Chapter 8

Other types of mesh generation methods

Introduction

This chapter briefly discusses some mesh generation methods which do not belong to the classical types covered in the previous chapters. The fact that there is such a wide variety of methods is some indication of the richness of the subject but also indicates that there is no one method[1] that is a universal solution for all possible situations. Thus, for instance, a semi-automatic method is sometimes more powerful, more flexible or easier to use for some cases compared with a purely classical automatic method. Moreover, some domains are defined by means of data input that is not directly suitable for an automatic approach.

★
★ ★

The first section discusses product methods, which represent an elegant meshing method when the geometry of the domain has the required aspect, *i.e.,* a cylindrical analogy in terms of topology. Grid or pattern based methods are then discussed which complete a mesh of a given domain starting from a given mesh composed of a simple grid or a partition covering the whole domain by means of a predefined pattern which is repeated. Therefore one can find the grid or the given patterns in the domain (or in a portion of this domain). A mesh generation method by means of optimization is briefly presented which is capable of handling problems with large deformation (such as forming processes). Moreover, this approach allows for more specific applications.

Constructing a quad mesh (in two dimensions) is then discussed in sections four and five where indirect approaches (by means of triangle combination) and direct approaches are given respectively. The extension to hex mesh construction in three dimensions can be found in section six. Then to conclude (without claiming

[1]As least, as far as we know.

exhaustivity), we mention some mesh generation methods that are well-suited to some specific applications. In particular, we focus on methods resulting in radial meshes and methods that make use of a recursive domain partitioning.

8.1 Product method

The aim of a product method consists in creating elements in d dimensions from the data, on the one hand, of elements in p dimensions, $0 < p \leq d$, mapped in the space of d dimensions, and, on the other hand, of a meshed line serving as a *generation line.*

8.1.1 Basic principles

Thus, locally, a point (*i.e.,* an element reduced to a point, a 0-dimensional item) defined in the p-dimensional space and seen in a d-dimensional space produces a series of segments (item of 1-dimension) defined in the d-dimensional space. Similarly, a segment (item of 1-dimension) defined in the p-dimensional space produces quadrilaterals in d dimensional space. A triangle serves as a support for creating pentahedra, while a quadrilateral produces hexahedra (Figure 8.1).

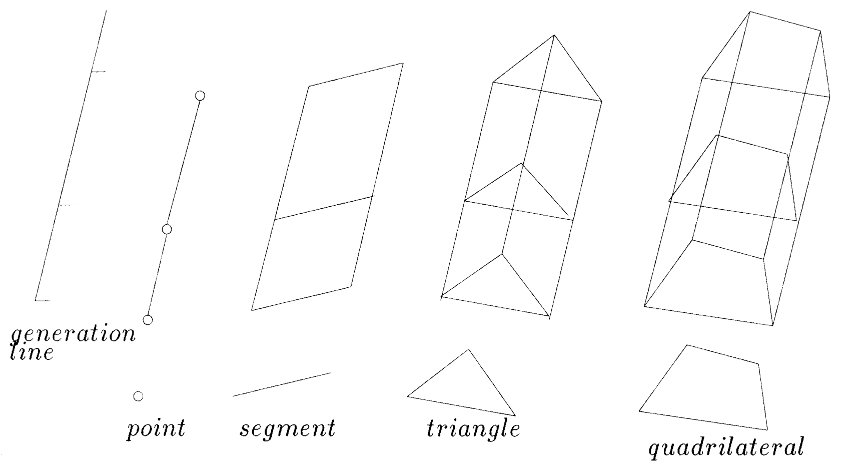

Figure 8.1: *Correspondences. On the left we show the generation line, then the construction by a product method associated with a point, a segment, a triangle and a quad.*

It could be observed that the basic mesh appears, in terms of topology, as a layer of the resulting mesh, such a layer being defined for each discretization step in the generation line. In this way, we obtain a cylindrical topology (which makes it clear in which cases the method applies).

Degeneracies may be encountered for some special positions of the generation line with respect to the given mesh serving as data. Such a phenomenon can be seen as the result of a merging operation where two vertices collapse due to some

property of invariance. In this case segments may produce, not only quadrilaterals, but triangles (two vertices collapse then resulting in a three vertex element), triangles may produce degenerated pentahedra and quadrilaterals may produce degenerated hexahedra. Depending on the geometry, the degenerate elements are valid, or not, in the usual finite element context. In this respect, for example, hexahedra may degenerate into pentahedra, which are admissible elements, or into non-admissible elements (Figure 8.2) with regard to the expected usage of these elements, while pentahedra may lead to the creation of tetrahedra or non-admissible elements (same figure and again related to the future application).

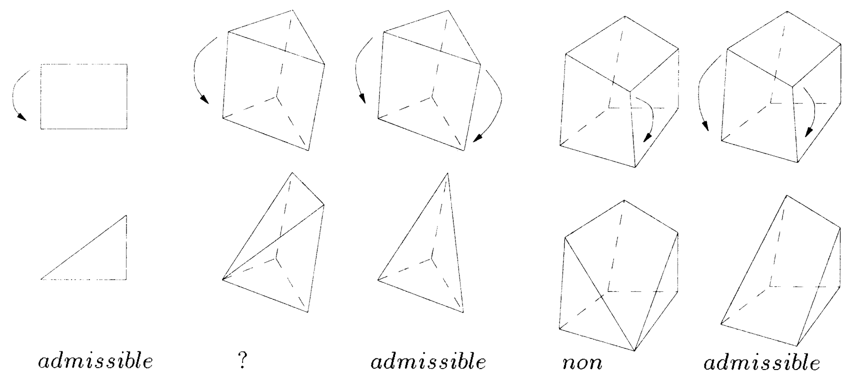

Figure 8.2: *Correspondences and possible degeneracies. On the left is a quad that degenerates into a suitable triangle. Then two forms of degeneracies for a pentahedral element are given, one of them being admissible while the other is suitable or not depending on the envisaged application. Finally, two patterns related to hex degeneracies are displayed, only one of the patterns being valid.*

8.1.2 Computational issues

In what follows, we discuss two methods that allow for the computation of the vertices in the various layers of the desired mesh. Then, we add some details about how to implement a product method.

Computing the intermediate vertices ("semi-manual" method). The vertices of the various layers are defined in a semi-automatic manner through a description of these layers. The possible choices, [George-1993], correspond to the following data input :

- a constant stepsize or a number of layers between the initial section (the basis) and the final one,
- a variable stepsize from one section to the next,
- the full definition of a function that associates a vertex position for each section with the current vertex position in the initial section,

- the full definition, for a given section (the source), of the way in which the next section (the target) must be constructed,
- etc.

Computing the intermediate vertices ("semi-automatic" method). More automatic, this method completes the vertices in the sections lying between two reference sections, [Staten *et al.* 1998], the first serving as a *source*, the other being the *target*. The first section is the initial section or basis (once one considers the entire domain). The target section is associated with an intermediary section or with the final one (the last target section). At the same time, we give how many sections must be constructed between the source and the target and we describe the side of the "cylinder" contained between these two particular sections by means of discretized edges meshed with consistency from one to the other. The source and the target sections are assumed to be known in a parametric form. Then the domain could be subdivided into several portions and an automatic algorithm can be applied in each of them in order to complete the intermediary points based on the available data input.

The computational scheme is as follows :

- one constructs a background mesh for the source section in its parametric space based on the points lying on the sides on this section. As assumed, this mesh is also a background mesh for the target section and thus will be used when the various intermediate sections are dealt with,
- one classifies the (internal) points in the source section with respect to the background mesh (we find the triangle that contains such a point and we obtain its barycentric coordinates) and the same is done for the points in the target section. At completion, two sets of three values[2] $(a_i, b_i, c_i)^{source}$ and $(a_i, b_i, c_i)^{target}$ are available for the point P_i considered in the source (resp. target) section,
- one computes the distances between these internal points (in $\mathbb{R}^3$) with respect to the triangles whose vertices (denoted by S_i in what follows) are the images of the three vertices of the triangles in the background mesh containing these points (now considered in the parametric space). To this end, we use the following formula, for point P_i :

$$d_i = ||a_i\, S_1 \,+\, b_i\, S_2 \,+\, c_i\, S_3 \,-\, P_i\,||\,,$$

with respect to the source section for the points in this section and using a similar formula for the points in the target section with respect to this section. In this way, we obtain two distances d_i^{source} and d_i^{target} that measure

[2]If the shape (the geometry) of the source and the target sections are identical, these two sets are the same. Otherwise, for instance, the triangle containing the source point is not necessarily that containing the corresponding target point assumed to be in front. In this case, to retrieve the same element, negative barycentric coordinates are allowed.

the distortion[3] at point P_i for the two sections bounding the domain being processed.

This information is then used to compute the position of the internal points in the sections contained between the source section and the target section. This calculus, for the point P_i in a given section, includes a linear interpolation between $(a_i, b_i, c_i)^{source}$ and $(a_i, b_i, c_i)^{target}$, a function of the section (defined through the points supplied in the sides of the above mentioned cylinder) and a correction step that modifies this value based on d_i^{source} and d_i^{target} (after an interpolation).

Implementation remarks The position of the intermediate points, the case where a degeneracy occurs or when the final section becomes identical to the first section (for instance in the case of a rotation) having been successfully established, the computational aspect does not induce any difficulty. In practice, the mesh generation problem comes down to numbering the vertices of the different layers or sections corresponding to the discretization of the generation line. This numbering step is rather similar to that already seen in a multibloc method, Chapter 4. Once this numbering has been done, the element vertex enumeration is trivial.

8.1.3 Limits

Product methods (also referred to as *extrusion* methods) can be applied to domains which have the desired properties, *i.e.*, one of the following topological types :

- cylindrical topology : the domain can be described via the data of a two-dimensional mesh and a generation line defining layers with which the three-dimensional elements are associated,
- hexahedral topology : the domain can be described via the data of a one-dimensional mesh and a generation line to construct a two-dimensional mesh which, in turn, is coupled with a second generation line and produces the desired three-dimensional elements.

Note, as a conclusion about the limits, that the validity (positive surface areas or volumes, absence of auto-intersections (inter-penetrations or overlapping elements)) of the resulting mesh must be checked explicitly, which may increase the computational cost.

8.1.4 Application examples

Figures 8.3 and 8.4 display two examples constructed by a product method applied respectively to a meshed line and a two-dimensional mesh serving as data and a rotation used as a generation line. In the first example, the generation line (a circle) is not connected with the data and classical expected elements are constructed (quads), while, in the second example, the generation line is partially common to

[3]If, for example, the source section is planar, we have $d_i^{source} = 0$.

some edges of the data and pentahedra (instead of hexahedra) are produced in this region.

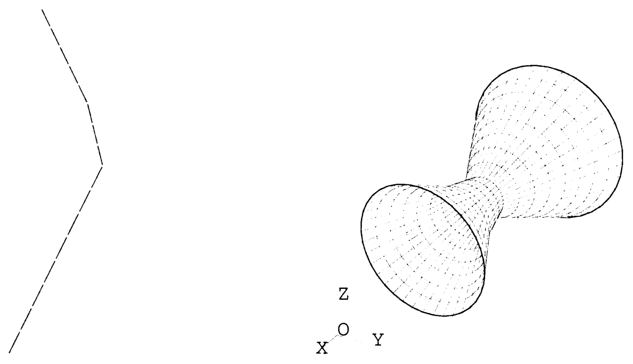

Figure 8.3: *Line serving as data (left-hand side) and resulting mesh (right-hand side) obtained by a product method corresponding to a rotation.*

The examples in Figure 8.5 have been completed using a method described by [Staten *et al.* 1998] which is used in ANSYS. One can see a mesh resulting from translation and rotation (left-hand side) and a mesh resulting from translation (right-hand side). The latter example has been completed in three steps whose respective meshes, after merging, provide the final mesh. This partition into three parts has been made so as to make the product method suitable for each of them. This partition is defined so as to permit a compatible mesh merge (see Chapter 17 on how to merge two meshes).

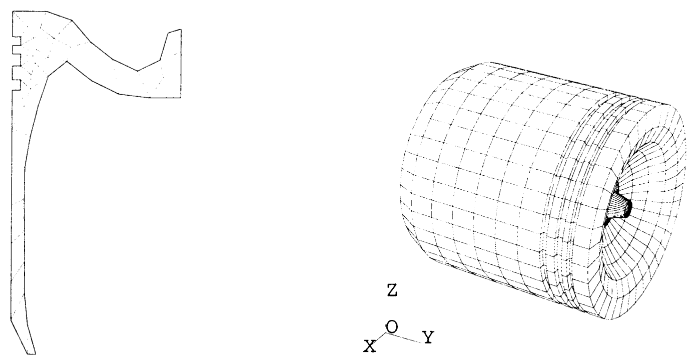

Figure 8.4: *2D mesh serving as data (left-hand side) and resulting mesh (right-hand side) obtained by a product method corresponding to a rotation.*

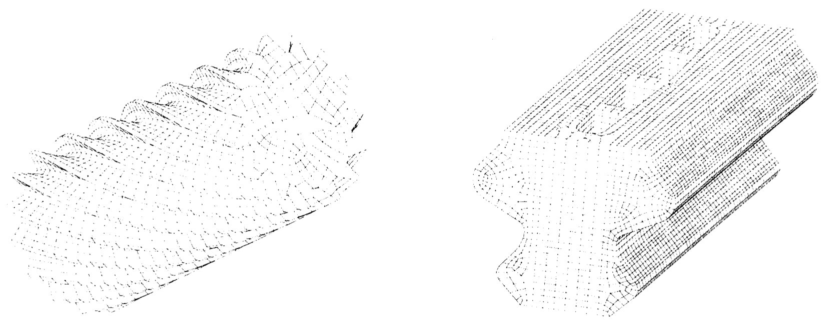

Figure 8.5: *Examples of meshes resulting from the product method incorporated in ANSYS. Left-hand side : the various sections result from the basis after a translation combined with a rotation. Right-hand side : the domain is split into three parts and, for each of them, a translation of the corresponding basis is used.*

8.2 Grid or pattern-based methods

To some degree, these methods recall some of the constructions seen in Chapter 5 and also some ideas that will be seen in Chapter 16. The domain of interest is merged into a regular grid G composed of uniform squares (in two dimensions) or uniform cubes (in three dimensions). The cell size is a function of the smallest distance between two contour points of the domain and depends also on available memory resources. A meshing process based on the (easily) so-defined grid can be developed. It involves the following phases :

Step 1 : the removal of the cells of G which do not intersect the domain;

Step 2 : the processing of the cells of G containing a portion of the boundary of the domain; two variations can be advocated:

- a cell whose intersection with the boundary is not empty is considered as a mesh element; in this way, the final mesh will only be a piecewise linear approximation of the given domain (the accuracy of this approximation depends on the stepsize of the grid cells),
- or this type of cell is modified in such a way that the boundary is better approximated.

Step 3 : the enumeration of the mesh elements :

- a purely internal cell becomes a mesh element (or can be split into simplices);
- a cell intersecting the boundary is considered as above. It defines a mesh element or it can be split into one or several triangular or quad elements, or tetrahedral or hex elements, such a splitting being based

on the analysis of the different possible configurations and the use of the corresponding splitting pattern (Chapters 17 and 18).

Simple in principle, this method is unlikely to be suitable when the boundary discretization, serving as data, includes widely different items (edges in two dimensions) in terms of size while this situation generally requires a very fine stepsize and then may lead to an overflow in the number of cells of grid G, and consequently in the number of elements of the resulting mesh. Consequently, except for some specific problems, more flexible methods must be used (such as the quadtree-octree methods as discussed in Chapter 5). Nevertheless, for certain specific applications (for example for forecast computation), a relatively regular mesh is a source of simplification at the computational level.

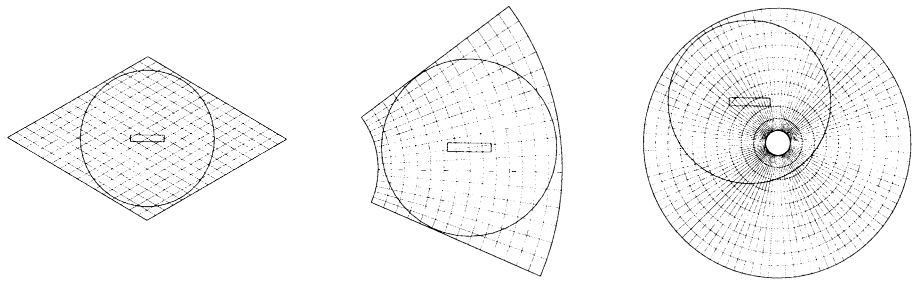
Figure 8.6: *The grid made of parallelograms (left), the portion of the ring made of radial quadrangles (middle) and the ring made of radial quads (right).*

The above approach can be followed not with a regular grid, but when a pre-specified mesh is given which encloses the domain. In this way, using as pre-specified mesh a mesh composed of pre-defined patterns results in a final mesh where most of the elements conform to this pattern. Thus, most of the domain is covered with the pre-specified patterns and the remaining regions (*i.e.*, the regions intersected or close to the boundary where the given pattern is not suitable) can be dealt with either using a general purpose mesh generation method or a specific processing of the patterns that intersect the domain boundary.

