Schroeder *et al.* 1992] W.J. SCHROEDER, J.A. ZARGE AND W. LORENSEN, Decimation of triangle mesh, *ACM Comput. Graphics*, **26**(2), 65-70.

[Sclaroff,Pentland-1991] S. SCLAROFF AND A. PENTLAND, generalized implicit functions for computer graphics, *Comput. Graph.*,**25**, 247-250.

[Schoofs *et al.* 1979] A.J.G. SCHOOFS, L.H.T. VANBEUKERING AND M.L.C. SLUITER (1979), A general purpose two-dimensional mesh generator, *Adv. Eng. Software*, **1**, 131-136.

[Sederberg-1983] T.W. SEDERBERG (1983), Implicit and parametric curves and surfaces for Computer-Aided Design, *PhD thesis*, Purdue Univ., West Lafayette, Ind.

[Sederberg,Parry-1986] T. W. SEDERBERG AND S. R. PARRY (1986), Comparison of three curve intersection algorithms, *Computer Aided Design*, **18**(1), 58-63.

[Sederberg-1987] T. W. SEDERBERG (1987), Algebraic geometry for surface and solid modeling, in *Geometric Modeling : algorithms and new trends*, G.E. Farin, Ed. SIAM, 29-42.

[Sederberg,Chen-1995] T.W. SEDERBERG AND F. CHEN (1995), Implicitization using moving curves and surfaces, *Computer Graphics Annual Conf. Series*, 301-308.

[Sedgewick-1988] R. SEDGEWICK (1988), *Algorithms*, Addison-Wesley.

[Sedgewick,Flajolet-1996] R. SEDGEWICK AND PH. FLAJOLET (1996), Lecture Notes on Bucket Algorithms, *Birkauser.*

[Seidel-1982] R. SEIDEL (1982), The complexity of Voronoi diagrams in higher dimensions, *Proc. 20th Ann. Allerton Conf. Commun., Control, Comput.*, 94-95.

[Seidel-1991] R. SEIDEL (1991), Small-dimensional linear programming and convex hulls made easy, *Discrete Comput. Geom.*, **6**, 423-434.

[Seveno-1997] E. SEVENO (1997), Towards an adaptive advancing-front mesh generation, *Proc. 6th Int. Meshing Roundtable*, Park City, Utah, Oct. 12-15, 349-360.

[Seveno-1998] E. SEVENO (1998), Génération automatique de maillages tridimensionnels isotropes par une méthdoe frontale, Thèse d'Université, Paris 6.

[Shamos,Preparata-1985] F.P. PREPARATA AND M.I. SHAMOS (1985), *Computational geometry, an introduction*, Springer-Verlag.

[Shapiro-1994] V. SHAPIRO (1994), Real functions for representation of rigid solids, *CAGD*, **11**, 153-175.

[Sheehy *et al.* 1996] D.S. SHEEHY, G.C. ARMSTRONG AND D.J. ROBINSON (1996), Shape description by medial surface construction, *IEEE Trans. on Vizualization and Comput. Graph.*, **2**(1), 62-72.

[Sheng,Hirsch-1992] X. SHENG AND B.E. HIRSCH (1992), Triangulation of trimmed surfaces in parametric space, *Comput. Aided Des.*, **24**(8), 437-444.

[Shenton *et al.* 1985] D.N. SHENTON AND Z.J. CENDES (1985), Three-dimensional finite element mesh generation using Delaunay tesselation, *IEEE Trans. Magnetics*, **21**(6), 2535-2538.

[Shephard-1988] M.S. SHEPHARD (1988), Approaches to the automatic generation and control of finite element meshes, *Applied Mechanics Reviews*, **41**, 169-185.

[Shephard *et al.* 1988] M.S. SHEPHARD, F. GUERINONI, J.E. FLAHERTY, R.A. LUDWIG, P.L. BAEHMANN (1988), Finite octree mesh generation for automated adaptive 3D flow analysis, *Numerical grid generation in computational fluid mechanics '88*, Miami.