Figures 8.6 and 8.7 demonstrate the principle of this method. In the first figure, one can see the domain in question, the part inside a circle having a rectangular hole at its centroid. Three enclosing meshes are depicted. On the left-hand side, the pattern is a parallelogram, in the middle, we have a part of a ring covered by quads and, finally, on the right-hand side, we have a ring covered with radial quads which are excentered with respect to the domain. In the other figures, we illustrate a plausible scenario as to what the construction could be. We see successively the patterns with (at least) one portion inside the domain, i), and in ii), the retained patterns, *i.e.*, among the previous, we have discarded those partially included in

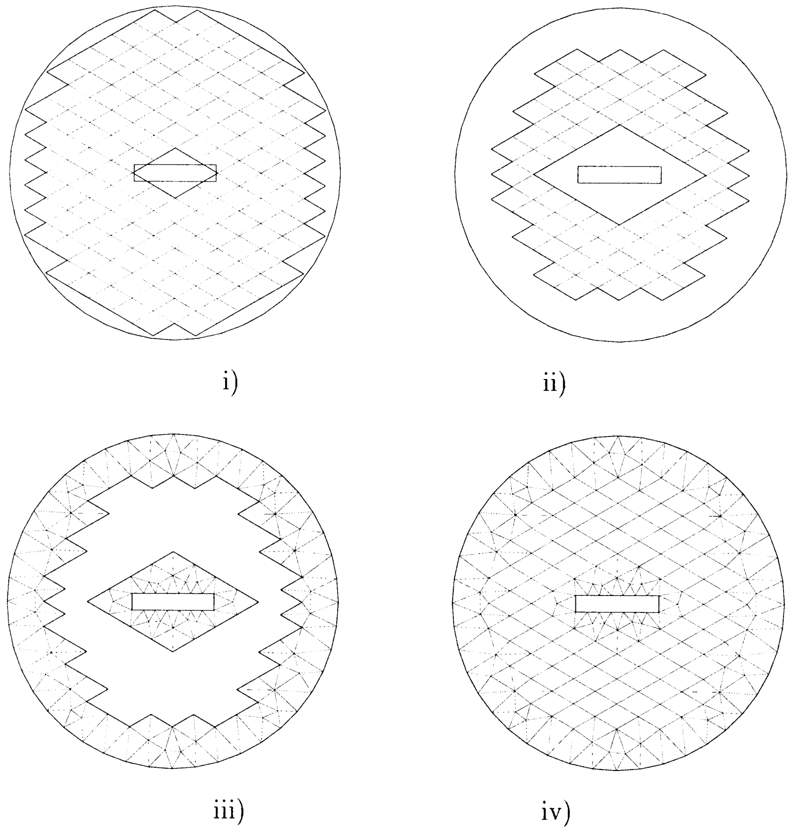
Figure 8.7: *From i) to iv), one can see the quads strictly included in the hollow disc, those included in this disc that are "consistent" with the boundary, the triangle mesh resulting from an automatic mesh generator and the final mesh of the disc.*

the domain and those judged too close to the domain boundaries to define a large enough region in the vicinity of these boundaries. Then, in iii), the not yet meshed regions are meshed, this point concerns the regions close to the boundaries. Finally, in iv), the resulting mesh is visible and is formed by the patterns that have been retained and the mesh of the boundary regions as completed previously.

Various choices about the pattern are possible. For instance, in two dimensions, one may think of the pattern related to the union of six equilateral triangles (a regular hexagon) or to the configuration composed by elements whose vertices are located in a radial manner with respect to a given point.

8.3 Optimization-based method

Proposed by [Coupez-1991] in the context of forming processes where the geometry can change dramatically from one time step to another, this method can also be used as a mesh generation method or even as a mesh optimization method or a mesh correction method.

Before going further in the discussion, we now introduce the theoretical background which serves at the basis of this method.

8.3.1 Theoretical background

For the sake of simplicity, we consider a two-dimensional problem where the elements are triangles only.

Notations and definitions. A triangle K is an oriented triple of (distinct) integers. A mesh edge is a pair of (distinct) integers. The boundary of K, denoted as δK, is defined by the three edges of K. Now, let $\mathcal{T}$ be a set of triangles, we denote by $\mathcal{E}$ the set of the edges of the elements in $\mathcal{T}$, $\mathcal{S}$ the set of the element vertices and $\delta\mathcal{S}$ the set of vertices in $\delta\mathcal{T}$, the boundary of $\mathcal{T}$, *i.e.*, both $\mathcal{S}$ and $\delta\mathcal{S}$ are a set of integers (or indices). With these notations, we first introduce the notion of a so-called *mesh topology*.

Definition 8.1 *A given mesh $\mathcal{T}$ conforms to a mesh topology if $\mathcal{E}$, the corresponding set of edges, is such that*

$$\forall e \in \mathcal{E} \ , 1 \leq card\{\, K \in \mathcal{T} \text{ such that } e \in \delta K\} \leq 2 \, ,$$

where $card\{.\}$ stands for the number of elements in set $\{.\}$. In other words, any edge in $\mathcal{E}$ is shared by two triangles (manifold case in three dimensions) or is a boundary edge.

Given a mesh topology $\mathcal{T}$, its boundary $\delta\mathcal{T}$ consists of the edges in $\mathcal{E}$ which belong to only one element.

Remark 8.1 *Unlike a mesh (as defined in Chapter 1), a mesh topology relies only on topological properties. In this respect a mesh can be defined as a mesh topology whose corresponding vertex coordinates result in a valid mesh (in the usual sense) while a mesh topology is valid only in terms of neighboring relationships.*

Following this remark, we can give a rough idea of the present mesh generation method. First, a mesh topology is constructed, then it is modified so as to produce a (valid) mesh. To this end, a local operator is introduced.

A simple local operator. Let $\mathcal{T}$ be a mesh topology and $\mathcal{S}$ the corresponding vertex set, let P be a point, then joining P with the edges of $\delta\mathcal{T}$ results in a new mesh topology as can be easily seen. In what follows, this operator will be referred to as $Op(P, \delta\mathcal{T})$. A more precise analysis of this operator leads to examining two situations :

- either $P \in \delta\mathcal{S}$ (point P is identified with index P),
- or $P \notin \delta\mathcal{S}$.

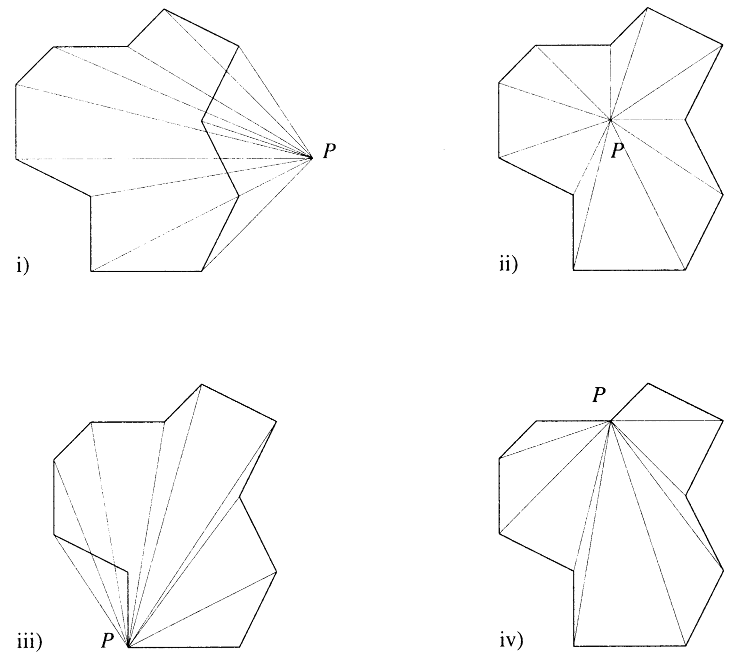

Figure 8.8: *Various applications of the $Op(P, \delta\mathcal{T})$ operator. The point P is outside $\mathcal{T}$, case i), P inside $\mathcal{T}$, case ii), P is a boundary point, cases iii) and iv).*

In the first situation, the operator basically joins the vertices in $\delta\mathcal{S}$, apart from P and its two immediate neighboring vertices, with P. In the second configuration, the operator leads to connecting all the vertices in $\delta\mathcal{S}$ with a given point P which could be inside or outside the region whose boundary is $\delta\mathcal{T}$. In Figure 8.8, one can see the first situation (cases iii) and iv)) as well as the second situation (cases i) and ii)).

Remark 8.2 *If $\mathcal{T}$ is a convex region, then applying $Op(P, \delta\mathcal{T})$ with P inside $\mathcal{T}$ or P is a member of $\delta\mathcal{T}$ results in a (valid) mesh following Definition (1.7).*

However, when $\mathcal{T}$ is an arbitrary mesh topology and not a valid mesh, applying $Op(P, \delta\mathcal{T})$ does not result, in general, in a (valid) mesh. The question is then to find some criteria to govern the application of the operator, e.g., how to repeat this single operator with appropriate points P so as to complete a (valid) mesh. Two criteria are introduced, one is based on the element surface area while the other concerns the element shape quality.

Given $\mathcal{T}$, the surface area of this set of elements, $Sur(\mathcal{T})$, is the sum of the (non oriented) surface areas of the triangles in $\mathcal{T}$. Similarly, the quality of $\mathcal{T}$, $Qua(\mathcal{T})$, is the quality of the worst element in this set. The quality measure is any of the measures discussed in Chapter 18 where the surface area involved in the expression is considered as positive (actually, we take the absolute value of the real surface area).

Some basic observations. A mesh topology $\mathcal{T}$ is not a valid mesh if, at least, one of its elements is negative thus resulting in an overlapping region. Hence, the surface of $\mathcal{T}$ is larger than (or equal to) the surface of the corresponding domain. As a consequence a mesh is a mesh topology whose surface area is minimum. This observation leads to the first idea to govern the operator $Op(P,\delta\mathcal{T})$: *minimize the surface area of the topology.*

A second observation is as follows. Given a mesh topology $\mathcal{T}$, we can apply the $Op(P,\delta\mathcal{T})$ operator for different points P and, in this way, various new mesh topologies can be constructed. In this situation, several new topologies may be equivalent in terms of surface areas. Thus we need a criterion to decide whether a given result is better than another. This is why, a quality based criterion is introduced. Thus, the second key of the method is : *retain the transformation that minimizes the surface area and, at the same time, optimizes the quality criterion.*

Now that we have specified the basic ideas of the method, we can turn to the issue of a meshing algorithm based on these simple ideas.

8.3.2 Synthetic algorithm

In short, given a domain discretization, we initialize a mesh topology and then we modify it by repeated use of the above operator until a valid mesh is completed. To this end, the current mesh topology is analyzed and the regions with negative or badly shaped elements are identified. These regions are considered one after the other. Given a region, we define the corresponding (sub) mesh topology and we apply the operator Op to this (sub) mesh topology.

A natural (sub) topology is the *ball* of a given vertex (see Chapter 2), *i.e.*, the set of elements that share this point. Let P be a given point, then its ball is denoted by $\mathcal{B}_P$. In fact, any connected set of elements can be considered as a (sub) topology. For instance, the two triangles sharing a given edge is a natural candidate.

Given a mesh topology $\mathcal{T}$ and its boundary $\delta\mathcal{T}$, we define $\mathcal{V}$ the set of points P such that

$$\mathcal{V} = (P \in \mathcal{P}\ ,\ Sur(Op(P,\delta\mathcal{T})) = \min_{Q\in\mathcal{P}} Sur(Op(Q,\delta\mathcal{T})))\,, \qquad (8.1)$$

in other words, $\mathcal{V}$ is the set of points (indices) that, after using Op, minimizes the surface area of the resulting mesh topology. In the above expression, $\mathcal{P} = \delta\mathcal{S} \cup G$ where G could be an arbitrary point (see below).

We also introduce $\mathcal{Q}$ the set of points P such that

$$\mathcal{Q} = (P \in \mathcal{P}\ ,\ Qua(Op(P,\delta\mathcal{T})) = \max_{Q\in\mathcal{P}} Qua(Op(Q,\delta\mathcal{T})))\,, \qquad (8.2)$$

i.e., $\mathcal{Q}$ is the set of points (indices) that, after using Op, optimizes the quality of the resulting mesh topology.

Now, the mesh generation algorithm is as follows :

- Initialization : $\mathcal{T} = Op(P,\delta\mathcal{S})$ where P is the point in $\delta\mathcal{S}$ which is the nearest point to the centroid[4] of $\delta\mathcal{S}$.
- (A) $opt = .false.$
- (B) Loop over the points of $\mathcal{T}$, let P be the current point, then
  - consider the ball $\mathcal{B}_P$ as a sub mesh topology,
  - $\mathcal{P}$ is the set of vertices in $\delta\mathcal{B}_P$,
  - we consider a point G which is the centroid of the points in $\delta\mathcal{B}_P$, *i.e.*, $G = \frac{\sum P_i}{n_P}$, where P_i denote a vertex in $\mathcal{T}_P$ other than P and n_P is the number of such vertices.
  - $\mathcal{P} = \mathcal{P} \cup \{G\}$,
  - we apply $Op(P,\delta\mathcal{B}_P)$ where $P \in \mathcal{P}$ and we define the corresponding sets $\mathcal{V}$ and $\mathcal{Q}$,
  - if $P \in \mathcal{Q} \cap \mathcal{V}$, return to (B),
  - otherwise, pick the point P in $\mathcal{Q}\cap\mathcal{V}$ and retain the corresponding mesh topology, the current topology is now the following :

    $$\mathcal{T} = \{\mathcal{T} - \mathcal{B}_P\} \cup Op(P,\delta\mathcal{B}_P)$$

    fix $opt = .true.$ and return to (B).
- End of Loop
- if $opt = .true.$, then return to (A), otherwise the current state is stable.

On completion, a stable mesh topology is obtained. The issue is to make sure that this state corresponds to a valid mesh.

Since the topology is valid (due to the way it is constructed), the (global) mesh validity is ensured if the (signed) surface of all elements is strictly positive. In fact, if there is a triangle with a negative surface, then an overlapping element exists (a null surface implies that there is a completely flat element while a very "small" surface, with regard to the edge lengths, indicates the existence of a bad quality element).

If the stable situation corresponds to a valid mesh (in the above sense), then the resulting mesh is a suitable mesh possibly after an optimization stage. If not, we have a stable situation which is invalid. This case deserves some comments. The stability is related to the choice of the sub-topology used in the method. In the case of a stable situation which is an invalid mesh, the only thing we can try is to modify the sub-topologies in order to make it possible to continue the process. In this way, the surface series still continues to decrease since the mesh is modified. As an example of other types of sub-topologies, we can consider the

[4] This is one possible candidate, in fact, any point in $\delta\mathcal{S}$ could be used.

pairs of adjacent triangles (or some other patterns) instead of the balls used in the previous algorithm.

In theory, such a process converges since we face a decreasing series whose targeted bound (the right surface of the domain) is known in advance and because the number of combinations (sub-topologies) that can be constructed is a finite number. Thus, the convergence holds because, in the worst case, it is sufficient to examine all these possibilities (irrespective of the cost of such a method related to the underlying combinatorial aspect).

8.3.3 Computational issues

Various remarks about the computational aspects of the above algorithm can be made.

First, maintaining a (valid) topology relies on checking that the corresponding definition holds and, in addition, that the intersection of the initial topology except the examined ball (the sub-topology, in the case where these sets are the balls) with the sub-topology resulting from operator Op is nothing other than the boundary $\delta\mathcal{B}_P$ of the ball $\mathcal{B}_P$. In other words,

$$(\mathcal{T} - \mathcal{B}_P) \cap Op(P, \delta\mathcal{B}_P) = \delta\mathcal{B}_P\,.$$

Figure 8.9 illustrates a case where this result is not satisfied. Triangle ANB will be formed twice if the sub-topology $Op(N, \delta\mathcal{B}_P)$ is retained.

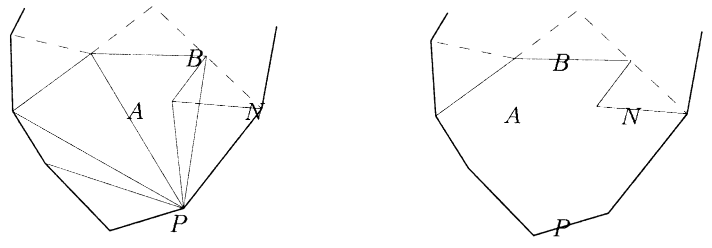

Figure 8.9: *Left-hand side : The initial sub-topology, right-hand side : joining point N with the other points in the boundary of the ball of P again results in triangle ANB which is already an element inside the ball in question.*

Leading on from these observations, it is of interest to define a suitable strategy regarding the way in which the operations are proceeded and the choice of the sub-topologies used. For instance, it is proposed to perform a first stage without any point insertion (by a pertinent choice of Op) and then to start again while point insertions may be used in this stage.

8.3.4 Extension in three dimensions

The above approach extends to three dimensions. In this case, the elements are only tetrahedra, *i.e.*, an oriented quadruple of integers. The set $\mathcal{E}$ is the set of element faces and Definition (8.1) still holds (where e is a triangular face).

The operator $Op(P, \delta\mathcal{T})$ acts on the boundary of the mesh topology $\mathcal{T}$ meaning that point P is connected with the triangles that form this boundary. The two above criteria are also used, the first criterion concerns the volumes of the elements under consideration while the second remains unchanged.

The synthetic scheme of the method is as above while the sub mesh topologies that can be dealt with include the balls of the current points, the shell of the current edges, *i.e.*, (see Chapter 2) the tets that share an edge or, finally, any connected set of tets.

8.3.5 Application examples

As indicated at the beginning, two different applications of this meshing method can be envisaged. First, the method can be seen as a mesh construction method for a domain starting from a discretization of the boundaries of this domain.

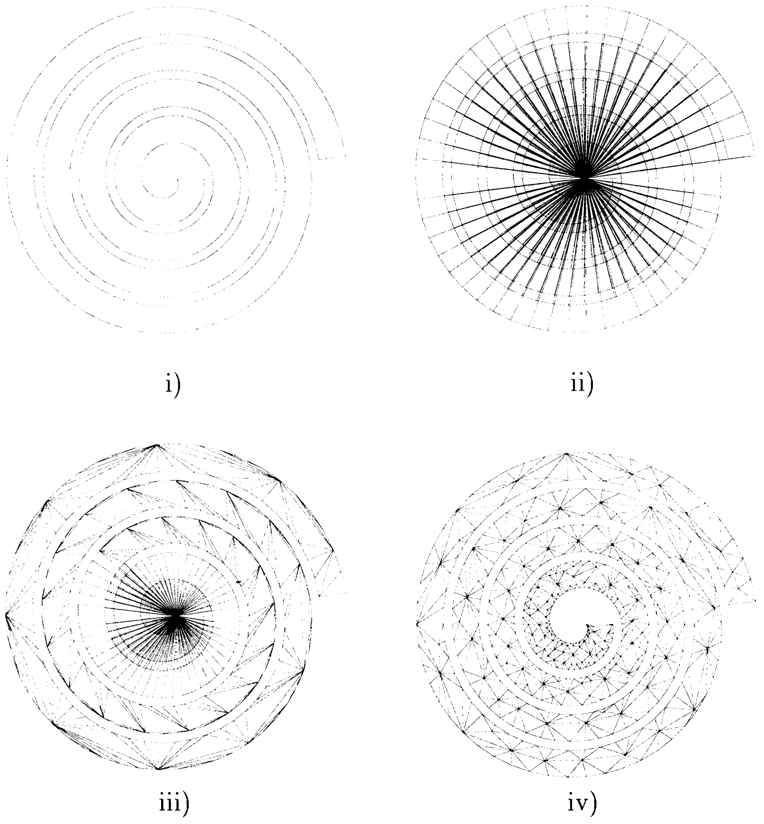

Figure 8.10: *i) : the domain boundary, ii) : the mesh after connecting one point (close to the domain centroid) with all the others. iii) : the mesh during the optimization stage and iv) : the resulting mesh. Indeed, the current topology is now a valid mesh.*

Then, this method can be used as a mesh optimization method or even as a mesh correction procedure that applied to a mesh being modified during a computational process. Another more "exotic" application can be also envisaged that makes it possible to develop an algorithm resulting in the boundary enforcement (in the context of a Delaunay type method, Chapter 7).

We now give some examples of meshes resulting from the present method. Figure 8.10 depicts several steps of a two dimensional example. Figure 8.11 gives a tridimensional case. The geometry is a node, a kind of cylinder (or a pipe) that closes in on itself. Numerous examples, including some very impressive ones, can be found in [Coupez-1991], [Coupez-1996], [Coupez-1997] that concern forming problems. An initial configuration whose geometry is rather simple is modified by stamping since a complex shape is obtained. The initial mesh is used as the initial topology and, due to the large deformations that are applied, this mesh becomes invalid. The method then is used to maintain a valid mesh at each state of the domain deformation.

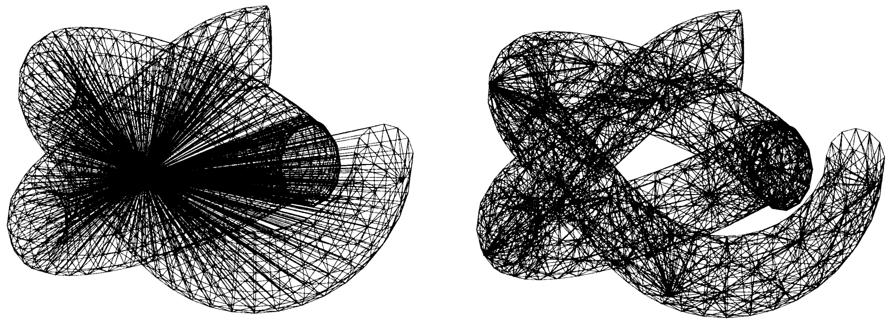

Figure 8.11: *The initial mesh topology resulting from the connexion of one boundary point with all the others (left-hand side) and the resulting mesh (right-hand side). Indeed, the current topology is now a true mesh.*

8.4 Quads by means of triangle combination

Constructing a quad mesh for an arbitrarily shaped domain (in Chapter 4, we saw the case where the domain geometry was suitable for a direct construction, either using an algebraic method or a P.D.E. based method or, finally, by means of a multiblock approach) is a tedious problem which can be handled in two manners. The first approach, a so-called *indirect approach* is based on using a triangular mesh and then to create the quad by triangle combination. The second approach, a *direct approach*, consists in designing a method that directly results in quads. In this section, the first approach is considered while the second approach will be discussed in the following section.

Before going further, we give two preliminary remarks. The first remark is a fundamental observation which concerns the existence of a solution to the problem in question. The second remark, of purely practical interest, mentions the expected cost of either approach.

Remark 8.3 *In general, the existence of a mesh composed only of quads is not guaranteed for an arbitrarily shaped domain. Indeed, provided with a polygonal discretization of a domain boundary, it is clear that, at least, an even number of edges is required if we want to have a solution. If not, a certain number of triangles (one for each connected component) may be necessary in order to cover the domain entirely.*

Remark 8.4 *The approach by means of combination starts from a triangular mesh (assumed to be conforming) of the domain and thus is nothing other than a post-processing of an existing mesh. As a consequence, the design, the validation (robustness, reliability) is only related to this processing that is, in principle, faster than validating an entire mesh construction method.*

Having made these remarks, we now describe an approach based on combination. We assume a favorable case (where the boundary includes an even number of segments) and we consider a domain in the plane (the surface case will be seen below).

8.4.1 Two basic combinations

The simplest pattern corresponds to two adjacent triangles that form a convex polygon. These two triangles make it possible to define one quad. A different pattern is formed by one triangle and its three neighbors. The corresponding polygon then has six sides. It allows, *a priori*, the construction of three quads (a point being introduced, for instance at the centroid of the initial triangle and then joined with the sides of the above polygon thus giving the three quads in question). Figure 8.12 demonstrates these two examples of triangle combination.

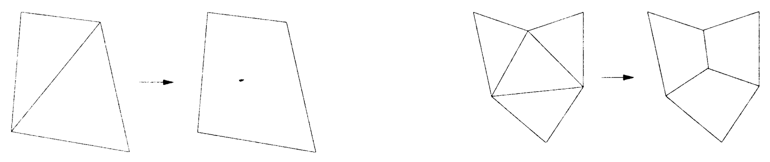

Figure 8.12: *Left-hand side : the combination of two triangles gives one quad, right-hand side, the combination concerns four triangles and results in three quads.*

8.4.2 Selection of a combination

Given one or several combination patterns, the question is to define the pairs (or the sets) of triangles that must be combined to define the quads. The basic technique consists in finding in the triangular mesh a path that makes it possible to pass from one triangle to the other. This path, obtained by visiting the triangles in some way, is stored in a stack. Thus, the triangles are put in a stack and pushed once the quads are constructed.

The construction of the stack (the path) follows some rules. A starting triangle is selected (having two boundary edges if such an element is found). This triangle is the first in the stack.

The current triangle is colored and one of its edges is selected. The neighbor (through this edge) is put into the stack if it is not already colored and if the quad thus formed is valid (actually, this quad is not really constructed). The stacked triangle is the *child* of the initial triangle. When it is no longer possible to continue, the triangles are removed from the stack. When carrying out such an operation we meet two situations.

There is an edge which has not yet been used. The corresponding neighbor is put into the stack as the child of the related triangle. Otherwise, the quad construction begins.

Let us assume that a certain number of triangles has been put into the stack and that some quads exist. Suppose we are at triangle K_1 while going from its neighbor after having passed through the edge e_1 of triangle K_1 (e_1 is then the edge common to the two triangles in question). We try to pass through the edge e_2, edge next to e_1 while in the direct sense. This is possible if, on the one hand, the edge e_2 is not a boundary edge and, on the other hand, the triangle neighbor of K_1 through this edge has not already been processed. If the edge cannot be traversed, we consider the next (using the same convention), in this way we find one triangle, say K_2, or this branch of the stack is no longer active (the edge is a boundary edge or the neighboring element is already in the stack). In the first case, K_2 is put into the stack and we report that this element follows K_1. In the second case, we count the number of *children* of K_1 not already involved in a quad. The children of a given triangle are its neighbors that have not already been taken into account. Therefore, a given element may have zero, one or two children.

If K_i is the last triangle in the stack, we use the following algorithm :

- if the number of children of K_i is null, we remove K_i from the stack and we return to triangle K_{i-1},

- if the number of children of K_i is one and if the triangle is still *active* (*i.e.*, has not been used to construct a quad), we construct a quad by combining K_i and its child while verifying that this element is convex. Then K_i is removed from the stack. If the combination of K_i and its child forms a non convex polygon (or a quad with two consecutive aligned sides), the triangle child will remain and we try to form a quad using K_i and its father if this pattern is adequate,

- if the number of children of K_i is two and if the triangle is still active, we from three quads from K_i and its three neighbors (its two children and its parent), then K_i is removed from the stack.

This algorithm continues as long as the stack is not empty. Once it is empty, the quad mesh is obtained. Nevertheless, some isolated triangles can be present in this mesh. Thus, the resulting mesh is a *mixed* mesh.

Remark 8.5 *The thus-constructed path may not pass through all the triangles in the domain. In such a case, we start again from one triangle not already visited and one path by the thus-defined component can be obtained.*

A solution resulting in a mesh without any triangles, consists in applying the combination strategy on triangles whose size is twice the desired size (in terms of edge length) and then splitting the resulting quad into four elements. The triangles that can still exist are subdivided into three by introducing centroids and edge midpoints. In this way, the mesh has only quads and, due to the method, is a conforming mesh.

8.4.3 Optimization

The resulting meshes, in most cases, are not of good quality everywhere in the domain. Adding strict criteria about this quality during the process is possible *a priori* but impedes the quad creation. In such a case, numerous triangles necessarily exist since the corresponding combinations are rejected. Therefore, it is advisable to leave some degree of freedom in the construction and, since, bad quality elements may appear, to apply an optimization stage *a posteriori*. The optimization tools are the local tools capable of carrying out local configurations (a set of adjacent elements). Chapter 18 contains a detailed description of general purpose optimization tools, so, here, we will just give some brief indications about tools acting on quad elements.

Among these tools, we can find some operators related to certain typical configurations :

- an internal vertex only common to two quads or common to one quad and one triangle can be removed. We then successively obtain one quad or one triangle whose quality is necessarily better than that of the initial configuration,

- a vertex shared by three quads is in general related to a bad quality quad. Then, the initial pattern can be replaced by two quads which are better *a priori*,

- a quad with two (internal) opposite vertices common to only three elements can be removed (the two points are merged and the quad disappears),

- the edge common to two quads can be removed by using the alternative connexion with the two other vertices (we find here the edge swapping operator as described in Chapter 18),

- a set of adjacent quads forming a six side polygon may be replaced using only two quads, etc.

An optimization operator is retained if the resulting quality is better (the initial context is analyzed according to a quality criterion or a topological criterion). Note that the relaxation, on average, of the degree of the internal vertices (see again Chapter 18) is a promising operation for this type of mesh.

8.4.4 Alternate method

In this approach, one examines, for all triangles in the initial mesh, the quality of the quad formed by combining the triangle in question, K, with its neighbors, K_i. Therefore, there are three possibilities *a priori*. Each possible quad is identified by the edge common to K and K_i. With each of these edges is associated a quality value which involves the angle in the quad at the two vertices endpoints of the edge in question.

Let $[ABCD]$ be a quad whose vertices are defined in the direct sense. In the classical isotropic case, the quality measure of a quad is defined at vertex P by means of the formula :

$$\mathcal{Q}_P = \begin{cases} \dfrac{2\theta}{\pi} & \text{if} \quad 0 \leq \theta < \frac{\pi}{2} \\ 2 - \dfrac{2\theta}{\pi} & \text{if} \quad \frac{\pi}{2} \leq \theta < \pi \\ 0 & \text{if} \quad \pi \leq \theta \end{cases}, \tag{8.3}$$

where θ is the angle between three consecutive vertices, P_iPP_j, in the quad. This function varies linearly between 0 (flat quad) and 1 (rectangle). The quad quality is then defined as the minimum value of the quality values in its vertices :

$$\mathcal{Q} = \min_{P \in \{A,B,C,D\}} \mathcal{Q}_P . \tag{8.4}$$

This quality measure can be extended to the general case (anisotropic case). To this end, it is just necessary to define the dot product with regard to a given metric.

Dot product in a given metric. The dot product of two vectors $\vec{u}$ and $\vec{v}$ in the Euclidean space characterized by a metric $\mathcal{M}_2(X)$ is defined as :

$$\langle \vec{u}, \vec{u} \rangle_{\mathcal{M}_2(X)} = {}^t\, \vec{u}\mathcal{M}_2(X)\vec{v} .$$

The norm of a vector w is then given by :

$$||\vec{w}||_{\mathcal{M}_2(X)} = \sqrt{{}^t\vec{w}\mathcal{M}_2(X)\vec{w}} .$$

Angle measurement in a metric. Let A, B and C be three points such that $\overrightarrow{BC} \wedge \overrightarrow{BA} \geq 0$, the notation $\wedge$ stands for the cross product in the classical Euclidean metric. The value of the angle (in radians) between the vectors $\overrightarrow{BC}$ and $\overrightarrow{BA}$ with respect to the metric $\mathcal{M}_2(X)$ is given by :

$$\theta_{B\mathcal{M}_2(X)} = \arccos \left(\frac{\langle \overrightarrow{BC}, \overrightarrow{BA} \rangle_{\mathcal{M}_2(X)}}{||\overrightarrow{BC}||_{\mathcal{M}_2(X)} ||\overrightarrow{BA}||_{\mathcal{M}_2(X)}} \right) . \tag{8.5}$$

Quad quality in a metric. The quality function defined in the isotropic case (Relationship (8.3)) can be extended to an anisotropic metric in the following way :

$$(\mathcal{Q}_P)_{\mathcal{M}_2(X)} = \begin{cases} \dfrac{2\theta_{\mathcal{M}_2(X)}}{\pi} & \text{if} \quad 0 \leq \theta_{\mathcal{M}_2(X)} < \frac{\pi}{2} \\ 2 - \dfrac{2\theta_{\mathcal{M}_2(X)}}{\pi} & \text{if} \quad \frac{\pi}{2} \leq \theta_{\mathcal{M}_2(X)} < \pi \\ 0 & \text{if} \quad \pi \leq \theta_{\mathcal{M}_2(X)} \end{cases},$$

and, thus :

$$\mathcal{Q} = \min_{(X,P) \in \{A,B,C,D\}} (\mathcal{Q}_P)_{\mathcal{M}_2(X)} . \tag{8.6}$$

Remark 8.6 *The quad quality computation in a Riemannian case (anisotropic metric) requires evaluating 16 quality values in the Euclidean case.*

A simplified quality function can be used. Let $ABCD$ be a quad, its quality can be defined as the value of an angle associated with the edge AC (common to the two triangles) as :

$$\mathcal{Q}_{\mathcal{M}_2(X)} = \min \left([\theta_A]_{\mathcal{M}_2(A)}, [\theta_A]_{\mathcal{M}_2(C)}, [\theta_C]_{\mathcal{M}_2(A)}, [\theta_C]_{\mathcal{M}_2(C)} \right) . \tag{8.7}$$

General scheme. The edges are sorted based on the decreasing quality value. The quads are formed by combining the triangles following this order. Once a quad is constructed, the corresponding edge is removed from the list of possible edges. A data structure such as a *heap* is used to store the triangles that remain to be combined. Indeed, at each quad creation, the list of the edges to be dealt with must be updated.

With this strategy, the quads that are constructed are, in some sense, the best possible. However, The number of isolated triangles is not minimal. Several ideas can be advocated so as to minimize this number of triangles or to avoid them altogether. On completion, there are still some triangles that have not been combined. These elements are then subdivided into three quads after introducing the edge midpoints and the centroids. This leads to propagating the refinement to some adjacent elements (cf. Figure 8.13). In such a case, the size of the elements in the final quad mesh is half the size of the triangles in the initial mesh (assumed to conform to a given size map). To solve this problem, one can use a technique based on the *twice size* (as previously discussed).

As the way in which the triangles to be combined are governed by the quality of the resulting quads, two particular drawbacks must be avoided :

- combining a triangle with some edges aligned with one of a neighbor,
- leaving too many isolated triangles. Indeed, the vertex introduced at the centroid of the element is of degree three and thus rigidifies the mesh thus making the quality optimization more delicate.