[Shephard *et al.* 1988a] M.S. SHEPHARD, F. GUERINONI, J.E. FLAHERTY, R.A. LUDWIG, P.L. BAEHMANN (1988), Adaptive solutions of the Euler equations using finite quadtree and octree grids, *Computers & Structures*, **30**, 327-336.

[Shephard *et al.* 1988b] M.S. SHEPHARD, K.R. GRICE, J.A. LO, W.J. SCHROEDER (1988), Trends in automatic three-dimensional mesh generation, *Comp. and Struct.*, **30**(1/2), 421-429.

[Shephard,Georges-1991] M.S. SHEPHARD AND M.K. GEORGES (1991), Automatic three-dimensional Mesh Generation by the Finite Octree Technique, *Int. j. numer. meth. eng.*, **32**, 709-749.

[Shephard,Georges-1992] M.S. SHEPHARD AND M.K. GEORGES (1992), Reliability of automatic 3D mesh generation, *Computer Methods in Appl. Mechanics and Engineering*, **101**, 443-462.

[Shephard *et al.* 1996] M.S. SHEPHARD, S. DEY AND M.K. GEORGES (1996), Automatic meshing of curved three-dimensional domains: curving finite elements and curvature-based mesh control, *Technical Report*, RPI Scientific Computation Research Center.

[Shephard-2000] M.S. SHEPHARD (2000), Meshing environment for geometry-based analysis, *Int. j. numer. eng.* **47**, 169-190.

[Shewchuk-1997a] J.R. SHEWCHUK (1997), Delaunay Refinement Mesh Generation, *Ph.D. thesis*, Technical Report CMU-CS-97-137, School of Computer Science, Carnegie Mellon University, Pittsburgh, Pennsylvania.

[Shewchuk-1997b] J.R. SHEWCHUK (1997), Adaptive precision floating-point arithmetic and fast robust geometric predicates, *Discrete and Computational Geometry*, **18**.

[Shirman,Séquin-1991] L.A. SHIRMAN AND C.H. SÉQUIN (1991), Procedural construction of patch-boundary-curves, in *Curves and surfaces*, Academic Press, Boston.

[Shmit-1982] R.E. SHMIT (1982), Algebraic grid generation, *Numerical grid generation*, Ed. J.F. Thompson, North Holland, 137.

[Shostko,Löhner-1992] A. SHOSTKO AND R. LÖHNER (1995), Three-Dimensional Parallel Unstructured Grid Generation, *Int. j. numer. methods eng.*, **38**, 905-925.

[Simon-1991] H. SIMON (1991), Partitioning of unstructured problems for parallel processing, *Comp. Systems in Eng.*, **2**, 135-148.

[Simpson-1994] R.B. SIMPSON (1994), Anisotropic mesh transformation and optimal error control, *Applied Num. Math.*, **4**, 183-198.

[Simpson-1998] R.B. SIMPSON (1998), C++ classes for 2d unstructured mesh programming, *RR INRIA* **3592**.

[Sloan-1987] S.W. SLOAN (1987), A fast algorithm for constructing Delaunay triangulations in the plane, *Adv. Eng. Soft.*, **9**(1), 34-55.

[Sloan,Houlsby-1984] S.W. SLOAN AND G.T. HOULSBY (1984), An implementation of Watson's algorithm for computing 2-dimensional Delaunay triangulations, *Adv. Eng. Soft.*, **6**(4), 192-197.

[Sloan-1986] S.W. SLOAN (1986), An algorithm for profile and wavefront reduction of sparse matrices, *Int. j. numer. methods eng.*, **23**, 239-251.

[Smith *et al.* 1996] B. SMITH, P. BJØRSTAD AND W. GROPP (1996), *Domain decomposition*, Cambridge Univ. Press, Cambridge.

[Stander,Hart-1997] B.T. STANDER AND J.C. HART (1997), Guaranteeing the topology of an implicit surface polygonization for interactive modeling, *Computer Graphics Proc.*, 279-286.

[Staten *et al.* 1998] M.L. STATEN, S.A. CANANN AND S.J. OWEN (1998), BMsweep : Locating Interior Nodes During Sweeping, *Proc 7th Int. Meshing Roundtable*, 7-18.