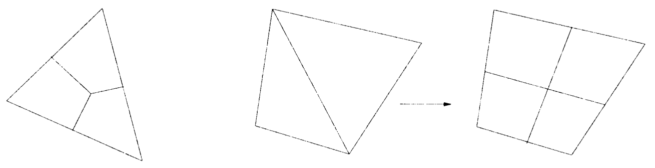

Figure 8.13: *Triangle combination so as to form quads. Left-hand side, an isolated triangle is subdivided into three quads. Right-hand side, the quad resulting from the combination of two triangles is split into four quads.*

Remark 8.7 *In the isotropic case, it is possible to use a different quality measure (see Chapter 18) :*

$$\mathcal{Q} = \alpha \frac{S_{min}}{h_{max}\sqrt{\sum\limits_{i=1,4} d_i^2}}, \tag{8.8}$$

where h_{max} is the largest length of the sides and the two diagonals, d_i is the distance between any two points in the quad (i.e., sides or diagonals), S_{min} is the minimum of the four (triangle) surfaces that can be associated with the quad and α is a normalization factor such that the square quality value is one.

Remark 8.8 *A procedure which is easy to implement, based on a frontal type propagation, can also be used to combine the triangles. Given a triangle, the triangles adjacent to an edge are examined and combined in this order (the list of triangles to be processed is no longer sorted according to a quality criterion, thus simplifying the algorithm).*

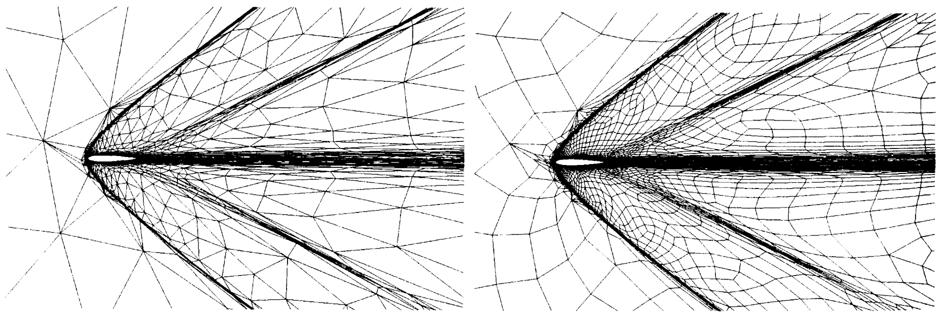

Figure 8.14: *Example of a quad mesh by triangle combination. Left-hand side, initial anisotropic triangular mesh. Right-hand side, resulting quad mesh. Note that the size of the quads is half that of the triangles.*

8.4.5 Dealing with a surface

Using a triangular surface mesh, it is also possible to obtain a quad mesh of this surface. The method is rather similar to that previously described.

In this approach, we assume a discrete parametric surface provided in the form $\sigma(u,v) = (x,y,z)$, the parameters u, v being defined in a rectangle. The metric $\mathcal{M}_2$ introduced in the general case corresponds here to the (anisotropic) metric of the principal radii of curvature or to the (isotropic) metric of the minimal radii of curvature, these metrics being defined in the parametric space. The definition of these different metric maps is described in Chapter 13.

8.5 Quad by means of a direct method

Unlike the previous schemes, various direct methods can be envisaged to obtain a quad mesh (without the temporary help of a triangular mesh). Three of these have received some attention :

- an advancing-front type method,
- a special understanding of the above grid superposition type method,
- a domain decomposition combined with an algebraic method in each sub-domain.

8.5.1 Advancing-front type method

This type of method consists in generating the domain *paving* by going through the domain starting from its boundaries (in this respect, we return to the advancing front type methods as seen in Chapter 6) [Blacker,Stephenson-1991].

More recently, [Cass et al.1996] advocated the use of this approach in the case of parametric surfaces while using the tangent planes together with the radii of curvature.

Another method in this class makes use of the STC, the *spatial twist continuum*, [Murdoch,Benzley-1995]. Given a quad mesh, the STC can be constructed. This structure is made of chords which links the quads. It can be seen as the dual of the mesh (in some sense, this structure is what is the Voronoï diagram for a triangular mesh). Conversely, given a STC, say a series of chords inside the domain which follow some rules, it is possible to define a quad mesh.

Remark 8.9 *In essence, this approach is sensitive to the domain (surface) boundary discretization. In addition, the number of segments in the boundary mesh must be even.*

8.5.2 Grid Superposition

Here, one follows an idea close to that seen in the section devoted to mesh construction by means of a grid (a set of quad or square boxes) or by using predefined patterns (quads in this context). Figures 8.6 and 8.7 give one academic example of the aspect of the meshes completed by using such an approach.

8.5.3 Using the medial axis

In Chapter 9, we will see the precise definition of the medial axis of a domain in two dimensions together with a method suitable for such a construction (actually, a discrete approximation of this line). Briefly, we assume the data of a discretization of the domain boundaries and an approximation of its medial axis. Then, this information can be used to subdivide the domain into regions with a simple geometry. These regions are constructed in such a way that certain properties hold, ensuring it is possible to cover them by means of quads. Depending on the case, the mesh is completed by using an algebraic type method or by following a midpoint subdivision method.

Domain partitioning. The domain's boundaries and its medial axis make it possible to define a domain partition composed of a certain number of regions that are bounded by a portion of this line, one of several portions of the boundary and one or several "cuts" joining these two categories of sides.

Mid-point subdivision. This technique makes it possible to cover a polygon by means of quadrilateral regions. An internal point is constructed, for instance the centroid G of the initial polygon then the edge midpoints are introduced (for the edge boundary of the polygon). It is then sufficient to join point G with two consecutive midpoints to obtain the quad regions that are sought.

The regions resulting from the domain partition are dealt with in this way and a first quad covering-up is thus obtained.

Quad construction. A repeated use of the above algorithm results in the final mesh completion. Another technique uses an algebraic method (as in Chapter 4) in each region while ensuring the conformity of the output mesh (we meet again the consistency constraints of the multiblock methods about the way in which the subdivision parameters propagate from one region to another).

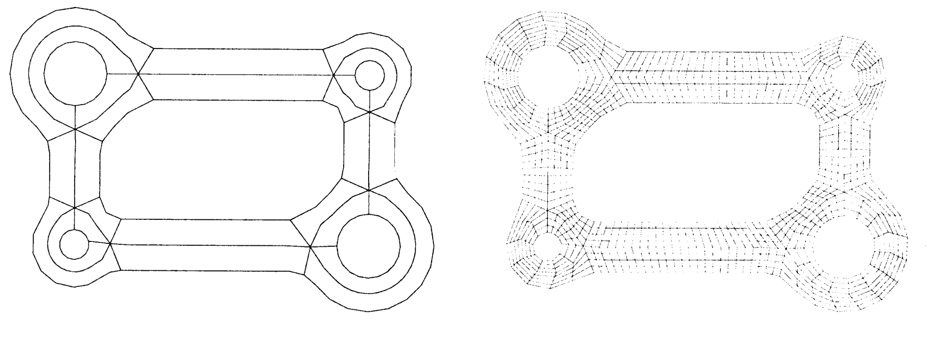

Figure 8.15: *Partitioning method based on the medial axis : approximate skeleton of the domain and cut lines (left-hand side), quad mesh resulting from this partition (right-hand side).*

8.6 Hex meshing

We consider here only the case of an arbitrarily shaped domain. Indeed, if the domain geometry is adequate, we already know some methods capable of constructing a hex mesh (algebraic method, P.D.E.'s based method, product method, etc.). First, it could be observed that the triangle combination method for quad meshing in two dimensions does not extend to three dimensions as it is tedious to define a tet combination resulting in a hex that uses all or, at least, the majority of the elements in the mesh.

On the other hand, splitting the elements of a given tetrahedral mesh by means of 4 hexes results in poorly shaped elements and bad connectivities (or vertex valence, see Chapter 18).

Thus, a direct construction method must be considered and, in principle, we return to the three methods that have been introduced for quad meshing purposes in two dimensions, the advancing-front method, grid based method or use of medial surface :

- advancing front based method. We meet here the principles of any classical advancing front method (Chapter 6). Nevertheless, the expected difficulties are now more critical. Indeed, it is tedious, given a face, to find the various points candidate for hex creation. Several techniques have been proposed, [Blacker,Meyers-1993], [Murdoch,Benzley-1995] and [Folwell,Mitchell-1998], which, in some cases, introduce some non hexahedral elements (pyramids, prisms, tets) and thus lead to *mixed* meshes;

- grid based method. The chosen grid could be uniform or hierarchical (an octree, or even an octree whose cells are subdivided into 27 octants (and not 8 as in the classical case)). The main difficulty then consists in dealing properly with the octants that intersect the boundary or that are close to it. The main references in this area are [Schneiders *et al.* 1996] and [Schneiders-1996a];

- method using the midsurface. A midsurface (approximated or having the same topology as the exact midsurface) is constructed (Chapter 9). It then serves to partition the domain in terms of regions whose topology is simpler so that a direct method can be envisaged, [Price *et al.* 1995], [Price,Armstrong-1997].

8.7 Miscellaneous

In this section, we mention some methods which can be suitable in certain situations and which have not been covered in the previous discussion.

8.7.1 Radial meshes

A radial mesh can be a source of benefit for some numerical simulations that can account with this specificity. Radial meshes can be developed based on various methods. For instance, a product type method is a natural candidate for such a result. An alternative method is the pattern based mesh generation method as briefly discussed above. Using a classical method (such as an advancing front or a Delaunay based method) with pre-placed vertices may also be a solution.

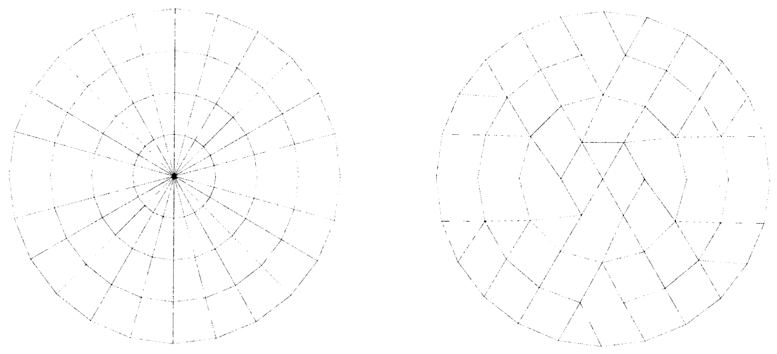

Figure 8.16: *Classical radial mesh as completed by a product method (left-hand side) and non classical radial mesh as resulting from a series of vertex and edge collapsing (right-hand side).*

In Figure 8.16 (left), one can see the classical result obtained by a product type method where the data consists of a discretized segment and a generation line that reduces to a rotation. One should note that the mesh elements near the center of the domain are very badly shaped due to the rigidity of the method which imposes a constant number of subdivisions along each section. Thus, in Figure 8.16 (right), we can see the mesh obtained using a variation developed by [Hecht,Saltel-1989] which consists in balancing the element sizes (note that some triangles remain in the resulting mesh). In short, the classical product mesh is modified by means of edge and vertex collapsing (see Chapter 18) so as to prevent the construction of elements that are too small. In this way, the number of subdivisions may vary from one section to another, thereby reducing, to some extent, the rigidity of the classical approach.

8.7.2 Recursive partition

The key-idea is to make use of the "divide and conquer" paradigm (Chapter 2) to construct the mesh of a domain using its boundary discretization as input data.

Given a domain by means of its boundary discretization (a series of polygonal segments in two dimensions, or a triangular surface mesh in three dimensions), a mesh generation method can be designed with is based on a repeated partition into two sub-domains of the current domain. Once a polygonal (polyhedral) sub-domain is composed of only three (or a small number) of segments or of four (or a small number) of triangular faces, it is then possible to mesh it by means of one (or a small number) of triangles (tets).

Let us restrict ourselves to a two dimensional problem. We consider the polygonal line serving as boundary discretization. A boundary vertex is selected and

a line passing through this point is constructed in such a way as to separate the domain into two sub-domains. This line is furthermore subdivided by means of segments and thus, two polygonal contours are defined. Thus, the initial meshing problem is replaced by two similar sub-problems. When a polygonal contour reduces to three (four or a meshable pattern), the subdivision is no longer pursued.

From a practical point of view, several issues must be carefully addressed including :

- the choice of a candidate vertex, the basis of the separating line,
- this line by itself and its appropriate subdivision so as to reflect the initial boundary stepsizes,
- the determination of meshable patterns.

Finding a separatrice line leads to selecting two points on the boundary such that the line whose extremities are these points is fully inside the domain with such a boundary. Therefore, we find an intersection problem that is, on the one hand, very sensitive to numerical errors (round-off errors) and, on the other hand, one which could be time consuming. Then, provided with such a line, it is generally necessary to subdivide it into segments so as to reflect the discretization of the initial boundary. A technique, close to that used when splitting the edges when creating the internal points in a Delaunay type method (Chapter 7), is then a possible solution. A sizing value is associated with the line endpoints, the length of this line is computed and, based on the desired point distribution, the line is subdivided into several adequately sized segments. Notice, in passing, that the separatrice lines will be present in the final mesh (as they stand or slightly modified if a point relocation step is performed with a view to optimization).

Remark 8.10 *The data of a size map (i.e., a control space) serves as input information that makes the above subdivision possible in accordance with some size specifications.*

A few remarks about this type of approach may be given. The choice of the separators strongly affects the resulting solution and a strategy must be defined to ensure a nice solution. In this respect, priorities in the selection of the basic points of the partition can be introduced (in a way, we return to some ideas close to those used in an advancing front type method, using the local aspects (such as angles), partition balancing, etc.).

In three dimensions, the same principle applies, at least formally speaking. However, the numerical problems are relatively much greater, in particular those related to intersection problems (the separatrice surface must remain inside the domain). Also, preventing the construction of ill-shaped elements (flat tets) and the definition of not meshable regions (such as the Schönhardt polyhedron) must be carefully considered.

In conclusion, this type of method, widely used in some commercial software packages, is a solution that enables automatic mesh construction for complex domains. Nevertheless, various numerical troubles may be expected and, in addition, the complexity (in terms of CPU cost) is far from easy to evaluate *a priori*.

Other methods ?

Do methods other than those covered in the previous chapters and more briefly in the present chapter exist ? The answer is not so clear. It is likely that solutions do exist for some particular configurations. And the answer would certainly be yes again if we consider some applications related to a "domain" other than the finite element domain (computer graphics, physical analogy, etc.), while observing that some of these methods developed in a specific context (spring analogy, molecular dynamics, simulated annealing, genetic algorithms, etc.) can be seen as occurrences of some previously known methods (although possibly under a different name) or as variations of some "classical" methods. In conclusion, it is necessary to remain attentive and open-minded in such judgments while retaining a critical view about any *a priori* new or novel method.

Chapter 9

Medial axis, mid-surface and applications

Introduction

Mid-surface or *medial surface* (*medial axis* in two dimensions) construction for a given domain is a very promising field of research that has multiple applications. For instance, given a two dimensional domain, the medial axis can be used for domain simplification, spatial dimension reduction and also as a first step to design an algorithm for quad mesh generation. Similarly the medial surface or mid-surface associated with a three dimensional domain can serve the same purposes. This is why we believe it worthwhile devoting a chapter to this class of methods.

Several methods can be advocated to construct the medial entity of a domain, also referred to as the domain *skeleton* or *neutral fiber*. A brief survey of the possible methods that have been investigated can be found in [Turkiyyah *et al.* 1997]. In this respect, quadtree-octree based methods, tracing algorithms, adequate P.D.E. solutions as well as Voronoï based methods can be considered for medial axis (mid-surface) extraction. Among these methods, we focus here on the last approach. Since this method considers the Voronoï cells of a given domain, it can be based on Delaunay triangulation of the domain. This opens up the discussion as to what is termed a *Delaunay admissible* discretization (in two dimensions) or a *Delaunay admissible* simplicial surface (in three dimensions) when the domain is a polygon or a polyhedron.

Based on such a boundary discretization, it is possible to complete an "empty" Delaunay triangulation[1] of the domain of interest. From that, algorithms can be developed resulting in the construction of the corresponding medial entity (actually, a discrete approximation of the entity).

Thus before considering medial entity construction and the various applications that can be derived from it, the first sections focus on Delaunay admissible boundary discretization.

[1]This triangulation is termed empty in the sense that it does not include any vertex inside the domain. The sole element vertices are those serving at the boundary discretization.

★
★ ★

The case of Delaunay admissible edges in a two dimensional space is considered, then we turn to the same edge problem in three dimensions before going on to the case of (triangular) faces. Then we discuss the construction of the desired medial entity and, finally, we briefly introduce various applications based on this entity.

9.1 Delaunay-admissible set of segments in $\mathbb{R}^2$

Let $\mathcal{S}$ be a set of points[2] in $\mathbb{R}^2$ and let $\mathcal{F}$ be a set of segments whose endpoints are members of $\mathcal{S}$. The question is to make sure that applying a Delaunay triangulation algorithm (Chapter 7) to $\mathcal{S}$ results in a mesh where the segments of $\mathcal{F}$ exist as element edges. In other words, given any segment f in $\mathcal{F}$ whose endpoints are A and B, we wish to have f a triangle edge at the time the points in $\mathcal{S}$ (in particular A and B) have been inserted by the Delaunay triangulation method.

Remark 9.1 *Actually, f will be formed if there exists a point P such that a Delaunay triangle fP exists. Another naive equivalent condition is that there is no pair of points, P and Q, located each on one "side" of f such that edge PQ will be formed that intersects f.*

If $\mathcal{F}$ is such that the previous property holds for all of its members, then $\mathcal{F}$ is said to be *Delaunay-admissible* (or *Delaunay-conforming*) and, for the time being, we just retain this naive definition for this notion.

Now, given $\mathcal{F}$ a set of segments, it is not guaranteed that $\mathcal{F}$ is Delaunay admissible. The problem is then to characterize this property, to find the condition(s) the segments in $\mathcal{F}$ must have to meet it and, in the case where $\mathcal{F}$ is not Delaunay admissible, to modify $\mathcal{F}$ in such a a way as to obtain a new set of segments which conforms to the desired property[3].

9.1.1 Terminology and notations

Before going further, we introduce some notations (see also Figure 9.1 and Table 9.1). Given $\mathcal{F}$, a set of segments, $f = AB$ denotes a segment of this set, $\mathcal{S}$ stands for the set of the endpoints of the f's. $\mathcal{L}_f$ denotes the line supporting f, this line defines two subspaces, $\mathcal{H}_f^+$ and $\mathcal{H}_f^-$. Given f, B_f^{min} is the open ball whose diameter is f (this ball corresponds to the smallest circle that can be constructed passing through the endpoints of f). Finally, given a triangle, say a triple ABC or a triple like fM where f is a segment (an edge) and M is a point, B_{ABC} (resp. B_{fM}) is the open ball circumscribing the triangle ABC (resp. fM).

[2]The points are not assumed to be in general position. Indeed, in what follows, a co-circular pattern, possibly after a swap, does not lead to difficulty.

[3]A precise definition of the notion of a Delaunay admissibility of an set of k–faces can be found in the *ad-hoc* literature and, for instance, in [George,Borouchaki-1997]. Such a theoretical definition involves the Voronoï cells, dual of the Delaunay triangulation. Actually, using such a duality is not strictly required in two-dimensions if ones wants to prove some theoretical issues (see below). Nevertheless, this argument will be reviewed later since it leads to a rather elegant proof when the dual problem (Voronoï) is easier to solve than the primal problem (Delaunay).

Notation	*Meaning*
f *or* AB	*segment in question*
$\mathcal{P}roj_f(P)$	*projection of point* P *on* f
$\mathcal{L}_f$	*line support of* f
$\mathcal{H}_f^+$, $\mathcal{H}_f^-$	*two half-planes related to* $\mathcal{L}_f$
B_f^{min}, B_{ABC}, B_{fP}	*small ball of* f, *circle circumscribing* $ABC(fP)$

Table 9.1: Notations for the entities involved in the edge problem in two dimensions.

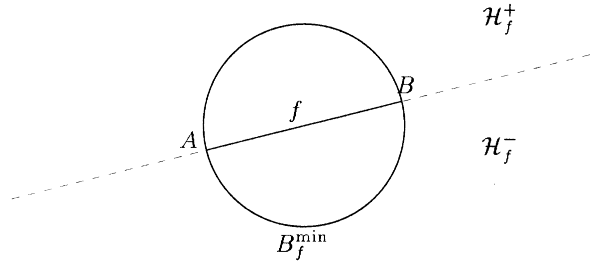

Figure 9.1: *A segment $f = AB$, member of $\mathcal{F}$, the corresponding ball B_f^{min} and the two associated half-spaces $\mathcal{H}_f^+$ and $\mathcal{H}_f^-$.*

9.1.2 Classification

Given a set $\mathcal{F}$ (and the corresponding set $\mathcal{S}$ where, to make sense, we assume that more than three non-aligned points exist), we need to analyze the configurations associated with all any f in $\mathcal{F}$ and, depending on the case, to see whether or not $\mathcal{F}$ is suitable or must be modified. This analysis concerns the various configurations that can be encountered. Let us consider an edge f, then the following cases can be found [Pebay-1998b] :

Case 0 : $\mathcal{H}_f^+$ (or $\mathcal{H}_f^-$) is empty,

Case 1 : B_f^{min} is empty,

Case 2 : B_f^{min} contains one or several points (other than A and B, remember that we are considering an open set).

9.1.3 Analysis of the three configurations

Obviously, a segment f falling within *(Case 0)* will be formed and then retrieved as an element edge. So it is for a segment f corresponding to *(Case 1)* as proved in what follows. Let A and B be the endpoints of segment f. Then, a condition to have AB an element edge is that there exists a point P in $\mathcal{S}$ such that $B_{fP} = \emptyset$. Since such a point exists, then a segment in *(Case 1)* is Delaunay.

Proof. Let P be an arbitrary point in $\mathcal{S}$, for instance in $\mathcal{H}_f^-$. Since $P \notin B_f^{min}$, the circumball B_{fP} is such that $B_{fP} \cap \mathcal{H}_f^+ \subset B_f^{min}$ then $B_{fP} \cap \mathcal{H}_f^+ = \emptyset$. If $B_{fP} \cap \mathcal{H}_f^- = \emptyset$, then $B_{fP} = \emptyset$ and AB is Delaunay. Otherwise, at least one point, say Q, exists in $B_{fP} \cap \mathcal{H}_f^-$. Replace P by Q and repeat the same process (*i.e.*, return to "since"). At completion we have found a point P such that $B_{fP} = \emptyset$ and thus AB is Delaunay admissible. Actually, the solution point is that for which the angle $\widehat{APB}$ is maximal. □

Note that, given f, the condition "B_f^{min} is empty" is a *sufficient condition* of Delaunay admissibility which, in fact, is too demanding.

Now, we turn to *(Case 2.)*. In this case, one or more points are located in B_f^{min}. Let P be the point such that angle $\widehat{PAB}$ is maximal. Then,

- if B_{fP} is empty, triangle fP is Delaunay and thus will be formed (and f will be formed),
- otherwise one or several points exist in B_{fP} and triangle fP is not Delaunay meaning that f is not Delaunay.

We then have a condition of Delaunay admissibility for f. It involves the point in set $\mathcal{S}$ with maximal angle and the simple examination of the Delaunay criterion for the corresponding triangle makes it possible to decide whether or not f is Delaunay.

Delaunay-admissibility and Voronoï cells. The Delaunay admissibility of a given segment in $\mathbb{R}^2$ can be also easily expressed in terms of the properties the corresponding Voronoï cells must have.

Thus, following Chapter 7 and more precisely Relationship (7.1), the Voronoï cell associated with a point A is the region V_A such that :

$$V_A = \{M \quad \text{such that} \quad d(M,A) \leq d(M,N), \quad \forall N \in \mathcal{S}\} \tag{9.1}$$

where $d(.,.)$ is the usual distance between two points. Now, consider four points in $\mathcal{S}$, the two endpoints of segment $f = AB$ under consideration and two other arbitrary (non-aligned with the previous) points, P and Q that can impede the construction of AB. Four Voronoï cells can be defined, successively V_A, V_B, V_P and V_Q. It is obvious that these four regions fall within one of the three situations depicted in Figure 9.2.

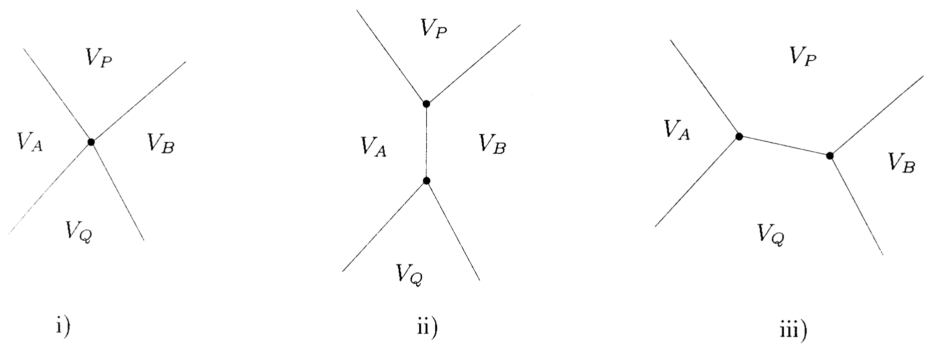

Figure 9.2: *The three possible patterns that can be encountered when considering the Voronoï cells related to four (non-aligned) points in $\mathbb{R}^2$.*

Now, $f = AB$ is Delaunay-admissible if edge AB exists in the Delaunay triangulation. In terms of Voronoï cells, AB is a Delaunay edge if V_A and V_B share a (Voronoï) edge or a (Voronoï) vertex[4]. Thus, AB will be formed in case i), in case ii) and not in case iii) as displayed in Figure 9.2. Thus we have a condition about the Voronoï cells that allows us to see whether or not a given segment is Delaunay.

Exercise 9.1 *Write the fact that V_A and V_B share an entity in terms of properties about the balls involved in the context. On the fly, retrieve the above conditions by showing that*

$$V_A \cap V_B \cap AB \neq \emptyset \iff B_f^{min} \ \textit{empty},$$
$$V_A \cap V_B \neq \emptyset \ \textit{and} \ V_A \cap V_B \cap AB = \emptyset \iff B_{fP} \ \textit{empty},$$

where P is such that angle $\widehat{APB}$ is maximal.

9.1.4 Construction of a Delaunay-admissible set of edges

Given $\mathcal{F}$ a set of segments in $\mathbb{R}^2$ and the corresponding set $\mathcal{S}$, we want to see whether of not this set is Delaunay and, if not, how to modify the unsuitable segments so as to form a Delaunay set of edges. The segments in $\mathcal{F}$ are examined. Only those falling within *(Case 2)* are considered. Let f be such a segment.

We first examine the case where only one segment exists in $\mathcal{F}$ and where only one point, P, falls within B_f^{min}, for instance in $\mathcal{H}_f^-$, Figure 9.3 :

- if $B_{fP} \cap \mathcal{H}_f^+$ is empty, triangle fP is Delaunay and thus will be formed (and f will be formed),
- otherwise one or several points exist in B_{fP} not in B_f^{min} and f is not formed[5]. But, a point M exists along $f = AB$ such that $B_{f_1}^{min}$ and $B_{f_2}^{min}$, where

[4] In such a case, the four points under consideration for the local analysis are co-circular and AB can be easily obtained (at least after a swap, Chapter 18).

[5] It can be easily proved that there is no point Q, other than the points falling within B_{fP}, such that the triangle fQ is Delaunay.

$f_1 = AM$ and $f_2 = MB$, are empty and we return to *(Case 1)* for these two new segments. Thus removing f from $\mathcal{F}$ and replacing it by f_1 and f_2 is a suitable solution.

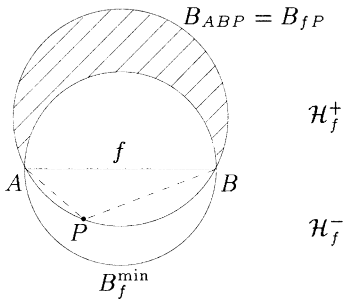

Figure 9.3: *In this pattern, B_f^{min} contains only one point, P, then assuming that $P \in \mathcal{H}_f^-$, the circumball C_{fP} may comprise one or several points in the region of $\mathcal{H}_f^+$ which is shaded in the figure.*

Proof. To prove this result, we just have to define the above point M. One solution is to define M as the projection of point P on f, the point denoted by $\mathcal{P}roj_f(P)$ in Figure 9.4 (left). Then after replacing f by the above f_1 and f_2, we analyze these two new situations. Obviously, $B_{f_1}^{min}$ and $B_{f_2}^{min}$ are empty. This is due to the fact that relationships

$$B_{f_1}^{min} \subset B_f^{min} \quad \text{and} \quad B_{f_2}^{min} \subset B_f^{min} \tag{9.2}$$

hold, combined with the fact that the possible points of $\mathcal{H}_f^+$ that can interact are outside B_f^{min} (see Figure 9.3) and thus outside $B_{f_1}^{min}$ and $B_{f_2}^{min}$ (see Figure 9.4, left-hand side). □

Remark 9.2 *Relationship (9.2) is the basic key to the problem.*

Remark 9.3 *Following Figure 9.4 right, one could observe that using as point M, the midpoint of AB may lead to a solution only after several subdivisions.*

To pursue, we have to examine the case where we have only one f and several points in B_f^{min}. The following scheme

- Pick the point P in B_f^{min} such that $\widehat{APB}$ is maximal,
- If B_{fP} is empty, f is Delaunay, END.
- Otherwise, apply the above construction. Consider the new segments f_1 and f_2 and repeat the process.

completes the desired solution.

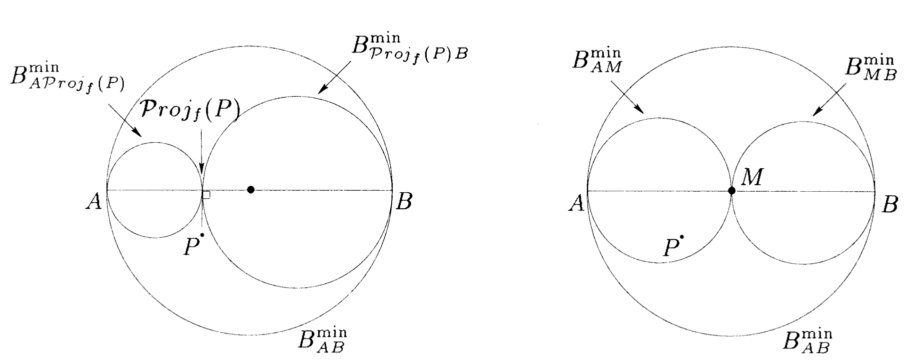

Figure 9.4: *Left-hand side : P in B_f^{min} no longer belongs to $B_{A\mathcal{P}roj_f(P)}^{min}$ and $B_{\mathcal{P}roj_f(P)B}^{min}$. Right-hand side : P in B_f^{min} falls within B_{AM}^{min} where M is the midpoint of AB.*

Proof. Actually, Relationship (9.2) gives the key. The radii of the relevant minimal balls decrease and then the number of points in these balls decreases. As the number of points is finite, the solution is yielded. □

Remark 9.4 *Note that the above construction is based on the projection of the point with maximal angle while another point impeding the construction could be used. This means that the resulting solution is not necessarily optimal (in specific, it is tedious to decide whether or not a given f has been split too much).*

Now, by means of exercises, we consider the case where the set $\mathcal{F}$ includes more than one segment.

Exercise 9.2 *Discuss the validity of the above proof in the case of two segments AB and AC sharing point A. Hint : first, observe that if the angle $\widehat{BAC}$ is obtuse, segments AB and AC can be decoupled in some sense. Otherwise if $\widehat{BAC}$ is acute, the two segments are coupled. Adding a point in AB may lead to adding a point for AC and conversely. Nevertheless, among the points impeding the construction of AB and AC, one can be retained that further decouples the resulting situation.*

Exercise 9.3 *In the same configuration, find a counterexample where subdividing edge AB using the midpoint results in a solution for edge AB which is no longer valid when considering edge AC. Hint : assume the angle between the two segments to be acute and show that adding a midpoint may lead to repeating to infinity the same pathology.*

Exercise 9.4 *Turn now to a non manifold boundary. Consider the case of a point A where several edges meet and where every angle between two segments is acute. Hint : examine the vertices surrounding point A. Find the smallest distance between these points and A. Introduce a point on all edges emanating from A at a distance less than the above value.*

Remark 9.5 *As previously mentioned, we have some flexibility when a projection is demanded. In the above construction, a specific point has been chosen and then projected but other projections may lead to the desired property. Thus a more subtle analysis can be made, in specific, if some criteria must be followed. For instance, it could be of interest to ensure that the resulting f_1 and f_2 are not too different in terms of length.*

Exercise 9.5 *Look at the case where the open balls are replaced by the closed balls. Note that these cases may result in some small changes in the previous discussion.*

9.1.5 Delaunay-admissible boundary discretization

The previous material can be used for a slightly different problem. Given a polygonal discretization of a closed curve Γ boundary of a given domain, we want to see whether or not this discretization is Delaunay conforming. If not, we want to modify it so as to complete such a Delaunay conforming discretization.

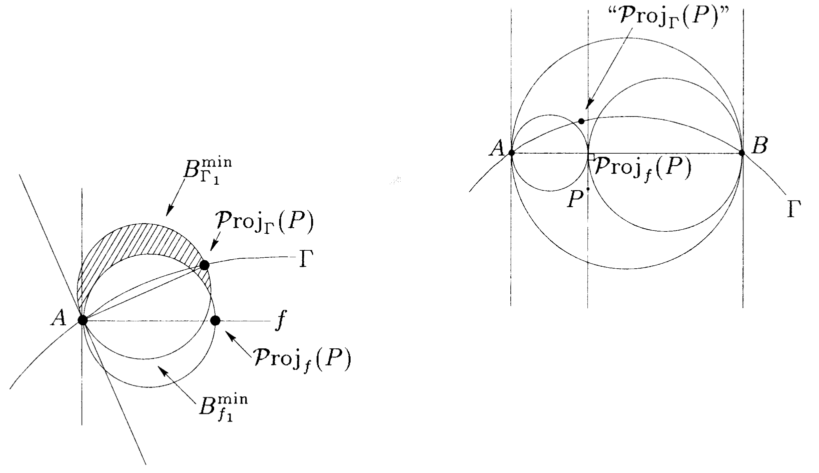
$B_{\Gamma_1}^{\min}$
$\mathcal{P}\mathrm{roj}_\Gamma(P)$
Γ
A
f
$\mathcal{P}\mathrm{roj}_f(P)$
$B_{f_1}^{\min}$
"$\mathcal{P}\mathrm{roj}_\Gamma(P)$"
A
B
$\mathcal{P}\mathrm{roj}_f(P)$
P
Γ

Figure 9.5: *Right-hand side : the context when a point P must be projected, the initial edge f and the curve Γ. The small ball of f, those of f_1 and f_2 resulting from the "theoretical" method. Left-hand side : the small ball that is not strictly included in the initial one when P is projected on Γ and not on f.*

Then, the above approach can be retained and, in specific, the construction of the projection of a point that impedes the Delaunay process can be used. Nevertheless, in contrast to the above theoretical framework, such a point must be precisely located on Γ and not on an edge f of Γ which leads to a different location when Γ is a curved boundary. In other words, if $f \in \Gamma$, then f_1 and f_2 as defined above are not necessarily close to Γ and we wish to avoid such a case.