[Steger,Sorenson-1980] J.L. STEGER AND R.L. SORENSON (1980), Use of hyperbolic partial differential equations to generate body-fitted coordinates, *Proc. NASA Langley workshop on Numerical grid generation techniques.*

[Tacher,Parriaux-1996] L. TACHER AND A, PARRIAUX (1996), Automatic nodes generation in N-dimensional space *Commun. numer. methods eng.*, **12**, 243-248.

[Talbert,Parkinson-1991] J.A. TALBERT AND A.R. PARKINSON (1991), Development of an automatic two-dimensional finite element mesh generator using quadrilateral elements and Bezier curve boundary definition, *Int. j. numer. methods eng.*, **29**, 1551-1567.

[Talon-1987a] J.Y. TALON (1987), Algorithmes de génération et d'amélioration de maillages en 2D, Rapport Technique Artemis-Imag **20**.

[Talon-1987b] J.Y. TALON (1987), Algorithmes d'amélioration de maillages tétraèdriques en 3 dimensions, Rapport Technique Artemis-Imag **21**.

[Tam,Armstrong-1991] T.K. TAM AND G.C. ARMSTRONG (1991), 2D finite element mesh generation by medial axis subdivision, *Adv. in Engrg. Soft.*, **13**, 313-324.

[Tamminen,Jansen-1985] M. TAMMINEN AND F.W. JANSEN (1985), An integrity filter for recursive subdivision meshes, *Computers and Graphics*, **9**(4), 351-363.

[Tanaka *et al.* 1990] H. TANAKA, O. KLING AND D.T. LEE (1990), On surface curvature computation from level set contours, *Proc. 10th Int. Conf. Pattern Recognition*, Atlantic City, NJ, 155-160.

[Tanemura *et al.* 1983] M. TANEMURA, T. OGAWA AND N. OGITA (1983), A new algorithm for three-dimensional Voronoï tesselation, *J. Comp. Phys.*, **51**, 191-207.

[Tarjan,Van Wyk-1988] R.E. TARJAN AND C.J. VAN WYK (1988), An $O(n\log\log n)$-time algorithm for triangulating a simple polygon, *SIAM J. Comput.*, **17**, 143-178.

[Taubin-1992] G. TAUBIN (1992), Rasterizing implicit curves by space subdivision, *IBM Research Report*, **RC 17913**.

[Taubin,Rossignac-1998] G. TAUBIN AND J. ROSSIGNAC (1998), Geometric compression through topological surgery, *ACM Trans. on Graphics*, **17**(2), 84-115.

[Thacker-1980] W.C. THACKER (1980), A brief review of techniques for generating irregular computational grids, *Int. j. numer. methods eng.*, **15**, 1335-1341.

[Thiriet *et al.* 1998] M. THIRIET, P. BRUGIÈRES, F. RICOLTI, A. GASTON AND J. BITTOUN (1998), Modélisation numérique de l'écoulement dans les anévrismes cérébraux, *Proc. SNFR*, Paris.

[Thiriet *et al.* 1998] M. THIRIET, S. PIPERNO, G. MALLANDAIN, P. FREY, J. BITTOUN AND A. GASTON (1998), Computational models of flow in cerebral aneuvrisms, In *Proc. 11th Conf. of the ESB*, Toulouse, July 8-11, 1998.

[Thirion-1993] J.P. THIRION (1993), The marching lines algorithm: new results and proofs, *RR INRIA*, **1881**.

[Thompson-1982a] J.F. THOMPSON (1982), Numerical grids generation, *Appl. Math. and Comp.*, **10-11**.

[Thompson-1982b] J.F. THOMPSON (1982), *Elliptic grids generation, numerical grids generation*, Elsevier Science, 79-105.

[Thompson *et al.* 1985] J.F. THOMPSON, Z.U.A. WARSI, C.W. MASTIN (1985), *Numerical grids generation, foundations and applications*, North Holland.

[Thompson-1987] J.F. THOMPSON (1987), A general three dimensional elliptic grid generation system on a composite block-structure, *Comp. Meth. Appl. Mech. and Eng.*, **64**.