Now, following Figure 9.5, Relationship (9.2) may be not satisfied. Thus, the convergence of the method is an issue. Actually, based on the strict decreasing of the radii of the small balls that are involved, the same result as above can be obtained.

9.2 Delaunay-admissible set of segments in $\mathbb{R}^3$

Following [George,Borouchaki-1997], dealing with the non Delaunay admissible segments included in a set of three-dimensional segments leads to the same result as in two dimensions and the same sufficient condition holds. Such a segment can be split, if necessary, using the adequate projections, as previously detailed in two dimensions. Similar reasons ensure the convergence. Indeed, the spheres involved in the construction are strictly enclosed in the former spheres they replace.

Let $f = AB$ be a segment in $\mathbb{R}^3$. Segment AB is Delaunay if the Voronoï cells V_A and V_B share a Voronoï entity (vertex, edge or facet). Thus, B_{AB} empty is a *sufficient condition* for Delaunay admissibility.

Proof. Since B_{AB} is empty, the midpoint of AB is in $V_A \cap V_B$, thus we have $V_A \cap V_B \neq \emptyset$. □

Remark 9.6 *The solution may be obtained after a swap (in the case where $V_A \cap V_B$ reduces to a single point).*

Now, assume that B_{AB} is not empty. Then one or several points exist in B_{AB}. One of these points, say P_1, is such that angle $\widehat{AP_1B}$ is maximal. Now, left as an exercise,

Exercise 9.6 *Show that for any point P in B_{AB} such that $\widehat{APB} < \widehat{AP_1B}$, we have $P_1 \in B_{APB}$ where B_{APB} is the ball whose great circle is the circumcircle of triangle APB.*

Exercise 9.7 *Show that if face ABP_1 is not created, then AB cannot exist.*

As a consequence, the above point P_1 is a natural candidate for forming a face with AB. Then, B_{ABP_1} empty is a condition for AB to be Delaunay.

Proof. Since B_{ABP_1} is empty, the center of the circle passing through A, B and P_1 is in $V_A \cap V_B \cap V_{P_1}$ then $V_A \cap V_B \cap V_{P_1} \neq \emptyset$. □

The above result can be obtained by considering the triangulation point of view. The segment AB is Delaunay admissible if, obviously, there exists one Delaunay tet (a series of Delaunay tets) with AB as an edge (to make sense, set $\mathcal{S}$ must include more than four points, including A and B, and then the number of tets is at least two).

Let P_1 be the above point (that in B_{AB} with $\widehat{AP_1B}$ maximal). B_{ABP_1} empty is a condition that ensures the existence of a Delaunay tet with face ABP_1 and thus with edge AB. Such a tet exists.

Proof. Let r_{init} be the radius of the disc ABP_1. Then, we consider a family of balls passing through A, B and the above P_1 whose radius varies from r_{init} to infinity. For a value of this radius, the ball touches a point, let P_2 be this point. Then, tet P_1ABP_2 is Delaunay since, by construction, its ball is empty. □

Exercise 9.8 *After element P_1ABP_2, find the other elements sharing AB. Show that they are Delaunay. Hint : use the same method to find the desired points based on the disc of the face common to the previous element. Then show that each resulting tet is Delaunay.*

Another less demanding condition is now introduced. Still based on ball ABP_1 which, now, is not assumed to be empty. Among the points in this ball, we select one, say P_2, such that the solid angle between P_2 and triangle ABP_1 is maximal, then if tet P_1ABP_2 is Delaunay, AB will be formed. In this way, we have a condition based on two specific points.

Remark 9.7 *Above tet P_1ABP_2 is the tet of maximal (circum)ball among all the tets with a vertex in B_{ABP_1}.*

Now, to complete the discussion, we have to follow the same scheme as in the above section starting from a simple situation and going further to the general case (and, in specific, a case where several edges emanate from one point).

9.3 Delaunay-admissible set of triangular faces

We now face the same problem for $\mathcal{F}$, a given set of triangular faces, where one face is denoted as f. The main difficulty that is expected is related to the fact that Relationship (9.2) does not extend to a face in three dimensions. Given a triangle f in $\mathbb{R}^3$ along with B_f^{min} its minimum ball having as *great circle*, gc_f, the circle circumscribing f, then any subdivision of f results into sub-triangles and, Figure 9.6, in general, we don't have

$$B_{f_1}^{min} \subset B_f^{min}, \qquad (9.3)$$

where f_1 denotes one of these sub-triangles. Actually, introducing a subdivision point inside the initial face f or along one of its edges leads to great circles that are not included in the initial great circle. Moreover, in the case of a surface mesh, splitting a face may lead to splitting the neighboring faces so as to preserve the conformity of the mesh.

Despite these observations, we want to find a solution[6]. First, we establish the catalogue of the possible patterns. Then we propose a method while discussing the convergence issues.

[6]As in Section 9.1, a naive approach to the problem is to make sure that a Delaunay tetrahedron exists with f as a face. Note that the cases where a face f does not exist do not reduce to a situation where an edge PQ, where P and Q are two points one on each of the half-spaces separated by a plane supporting f, intersects f.

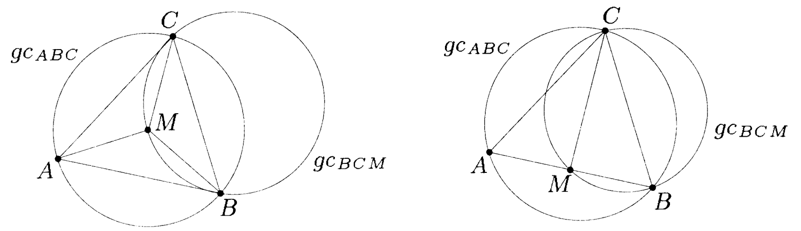

Figure 9.6: *The great circles, $gc_{...}$, (thus the corresponding balls) after subdivision are not strictly included in the initial circles (the initial ball). Moreover, some radii can be significantly greater than the initial one. Left : the face $f = ABC$ is subdivided based on an internal point M. Right : the subdivision point M is located along one edge of f.*

9.3.1 Notations

As for the two-dimensional case, we define some notations (see also Figure 9.7 and Table 9.2). $\mathcal{F}$ is now a set of triangular faces, $f = ABC$ denotes such a face and $\mathcal{S}$ stands for a set of points including the extremities of the f's. Π_f is the plane supporting f. It defines two subspaces, $\mathcal{H}_f^+$ and $\mathcal{H}_f^-$. Given f, B_f^{min} is the open ball whose great circle, gc_f, is the circumcircle of f (this ball corresponds to the smallest sphere that can be constructed passing through the vertices of f). Finally, given a tetrahedron, say a quadruple $ABCD$ or a quadruple like fM where f is a face (a triple) and M is a point, B_{ABCD} (resp. B_{fM}) is the open ball circumscribing the tetrahedron $ABCD$ (resp. fM).

Notation	*Meaning*
f *or* ABC	*the face in question*
Π_f	*the plane supporting* f
$\mathcal{H}_f^+$, $\mathcal{H}_f^-$	*the two sub − spaces related to* Π_f
B_f^{min}	*the small ball of* f
B_{AB}^{min}	*the small ball of segment* AB
B_{-C}^{min}	*the small ball of segment* AB *(opposite* C*)*
gc_f	*the great circle of* f $(gc_f = B_f^{min} \cap \Pi_f)$
B_{ABCD} , B_{fP}	*the circumball of* $ABCD$ *(resp.* fP *)*

Table 9.2: Notations for the entities involved in the face problem.

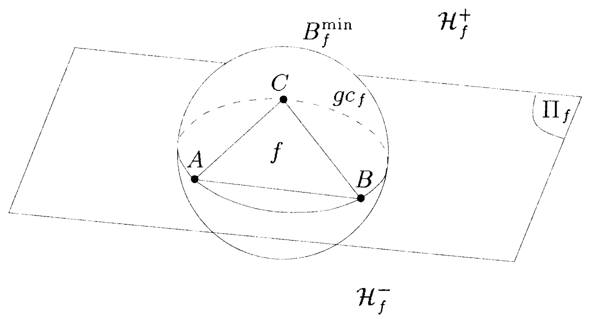

Figure 9.7: *A face $f = ABC$, member of $\mathcal{F}$, the corresponding ball B_f^{min}, the great circle (disc) gc_f, the plane Π_f and the two associated half-spaces $\mathcal{H}_f^+$ and $\mathcal{H}_f^-$.*

9.3.2 Classification

***A priori* classification.** Given a set $\mathcal{F}$ (and the set $\mathcal{S}$), we want to see whether or not $\mathcal{F}$ exists in $\mathcal{D}el(S)$ the Delaunay triangulation of $\mathcal{S}$ (to make sense we assume that more than four points compose $\mathcal{S}$). If not, we construct a new set $\mathcal{F}'$ (and the related set $\mathcal{S}'$) such that $\mathcal{D}el(S')$ has the desired property (*i.e.*, the faces in $\mathcal{F}'$ exist in the triangulation based on $\mathcal{S}'$).

Thus we need to analyze the configurations associated with any f in $\mathcal{F}$. For a given triangular face f, the following cases can be encountered :

(Case 0) : $\mathcal{H}_f^+$ (or $\mathcal{H}_f^-$) is empty, more precisely one find *(Case 0.1)* if gc_f is empty or *(Case 0.2)*, otherwise,

(Case 1) : B_f^{min} is empty

(Case 2) : B_f^{min} is not empty, which leads to several situations : *(Case 2.1)* if gc_f is empty or *(Case 2.2)* otherwise.

Reduced classification. A more precise analysis of the above classification leads us to remove *(Case 0.2)* as well as *(Case 2.2)*. In other words it is always possible to ensure that the gc_f's are empty.

Assume a configuration where gc_f is not empty. Let P be a point in gc_f (thus in Π_f), it is immediate to see that this point will impede the construction whatever the context (with respect to B_f^{min}, $\mathcal{H}_f^+$ or $\mathcal{H}_f^-$). Indeed, the circumball of any tet with f as a face will include point P and thus such a tet is not Delaunay. Actually we face a problem in two dimensions (in plane Π_f). Given an arbitrary triangulation in a plane (the plane Π_f here), it is possible to achieve a Delaunay triangulation by means of edge swapping. As a consequence the cases where a gc_f is not empty can be removed after edge swapping, thus resulting in a new set $\mathcal{F}$ with empty gc_f's. Thus, the classification reduces to :

- *(Case 0)* : $\mathcal{H}_f^+$ (or $\mathcal{H}_f^-$) is empty,
- *(Case 1)* : B_f^{min} is empty,
- *(Case 2)* : B_f^{min} is not empty,

while, at the same time, the gc_f's are empty. Based on this classification, we want to see if $\mathcal{F}$ is suitable or must be modified so as to obtain what is needed.

9.3.3 Analysis of the three configurations

We examine the three above situations, see [Pebay-1998a] for more details. A face in *(Case 0)* is Delaunay.

Proof. Assume that $\mathcal{H}_f^+$ is empty, then the points candidate for connection with f are in $\mathcal{H}_f^-$. Let Q be the point in $\mathcal{H}_f^-$ such that the solid angle formed with f is maximal, then the ball of tet $ABCQ$ is empty. □

Since gc_f is empty, the sole points that can impede the construction would be exactly located on the boundary of gc_f (and thus co-circular with A, B and C). In this case, f is easy to obtain, possibly after a swap.

Actually, above point Q could be obtained based on what follows :

- pick a point, say P, in $\mathcal{H}_f^-$. Consider the tet $ABCP$ and its ball B_{ABCP}.
  - if B_{ABCP} is empty, then $Q = P$.
  - otherwise, a point M exists in B_{ABCP}, set $P = M$ and repeat the same process.

The solution results from the fact that

$$\text{if} \quad M \in B_{ABCP} \cap \mathcal{H}_f^- \quad \text{then} \quad B_{ABCM} \cap \mathcal{H}_f^- \subset B_{ABCP} \cap \mathcal{H}_f^- . \tag{9.4}$$

Now, a face in *(Case 1)* is also Delaunay.

Proof. First, if P is a point in $\mathcal{H}_f^+$, not in B_f^{min}, for tet $ABCP$ we have $B_{ABCP} \cap \mathcal{H}_f^- \subset B_f^{min}$ thus $B_{ABCP} \cap \mathcal{H}_f^-$ is empty. If P is in $\mathcal{H}_f^-$, we have a similar conclusion, $B_{ABCP} \cap \mathcal{H}_f^+$ is empty.

Now, consider a point P in $\mathcal{H}_f^+$, not in B_f^{min}, form the tet $ABCP$. If $B_{ABCP} \cap \mathcal{H}_f^+$ is empty, set $Q = P$. Otherwise, a point M exists in $B_{ABCP} \cap \mathcal{H}_f^+$, set $P = M$ and repeat the process. At completion, we have found a point Q such that $B_{ABCQ} \cap \mathcal{H}_f^+$ is empty.

Since $B_{ABCQ} \cap \mathcal{H}_f^- \subset B_f^{min}$, then B_{ABCQ} empty holds. □

It may be noted that we again meet the conditions of Delaunay admissibility of a given face which are similar to those of an edge in $\mathbb{R}^2$. Now we have to examine the remaining case. Let us recall that in *(Case 2)*, for a given face f, we have a situation where :

$$B_f^{min} \neq \emptyset \ , \mathcal{H}_f^- \neq \emptyset \ , \mathcal{H}_f^+ \neq \emptyset \text{ and } gc_f = \emptyset \, , \tag{9.5}$$

in other words, B_f^{min} encloses one or several points, some in $\mathcal{H}_f^+$, some in $\mathcal{H}_f^-$. Let P be the point in B_f^{min} also in $\mathcal{H}_f^+$ such that B_{ABCP} is maximal, then $B_{ABCP} \cap \mathcal{H}_f^+$ is empty. Thus, a condition for f to be Delaunay is that $B_{ABCP} \cap \mathcal{H}_f^- = \emptyset$. Since the same holds when considering the points in $\mathcal{H}_f^-$, the condition is the same for the point in one or the other half-spaces which maximizes the corresponding circumball.

To summarize, unless the above condition holds (which is actually tedious to check), the faces in *(Case 2)* are not Delaunay.

9.3.4 Construction of a Delaunay-admissible set of faces

Based on the previous classification, some faces are Delaunay admissible whereas others require specific processing. As previously seen, only faces falling within *(Case 2)* must be examined. We have a situation where B_f^{min} encloses at least one point, say P, for instance in $\mathcal{H}_f^-$, and the ball circumscribing the quadruple fP encloses at least one point, say Q, in $\mathcal{H}_f^+$ (if the above circumball is empty, the face f is Delaunay) while the great circle related to f is empty.

The only case where this particular situation is not the situation encountered would be the following :

- a point P is in B_f^{min} as above for instance in $\mathcal{H}_f^-$ and, at the same time, one point (or more) exists in the boundary of gc_f and a Delaunay tet could be formed based on P and this (these) point(s). In this case, a swap may be required to retrieve f.

For the sake of simplicity, we discard this particular case and we first discuss the case where only two such points exist (P and Q). Before going further, bear in mind that any subdivision by given patterns of an unsuitable face may result in replacing this face by several Delaunay admissible faces but at the same time, due to conformity reasons, may lead to splitting some neighboring faces (if we consider a surface mesh) for which the corresponding small spheres may be as large as we want.

Towards a general scheme. After the above remark, the solution cannot be directly completed by "subdividing" the faces. Therefore, the expected result can be obtained only by using a multiple step scheme where, at each step, some *criterion* is enhanced. The scheme we propose is as follows :

- (A) we process the non Delaunay face edges,
- doing so, we introduce some new faces as well as some new edges,

- as long as, among these new edges, there are some non Delaunay edges, we return to (A),
- then we process the faces (and, now, all face edges are Delaunay-admissible).

Edge enforcement. The edges of the given faces that are not Delaunay admissible are dealt with using the previous method. In this way, they are subdivided, if judged useful, and an admissible partitioning is completed.

Nevertheless, when subdividing an edge, for instance into two sub-edges, a new edge is introduced. In fact, the problem doesn't reduce to considering stand alone edges in $\mathbb{R}^3$ but edges that are face edges. Thus, it is necessary to maintain a topological face structure.

At completion, we have a set of faces all of whose edges exist, and we then turn to the face enforcement step.

Face enforcement. Following Footnote 6 above, we examine more precisely a configuration where the given face is missing due to two points P and Q, one in $\mathcal{H}_f^-$, the other in $\mathcal{H}_f^+$ (this point may be also in B_f^{min} or not), such that edge PQ is Delaunay admissible and impedes the construction of face f. Some occurrences of such a pattern are depicted in Figure 9.8.

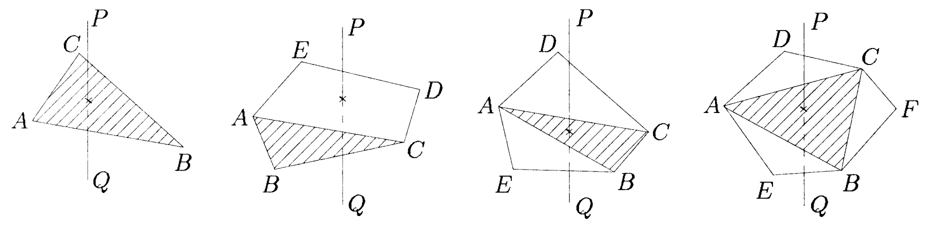

Figure 9.8: *A face $f = ABC$, member of $\mathcal{F}$, is not Delaunay due to an edge PQ. Several tets sharing PQ exist whose vertices, apart from P and Q, could be those shown in the figure.*

If edges AB, AC and BC exist, we are faced with a situation where PQ intersects ABC, then only tets $PQAB$, $PQAC$ and $PQBC$ exist as can be easily proved (left-hand side of the figure). Thus, the corresponding circumballs are empty. Then, observe that the only way to prevent the creation of the three above tets is to violate the Delaunay criterion. In other words, a solution is to introduce a point in the intersection of these circumballs.

We first consider the case where P, in B_f^{min} is such that the sphere passing through P and f is larger than that passing through Q and f. Let M be the projection of P onto the face f. *A priori*, based on the position of P with regard to f, M is either internal to f or is located on one of the edges of f. Since PQ cuts f, it is easy to see that M is necessarily inside f and falls within the three above balls, therefore this particular point will prevent the creation of edge PQ.

Using the so-defined point M, one subdivides the face f by means of three sub-faces. The new edges are processed so as to obtain Delaunay-admissible edges (via the above method or using some edge swap in the plane of f) and we consider the set of resulting faces.

At completion, three results hold. First, point P is no longer inside any balls associated with the sub-faces replacing face f. Next, the radii of the balls related to the edges and those related to the newly created faces decrease. Thus, point P has been discarded from the balls of the new faces. Indeed, point M acts as a "wall" between P and Q which are no longer coupled together (*i.e.*, P is not closer to Q than any other point in $\mathcal{S}$).

This ensures the convergence of the process.

We now turn to a case where there are several points in B_f^{min}, such that several edges intersect the face f. Among these points, we pick the one forming the maximal solid angle with f (the sphere passing through f and this point is maximal). We project this point onto f and we repeat the same process, then we consider again the B_f^{min} (the above point being discarded), we repeat the same process as long as a point exists inside one of these new balls.

Remark 9.8 *Notice that the only impeding edges are those which cut the face (thus, in Figure 9.8, three cases correspond to this situation while the fourth case is not of interest).*

Remarks about the convergence. We postulate that the global convergence holds. For the face edges, this results from the previous discussion. Regarding the faces, the same result holds if we can make sure that the process has no loop. This proof remains to be properly established, as far as we know (unless some particular assumptions are assumed about, for instance, the value of the angles between two edges).

From a practical point of view, one can find in [Pebay-1998a], an algorithm based on the previous issues which is completed by some heuristics. The key-idea is to enforce the edges prior to considering the resulting faces. Furthermore, to each of these faces, the following process is applied :

- a subdivision based on the a choice about the three possible cases (one point being added on one of the three edges) such that the retained pattern maximizes the minimum angle between the resulting triangles,
- an edge swap (in the case of coplanar faces, this being exactly or approximatively verified).

This heuristic strategy has proved to be adequate in numerous significant examples (see the above reference).

9.3.5 Delaunay-admissible boundary discretization

We face the same problem as in two dimensions. Given a discretization of a closed surface Σ boundary of a given domain, we want to see whether or not

this discretization is Delaunay conforming. If not, we want to modify it so as to complete such a discretization.

Then, the above approach can be employed and, specifically, any point, edge or face creation required to ensure the desired property must be such that this entity is precisely located on Σ and not on a face f of Σ.

9.4 Medial axis

First, we recall the definition of the medial axis of a domain. Given a domain Ω in $\mathbb{R}^2$, the medial axis of Ω is defined as follows :

Definition 9.1 *The medial axis of a domain is the locus of the centers of the circles of maximum radius that can be inscribed in the domain.*

9.4.1 Delaunay triangulation and medial axis construction

Following on from the above, it is possible to find the *medial axis* or the *skeleton* of a given *polygonal* domain based on its boundary discretization. From the above definition, the medial axis of a domain is the locus of the center of a circle of maximal diameter as it rolls inside this domain.

Given a Delaunay admissible discretization of a domain boundary, it is easy to obtain a Delaunay triangulation of this domain whose sole vertices are the endpoints of the edges of the above boundary discretization. In terms of the Voronoï diagram corresponding to the above triangulation, we encounter two types of Voronoï entities, the *Voronoï edges* and the *Voronoï vertices.* A Voronoï edge is equidistant from two Delaunay vertices while a Voronoï vertex is equidistant from three such vertices[7]. The Voronoï edges dual to a boundary edge are removed and the resulting edges form a polygonal line (which can reduce to a point). Under appropriate assumptions, this line enjoys some nice properties and thus gives a rough idea of what the medial axis is. Note that among the Voronoï edges that are removed, we encounter first the *non finite* Voronoï edges[8] (also referred to as unbounded edges) as well as some finite edges related to concave parts of the boundary.

A theoretical issue can be written as the following theorem :

Theorem 9.1 *If h tends to 0, where h is the length of the longest edge of the discretization of the boundary of Ω, then the union of the finite internal Voronoï edges associated with the triangulation of Ω based on its boundary vertices approaches the medial axis of Ω.*

In other words, the medial axis is basically obtained once h tends towards 0, by joining the centers of the circumcircles of the triangles in the Delaunay mesh.

First, we give a sketch of the proof. Then we discuss the convergence issue of the proposed method (*i.e.*, the finite internal Voronoï edges).

[7] or more in the case of co-circular points.
[8] Indeed, such edges intersect a boundary edge.

Proof. Since h tends to 0, the boundary discretization is Delaunay admissible. Then the triangulation of Ω based on the boundary vertices is Delaunay. As a consequence, the Voronoï edges are easy to obtain based on the dual of the current triangulation. Now, among these edges, those which are not related to a boundary Delaunay edge are inside the domain and the circles centered in these edges are inside the domain. Moreover the above circles have maximal radii. Thus, the so-selected Voronoï edges form the medial axis of Ω.

The first three points are obvious. Given an arbitrary boundary discretization, it is possible to modify it so as to complete a Delaunay admissible discretization. Moreover, such a discretization can be uniform which will further simplify matters. Let h_{min} be the smallest distance between two non consecutive boundary vertices, then $h \leq \frac{h_{min}}{2}$ is a suitable value for the definition of the distance between two consecutive boundary points. Indeed, a uniform mesh of the boundary can be constructed with boundary edges of length h. This mesh satisfies the condition of Delaunay requirement. As a consequence, a Delaunay triangulation of the domain can be easily constructed whose vertices are the above uniformly spaced boundary points.

The dual of this triangulation can be constructed. The edges of the dual that are non related to a Delaunay boundary edge are inside the domain. The latter property results from an appropriate choice of h as can be easily seen.

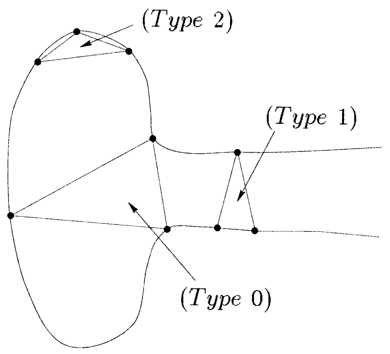

Figure 9.9: *The three types of triangles that can be encountered in an "empty" simplicial mesh in two dimensions.*

It still remains to prove that, when h tends to 0, the circles centered on these edges are inside the domain and are maximal. From a computational point of view, three types of Delaunay triangles exist based on the number of boundary entities they have. A Delaunay triangle may have (see Figure 9.9) :

- no boundary edge, termed a *(Type 0)* triangle,
- one boundary edge for a *(Type 1)* triangle
- two boundary edges[9] and the triangle is termed a *(Type 2)* triangle.

[9]The case where the three edges of a triangle are boundary edges is a specific case where the domain approximation reduces to this single triangle. Thus a (Type 3) triangle, corresponding to this case, is not really interesting.

Accordingly, the medial axis can be found by considering the way the different triangles contribute to it. First, we consider the case of a *(Type 0)* triangle before discussing a pair of two adjacent triangles.

- *(Type 0)* triangle.

Let Γ be the boundary of domain Ω and let ABC be the triangle in question. The circumcircle of this triangle touches the boundary at three vertices. It is maximal. If h is small enough, the circle is internal to Ω as it satisfies the Delaunay criterion. At A, B or C, it is tangent to Γ if this boundary is at least of class C^1. Otherwise, we have a simple contact, the contact point being a corner (Γ is only of class C^0 at this point). Therefore, the circumcenter (which is inside the triangle) belongs to the medial axis. This point is a *special node* or a *critical node* of the axis in which three Voronoï edges meet. Based on the configuration related to the three adjacent triangles (see below), such a node is the beginning endpoint of three branches of the axis (which can be reduced to this point in some cases, for instance in the case of co-circular points).

To know the regularity of the contact between this circle and the boundary, it is necessary to analyze the edges of Γ in one or other side of this contact vertex. If Γ is at least of class C^1, these two edges tend with h towards the (unique) tangent and the contact is of class C^1. If Γ is of class C^0, the two edges form an angle (other than Π) and we have a simple contact, with no tangency property.

Note, in addition, that whatever h, a *(Type 0)* triangle does not vanish (its surface area does generally not tend to 0 with h).

Now, to obtain the entire medial axis, apart from the critical nodes, we have to examine the contribution of all pairs of adjacent triangles. First, we discuss the case where one of the two triangles is of *(Type 0)*, thus leading *a priori* to three possibilities. Then we turn to the case where one triangle is of *(Type 1)* resulting in two new cases (while a pair *(Type 2)-(Type 2)* doesn't make sense).

- Case of a *(Type 0)-(Type 0)* pair.

A combination *(Type 0)-(Type 0)* is depicted in Figure 9.10. We denote by O_1 the center of the circumcircle of triangle ABC and by O_2 that of circle of ADB, AB being the common edge. If the four vertices are not co-circular, the centers O_1 and O_2 are distinct. The segment O_1O_2 is a part of the medial axis. Indeed, the circle centered at O_1, as well as that centered at O_2 obviously conforms to the definition and, moreover, all circle centered on O_1O_2 and passing through the endpoints A and B of the common edge is maximal, inside the domain and touches this domain at two points. The point O_1 is, potentially, the origin of three branches of the axis (see below) among which is the branch O_1O_2, so it is for the point O_2. The case where the four vertices are co-circular implies that $O_1 = O_2$ and this critical node is, potentially, the origin of four branches of the medial axis as will be seen shortly.

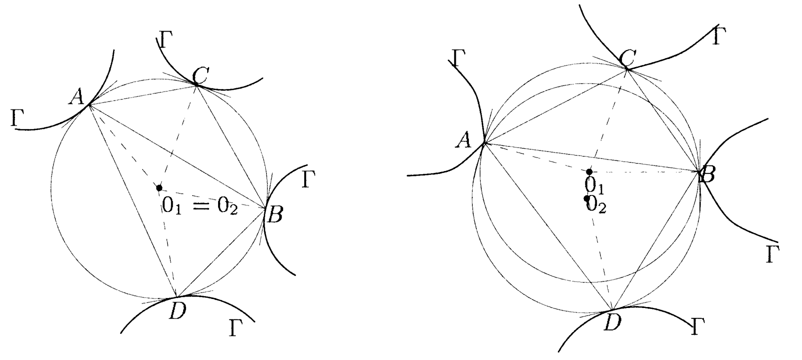

Figure 9.10: *The contribution of a (Type 0)-(Type 0) pair. Left : the four vertices are co-circular. Right : the four vertices are not co-circular.*

- Case of a *(Type 0)-(Type 1)* pair.

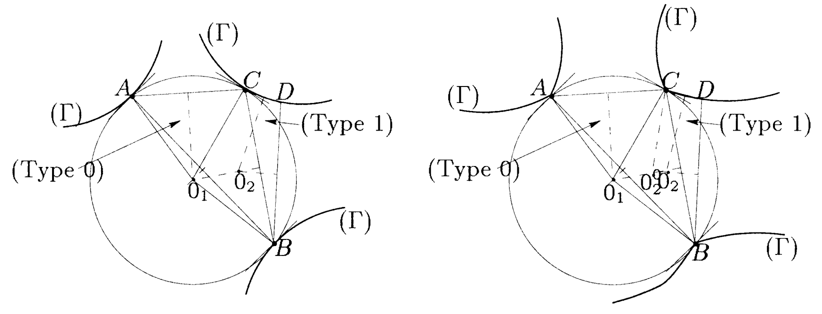

Figure 9.11: *The contribution of a (Type 0)-(Type 1) pair. Left : the case of a C^1 boundary at point C, O_2 tends towards O_1 with h. Right : the case of a C^0 boundary at point C, O_2 tends towards O_2^0 with h and, in general, O_2^0 is distinct from O_1.*

Figure 9.11 illustrates a *(Type 0)-(Type 1)* combination. Let ABC be the triangle of *(Type 0)* and BDC be the triangle of *(Type 1)* which is adjacent to the previous through the edge CB. Edge CD is a boundary Delaunay edge whose size is h. As above, we notice O_1 and O_2 the centers of the circles circumscribing the two triangles in question. If CD tends with h towards the tangent at C at Γ then O_2 tends towards O_1. Otherwise, O_2 tends towards a point O_2^0 distinct from O_1. If h is small enough (non null), in the first case, any point belonging to O_1O_2 is a point of the medial axis and thus O_1O_2 is a portion of this axis. In the second case, it is the same but the medial axis will have a discontinuity reflecting the corner in the boundary at C (see the remark below). Notice again that the various points O involved in the construction are or are not inside the corresponding triangles. The contribution to the medial axis of a pair of triangles is not necessarily located inside these triangles. To end, if D is co-circular with A, B and C, then the contribution to the axis is reduced to the center-point of the circle of ABC.

Remark 9.9 *The boundary regularity induces the medial axis smoothness. A point like O_2^0 results in a local regularity of type C^0 in the axis. Otherwise, this line is locally at least of class C^1.*

- Case of a *(Type 0)-(Type 2)* pair.

A *(Type 0)-(Type 2)* junction is *a priori* possible. Nevertheless, if h is small enough, this pattern does not exist. Indeed, when h vanishes, a *(Type 2)* triangle reduces to one point. Thus, this case is only feasible from a discrete point of view and does not participate to the theoretical discussion. Note that for numerical reasons, this case can arise for a sufficiently small (but non null) h.

- Case of a *(Type 1)-(Type 1)* pair.

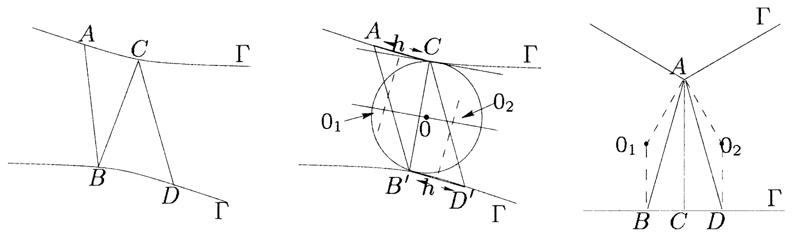

Figure 9.12: *The contribution of a (Type 1)-(Type 1) pair. Left and middle : first possible situation. Right : second case where a beam is encountered.*

In the following, Figure 9.12, *(Type 1)-(Type 1)* junction is discussed. Indeed, two such junctions can be found based on the geometry of the domain boundary. Let ABC and BDC be the two triangles sharing the Delaunay edge BC which traverses the domain. The first category of *(Type 1)-(Type 1)* junction is encountered when AC and BD are the Delaunay boundary edges while the second corresponds to the case where the Delaunay boundary edges are BC and CD meaning that point A is the base of a *beam* of edges traversing the domain from one boundary to the other.

Let us consider the first case, when h tends to 0, let us assume that C remains unchanged, thus A tends towards C. Then, the circles passing through C tangent to the boundary are centered on the "right" normal[10] at the boundary at vertex C. Among this family of circles, let us pick the one which is tangent to the opposite boundary, let O be its center. Let B' be the point where this property holds, then the retained circle is centered on the "right" normal at the boundary at this vertex B'. Let us examine[11] triangle $AB'C$ as well as triangle $B'D'C$. They are

[10]This notion of "right" normal is due to the fact that point C tends towards point A from the "right" side.

[11]Depending on the boundary curvature near points C and B', the Delaunay triangles that are formed are $AB'C$ and $B'D'C$ or $AB'D'$ and $B'D'C$. Nevertheless, the discussion is the same whatever the situation (while using the "left" normals).