[Thompson *et al.* 1999] J.F. THOMPSON, B.K. SONI AND N.P. WEATHERILL (1999), *Handbook of grid generation*, CRC Press.

[Todd,McLeod-1986] P.H. TODD AND R.J.H. MC LEOD (1986) Numerical estimation of the curvature of surfaces, *Computer Aided Design*, **18**, 33-37.

[Turk-1992] G. TURK (1992), Re-tiling polygonal surfaces, *Computer Graphics*, **26**(2), 55-64.

[Turkiyyah *et al.* 1997] G.M. TURKIYYAH, D.W. STORTI, M. GANTER, H. CHEN AND M. VIMAWALA (1997), An accelerated triangulation method for computing the skeletons of free-form solid models, *Computer Aided Design*, **29**(1), 5-19.

[Vallet-1990] M.G. VALLET (1990), Génération de maillages anisotropes adaptés - Application à la capture de couches limites, *RR INRIA* **1360**.

[Vavasis,Ye-1996] S. VAVASIS AND Y. YE (1996), A primal dual accelerated interior point method whose running time depends only on A, *Math. Progr.*, **74**, 79-120.

[Verfurth-1996] R. VERFÜRTH (1996), *A review of a posteriori error estimation and adaptive refinement techniques*, Wiley Teubner.

[Velho *et al.* 1997] L. VELHO, L.H. DEFIGUEIREDO AND J. GOMES (1997), A methodology for piecewise linear approximation of surfaces, *Journal of the brazilian computer Society*, **3**(3), 30-42.

[Voronoï-1908] G. VORONOÏ (1908), Nouvelles applications des paramètres continus à la théorie des formes quadratiques. Recherches sur les parallélloèdres primitifs. *Journal Reine angew. Math.*, **134**.

[Walton,Meek-1996] D.J. WALTON AND D.S. MEEK (1996), A triangular G^1 patch from boundary curves, *Comp. Aided Design*, **28**, 113-123.

[Wang, Liang-1989] Y.F. WANG AND P. LIANG (1989), A new method for computing intrinsic surface properties, *Proc. Conf. Computer Vision and Pattern recognition*, San Diego, CA, 235-240.

[Watson-1981] D.F. WATSON (1981), Computing the n-dimensional Delaunay Tesselation with applications to Voronoï polytopes, *Computer Journal*, **24**(2), 167-172.

[Weatherill-1985] N.P. WEATHERILL (1985), The generation of unstructured grids using Dirichlet tesselation, MAE report no. 1715, Princeton Univ.

[Weatherill-1988] N.P. WEATHERILL (1988), A method for generating irregular computational grids in multiply connected planar domains, *Int. j. numer. methods fluids*, **8**.

[Weatherill-1988] N.P. WEATHERILL (1988), A strategy for the use of hybrid structured-unstructured meshes in CFD, *Num. Meth. for Fluids Dynamics*, Oxford University Press.

[Weatherill-1990] N.P. WEATHERILL (1990), The integrity of geometrical boundaries in the 2-dimensional Delaunay triangulation, *Commun. numer. methods eng.*, **6**, 101-109.

[Weatherill,Hassan-1994] N.P. WEATHERILL AND O. HASSAN (1994), Efficient three-dimensionnal Delaunay triangulation with automatic point creation and imposed boundary constraints, *Int. j. numer. methods eng.*, **37**, 2005-2039.

[Weiler-1985] K. WEILER (1985), Edge-based data structure for solid modeling in curved-surface environments, *IEEE Computer Graphics and Applications*, **5**(1), 21-40.

[Weiler-1986] K. WEILER (1986), Topological structures for geometric modeling, *PhD thesis*, Computer and System Engineering, Rensselaer Polytechnic Institute, Troy, NY.

[Wilhelms,vanGelder-1990] J. WILHELMS AND A. VAN GELDER (1990), Topological considerations in isosurface generation, *Comput. Graphics*, **24**(5), 79-86.

[Winslow-1964b] A.M. WINSLOW (1964), Equi-potential zoning for two-dimensional meshes, *R. no. UCRL-7312.*

[Winslow-1967] A.M. WINSLOW (1967), Numerical solution of the quasilinear Poisson equation in a non uniform triangle mesh, *J. Comp. Phys.*, **1**

[Wirth-1986] N. WIRTH (1986), *Algorithms and Data-Structures*, Prentice Hall.