Delaunay triangles. The circumcenters of these two triangles, O_1 and O_2, define the Voronoï edge O_1O_2 and O is located on this edge. Since h tends to 0, the three points O, O_1 and O_2 tend towards the same point, actually point O. Now, from a discrete point of view, if h is small enough, O_1O_2 is the approximate medial axis as contributed by the two triangles in question.

Now we turn to the second possible situation, namely a beam corresponding to one or two concave parts of the boundary. A close look at Figure 9.12 (right-hand side) leads to proving that the circumcenters of any triangles in the beam tend towards O_1 (resp. O_2) with h based on the position of the triangle under examination. Depending on the boundary regularity, these two points are distinct or not, which is also visible about the axis regularity (which may contain a portion of a parabola whose links with the adjacent segments give the resulting smoothness).

- Case of a *(Type 1)-(Type 2)* pair.

A *(Type 1)-(Type 2)* junction is now discussed. As for the *(Type 0)-(Type 2)* case, the *(Type 2)* triangle vanishes thus contributing with its terminal node and the corresponding Voronoï edge (which may not exist in the case of co-circularity). The medial axis ends at the center of the circle of the triangle of *(Type 2)*, to complete it, one can link this point with the terminal node.

To summarize, one defines the critical nodes (related to the *(Type 0)* triangles) then the branches between two such nodes. These branches result from the contribution between a *(Type 0)* triangle and its neighbor of *(Type 1)*, and then from those of the pairs *(Type 1)-(Type 1)* until a contribution of a pair *(Type 1)-(Type 0)* or *(Type 1)-(Type 2)*, in which case the medial axis ends at a terminal node (except in a case of co-circularity where the axis ends before this point). Thus the proof of Theorem 9.1. holds. □

After this discussion, let us mention a few remarks.

Remark 9.10 *From a practical view point, h is a small value (and thus does not tend towards* 0*) and the above discussion only gives a general idea about the expected result.*

Remark 9.11 *If one considers a polygonal approximation of a curved boundary, the circles circumscribing (Type 2) triangles tend towards the osculating circles of the boundary curve.*

Remark 9.12 *As previously seen, the topology of the medial axis is defined based on the critical and terminal nodes. The branches in the axis are defined between two such nodes.*

Remark 9.13 *One should define the medial axis as the locus, when h tends towards* 0*, of the circumcenters of the Delaunay triangles. This method is nice* a priori *but is nevertheless badly suited to the case where the boundary is not sufficiently smooth. Indeed, two neighboring centers do not necessarily converge toward a unique point, thus leading to a discontinuity (a hole) in the axis. Examples a such cases have been seen in the above discussion and can be found in [Turkiyyah* et al. *1997].*

9.4.2 Voronoï cells of a set of points and medial axis

The data input of points located on the domain boundaries which are dense enough, while not explicitly defining the corresponding boundary edges, make it possible to retrieve the same result. A Delaunay triangulation is built, based on the insertion of these points and the corresponding Voronoï cells are considered so as to find the medial axis of this cloud of points. If this cloud is dense enough, this axis is that of the domain that is implicitly defined by this type of input data.

9.4.3 Computational issues

In practice, three types of numerical difficulties generally arise. First, the case where there are co-circular points leads to an imprecise definition of the axis (which should be reduced to one point) because the Delaunay triangulation is not uniquely defined in such a situation. The second issue concerns the choice of the h's in such a way as to ensure a suitable regularity of the line completed by the construction. A third problem, which is immediate, is related to numerical errors. For instance, one case of such a problem leads to find *(Type 2)* triangles and thus branches in the axis while such triangles (branches) do not exist in theory. Moreover, the two following exercises demonstrate some interesting issues.

Exercise 9.9 *Find the relationship between the h's on the boundary discretization and the lengths of the portions of lines in the approximate medial axis.*

Exercise 9.10 *Find the relationship between the h's on the boundary discretization and the boundary curvature in order to guarantee a nice enough smoothness in the approximated medial axis.*

Notice that the aim is to minimize the size of the mesh by choosing variable h's, the simplest solution clearly being to consider a constant and small value for these h's.

Finally, a rough idea of the medial axis is given by the following.

Remark 9.14 *Joining the midpoints of the internal Delaunay edges allows a line to be constructed whose topology, when h is appropriately sized, is similar to that of the medial axis. Notice that defining as bifurcation point ((Type 0) triangle) the element centroid is an easy solution (since this point falls within the triangle) but, depending on the context, could be a rather coarse approximation of the exact result.*

9.5 Mid-surface

Constructing the mid-surface of a domain using a Delaunay triangulation based only on surface boundary points adequately distributed on this surface is more tedious. Before giving a few remarks about this approach, we recall the formal definition of what a mid-surface is (similar to Definition 9.1).

Definition 9.2 *The medial surface of a domain is the locus of the centers of the spheres of maximal radius that can be inscribed in the domain.*

Find the mid-surface of an empty Delaunay triangulation (with no internal point) of the domain appeals an immediate comment. If the domain boundary mesh is not Delaunay-admissible, there is no guarantee (Chapter 7) that such an empty triangulation exists (due to the Steiner points). Moreover, when h tends towards 0, a situation assumed in theory, this phenomenum is not relevant. Thus, we can discard this difficulty, for the moment.

We therefore assume the existence of the desired triangulation. Then, the equivalent of Theorem 9.1 (which we prefer not to attempt to formulate and prove here) is simply assumed. Note, without going into detail, that the proof is necessarily harder than in two dimensions. Specifically, if we want to use a classification of the tets, we clearly find many more types than above.

Exercise 9.11 *Classify the tets as a function of how many of their edges or faces belong to the domain boundary (while their four vertices are assumed to be on this boundary).*

This being done, it is shown that there are tets having only boundary edges (and no boundary faces) and the precise examination of their neighboring elements is necessary in order to find a possible contribution to the mid-surface. Let us recall that in two dimensions, the contributions to the medial axis corresponded to the dual of the triangulation, say some Voronoï edges. In three dimensions, the situation is much more complex. regarding the dual, we find some Voronoï faces and some Voronoï edges whose presence indicates the way in which the underlying tet touches the boundary.

A few papers discuss, in greater or lesser detail, these aspects and, among these, the reader is referred to [Yu *et al.* 1991] and [Armstrong *et al.* 1993].

9.6 Medial axis (mid-surface) transform

This is a process commonly called the "*Medial Axis Transform*" (M.A.T. for short), see for instance [Price *et al.* 1995], [Armstrong *et al.* 1995] or [Sheehy *et al.* 1996]. This transformation, used in a meshing context, has also long been used for graphic purposes [Blum-1967].

The key-idea is that the data of the medial axis together with some additional information (radii, branches, etc.) make it possible to retrieve the boundary of the domain in question.

9.7 Applications

Numerous applications take advantage of the medial axis (resp. mid-surface) of a domain in $\mathbb{R}^2$ (resp. $\mathbb{R}^3$). A first application consists in partitioning a domain into several sub-domains. Furthermore, this approach allows quad (hex) mesh generation methods to be applied in these sub-regions. On the other hand, the

analysis of the entities in the medial axis (mid-surface) provides some indications about the domain geometry which, in turn, allows for some different operations such as dimensional reduction (a domain in $\mathbb{R}^3$ is seen as a surface, a domain in $\mathbb{R}^2$ is seen as a line), or a simplification of geometry details where a detail is found to be based on too "small" an entity in the axis.

9.7.1 Domain partitioning

In two dimensions, using medial axis identification, it is possible to split the domain into geometrically simple regions. This subdivision is based on the types of the triangles which indicate whether a branch exists or not. The existence of a branch (a triangle of *(Type 0)*) reflects the fact that several paths on the domain are possible starting from this branch. The types also indicate the local concavity or convexity of the domain boundary.

Following these observations, it is possible to define some *cut lines* that separate the domain. A *(Type 2)* triangle corresponds to a convex region. A *(Type 1)* triangle traverses the domain from one boundary to another and the examination of the elements neighbor of this triangle indicates the geometrical nature of the domain boundary.

A first method. The vertex of a *(Type 2)* triangle common to its two boundary edges is a terminal node of the approximated axis and a line emanating from this point defines a cut. This line is followed, passing through some triangles with *(Type 1)* and *(Type 0)* until a *(Type 2)* triangle is found (by vicinity). In this way, we define a graph which allows for the definition of a partition of the domain by means of polygonal regions (assuming the data of a polygonal boundary).

Provided the domain has no hole and assuming the boundary edges between two vertices in a *(Type 2)* triangle are considered as one side, assuming also that the edges included between such a vertex and a node in a *(Type 0)* triangle and also those between two nodes with *(Type 0)* as one side, then the number of sides in the so-defined polygon is the number of the triangles of *(Type 0)* traversed in the path plus two. In this way, the resulting polygons have three, four, five or six sides[12].

Another method. The above method can be replaced by a method which relies more explicitly on *(Type 0)* triangles. One constructs the three lines joining the node in the triangle and its three vertices. Then, while traversing the axis, one joins this node with that of the next *(Type 0)* triangle. This results in a partition of the domain into a set of quads and/or triangles only (Figure 9.13).

Either approach provides a partition of the given domain. Notice that numerous variations can be used in order to optimize such a partition, in particular, by explicitly taking into account the concave points (or, at least, some of them based on the angles) so as to convexify the resulting regions as well as possible.

[12]This result can be found in the literature. Is it possible to find more sides ? Presumably not.

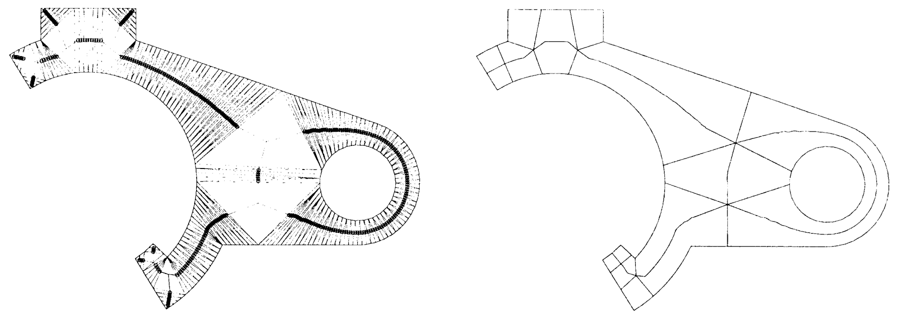

Figure 9.13: *Domain partitioning based on its medial axis, in two dimensions. Delaunay triangulation and medial axis approximation (left-hand side) and partition resulting from method 2 (right-hand side).*

Whatever the method, the resulting partition can be used with a view to quad construction.

The same partitioning principle can be used in three dimensions. We use the faces in the mid-surface and their type (*i.e.*, the type of the underlying tets) to find the cutting surfaces in the domain. However, it is clear that the actual implementation is much more complex and, have yet to be thoroughly mastered (the examples that can be seen in the literature are, for the most part, rather academic).

The process begins by analyzing the corners, the edges and the faces in the mid-surface. Based on this classification, one considers the way in which a sphere of maximal radius centered in one entity of the mid-surface touches the domain boundary. In accordance with the situations that arise, one find whether it is possible or not to complete a particular (polyhedral) region. The goal is then obtained once the domain has been entirely subdivided by means of such regions.

In conclusion and with no further discussion about this problem, we think that some nice research issues are likely to be found and various new programming developments may be expected in this area.

9.7.2 Quad mesh construction

First, it is possible to use a partition in polygons (each having a small number of sides) and to deduce from it a (coarse) quad partition. If the polygon is not a quad itself, the midpoint subdivision method completes what is expected.

The latter technique consists in inserting one point in the region (for instance, its centroid) and one point in each of its sides (for instance the midpoint for a free side, *i.e.*, a side related to a portion of the medial axis and one point, close in some sense, for a boundary side). It is easy to see that only quads are obtained in this way. The number of quads is the number of sides (a triangle results in three quads, etc.). This subdivision is made globally and results in a first coarse conforming covering-up of the domain (Figure 9.14, middle).

The resulting covering-up serves as a basis for the construction of the final mesh. To complete this mesh, there are two approaches. The first consists in repeating the midpoint subdivision process. The second method uses a classical algebraic mesh generation method (Chapter 4).

Whichever approach is used, the global conformity of the mesh must be maintained. This leads to imposing consistency about the number of subdivision of a given coarse element by taking into account its neighboring elements and, therefore, this imposes some constraints that propagate from element to element.

Finally, given the subdivision parameters, there is no special reason for having the points of the given boundary follow this point distribution. Thus, new points are constructed in the boundary and it is these points that define the new geometry of the resulting mesh and thus that of the final discretization of the domain (Figure 9.14, right-hand side).

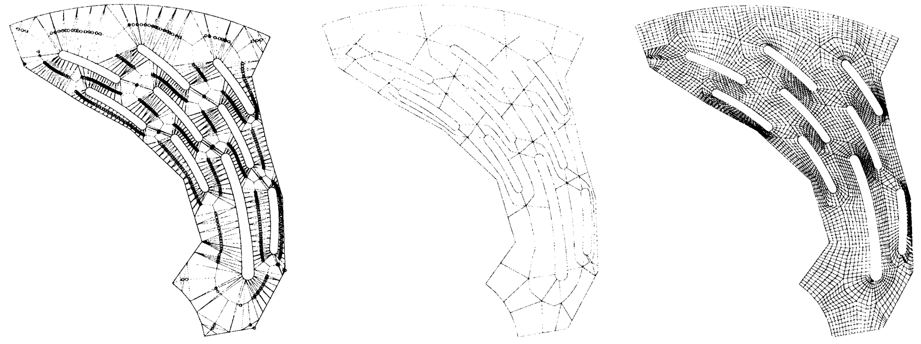

Figure 9.14: *Examples of quad meshes based on the medial axis, in two dimensions : Delaunay triangulation of the domain (left-hand side), domain partitioning using the approximated medial axis (middle) and domain mesh resulting from an algebraic method (right-hand side).*

9.7.3 Hex mesh construction

As can be imagined, this topic is in principle the extension to three dimensions of what we did in two dimensions. In practice, the expected difficulty is much greater. Simply notice that two mesh generation methods can be used to mesh a region resulting from the partition :

- as in two dimensions, a midpoint subdivision method (possibly repeated) or an algebraic type method (Chapter 4) or a similar method, [Price *et al.* 1995], [Price,Armstrong-1997],

- solely in three dimensions, an approach by extrusion (the product method as seen in Chapter 8) which consists in using a mesh of the considered region of the mid-surface and to extrude it in a third direction (towards the domain boundary),

while ensuring compatibility at the region interface level.

9.7.4 Other applications

Among the other applications using the medial axis or the mid-surface of a domain, one can envisage dimensional reduction [Donaghy *et al.* 1996] or domain simplification [Armstrong *et al.* 1995].

Dimensional reduction. In essence, this operation consists in replacing a computation in two dimensions by one in one dimension (in three dimensions by a computation based on a surface). Observe that the shell case is a case where the domain is a three dimensional domain but where the computation is made based on a surface (after some assumptions and using some data values that allow for the three dimensional aspect of the problem in question).

Dimensional reduction is mostly used in two situations :

- the problem, in its intrinsic dimension, can be approximated by a problem posed in a space with a lesser dimension (assuming some more or less restrictive hypotheses),
- the solution of the reduced problem provides an indication, albeit relatively coarse, about the solution of the exact problem and thus already makes it possible to know some important parameters for a reduced CPU cost.

Here again, the medial axis (in two dimensions) serves at a basis for the dimensional reduction procedure. However, the medial axis, by itself, is not usually the ideal solution. Indeed, the adequate solution can be formed by a combination of regions (entities) in zero dimension (some points), in one dimension (some edges) and, in some places, in two dimensions, all of them being adequately linked to as to permit the further usage of the reduced model. Notice also that a simplification process (see below) enables us, in some cases, to obtain a more complete reduction.

Geometry simplification. Simplifying the geometry of a domain is an operation that has proved useful in various contexts (and it will be discussed again later, in particular, in the surface mesh case, Chapter 15). For the moment, we use the properties of the medial entity in order to simplify the geometry. The key idea is to remove some details that are judged useless (in other words, too small in some sense) in the geometry while preserving the general shape and the topological structure of the geometric model.

Remark 9.15 *Detail removal must be made in accordance with the targeted application. Indeed, during the solving of the problem, a small detail may be the source of a singularity in the solution and thus removing it may alter the computed solution. Thus it is recommended to remove details whose influence remains local. For graphical purposes, the size of a detail is the only factor to be considered.*

The medial axis and the corresponding radii (in two dimensions) indicate the size of a given detail as compared with its neighborhood.

Visiting the medial axis, one detects the possible holes and the possible loops. Then, one evaluates the size of these holes (by observing the path visited in the axis by comparison with the average radius of a maximal circle traversing the same path). In this way, it is possible to decide whether a hole can be suppressed or not (edge collapsing, see Chapter 18, is then a suitable solution).

Then the edges are examined. With each edge is associated a value defined as the ratio between the edge length and the average radius of the maximal circle that touch it. Again, this value allows for the decision. In this way it is possible to simplify the geometry by suppressing the fillets, the small notches, the small protrusions or the small stepsizes.

In three dimensions, suppressing a face is much more a delicate operation and (see Chapter 19), such an operation will be made using a series of edge collapsing while (same chapter) maintaining topological coherence and smooth enough regularity for the thus-simplified surface.