[Wyvill *et al.* 1986] G. WYVILL, C. MCPHEETERS AND B. WYVILL (1986), Data structure for soft objects, *The Visual Computer*, **2**, 227-234.

[Wordenweber-1984] B. WORDENWEBER (1984), Finite element mesh generation, *Computer Aided Design*, **16**, 285-291.

[Wright,Jack-1994] J.P. WRIGHT AND A.G. JACK(1994), Aspects of three-dimensional constrained Delaunay meshing, *Int. j. numer. methods eng.*, **37**, 1841-1861.

[Wu,Houstis-1996] P. WU AND E.N. HOUSTIS (1996), Parallel adaptive mesh generation and decomposition, *Engineering with Computers*, **12**, 155-167.

[Yao-1981] A.C. YAO (1981), A lower bound to finding convex hulls, *J. ACM*, **28**, 780-787.

[Yerry,Shephard-1983] M.A. YERRY AND M.S. SHEPHARD (1983), A modified-quadtree approach to finite element mesh generation, *IEEE Computer Graphics Appl.*, **3**(1), 39-46.

[Yerry,Shephard-1984] M.A. YERRY AND M.S. SHEPHARD (1984), Automatic three-dimensional mesh generation by the modified-octree technique, *Int. j. numer. meth. eng.*, **20**, 1965-1990.

[Yerry,Shephard-1985] M.A. YERRY AND M.S. SHEPHARD (1985), Automatic three-dimensional mesh generation for three-dimensional solids, *Comp. Struct.*, **20**, 31-39.

[Yu *et al.* 1991] X. YU, J.A. GOLDAK AND L. DONG (1991), Constructing 3D discrete medial axis, *Proc. ACM Symp. Solid Modeling Found. and CAD/CAM Applic.*, Austin, 481-492.

[Zavattieri *et al.* 1996] P.D. ZAVATTIERI, E.A.DARI AND G.C. BUSCAGLIA (1996), Optimization strategies in unstructured mesh generation, *Int. j. numer. methods eng.*, **39**, 2055-2071.

[Zheng *et al.* 1996a] Y. ZHENG, R.W. LEWIS AND D.T. GETHIN (1996), Three-dimensional unstructured mesh generation: Part 1. Fundamental aspects of triangulation and point creation, *Comput. Methods Appl. Mech. Engrg.*, **134**, 249-268.

[Zheng *et al.* 1996b] Y. ZHENG, R.W. LEWIS AND D.T. GETHIN (1996), Three-dimensional unstructured mesh generation: Part 2. Surface meshes, *Comput. Methods Appl. Mech. Engrg.*, **134**, 269-284.

[Zhu *et al.* 1991] J.Z. ZHU, O.C. ZIENKIEWICZ, E. HINTON AND J. WU (1991), A new approach to the development of automatic quadrilateral mesh generation, *Int. j. numer. methods eng.*, **32**, 849-866.

[Zienkiewicz-1977] O.C. ZIENKIEWICZ (1977), *The Finite Element Method*, McGraw-Hill, London.

[Zienkiewicz,Phillips-1971] O.C. ZIENKIEWICZ, D.V. PHILLIPS (1971), An automatic mesh generation scheme for plane and curved surfaces by isoparametric co-ordinates, *Int. j. numer. methods eng.*, **3**, 519-528.

[Zienkiewicz-Zhu-1991] O.C. ZIENKIEWICZ AND X. ZHU (1991), Adaptivity and mesh generation, *Int. j. numer. methods eng.*, **32**, 783-810.

Index

CET OUVRAGE A ÉTÉ COMPOSÉ PAR
HERMÈS SCIENCE PUBLICATIONS
REPRODUIT ET ACHEVÉ D'IMPRIMER
PAR L'IMPRIMERIE FLOCH À MAYENNE
EN MAI 2000.

DÉPÔT LÉGAL : MAI 2000.
N° D'IMPRIMEUR : 48681.