International Conference on Adaptive Modeling and Simulation
ADMOS 2003
N.-E. Wiberg and P. Díez (Eds)

ADAPTIVE 3-D TRIANGULATIONS FOR ENVIRONMENTAL PROBLEMS

R. Montenegro, G. Montero, J.M. Escobar, E. Rodríguez and J.M. González-Yuste

University Institute of Intelligent Systems and Numerical Applications in Engineering
University of Las Palmas de Gran Canaria, Campus Universitario de Tafira
35017 Las Palmas de Gran Canaria, Spain
e-mail: rafa@dma.ulpgc.es

Key words: Mesh Generation, Mesh Smoothing, Mesh Untangling, Adaptive Refinement, Nested Meshes, 3-D Finite Element Method.

Abstract. *In the finite element simulation of environmental processes that occur in a three-dimensional domain defined over an irregular terrain, a mesh generator capable of adapting itself to the topographic characteristics and to the numerical solution is essential. The objective of this work is to present our recent results in these topics. We construct an unstructured tetrahedral mesh of a region bounded in its lower part by the terrain and in its upper part by a horizontal plane. The main ideas for the construction of this initial mesh combine the use of a refinement/derefinement algorithm for two-dimensional domains and a tetrahedral mesh generator based on Delaunay triangulation. The points density over the terrain increases with the complexity of the orography and the point generation in the domain is done attending to a vertical spacing function over different layers defined from the terrain to the upper part of the domain. To avoid conforming problems between mesh and orography, the mesh will be designed with the help of an auxiliary parallelepiped. Once the 3-D Delaunay triangulation of the set of points has been constructed on the parallelepiped, points are replaced on their real positions keeping the mesh topology. In this last stage there can be occasional low quality elements, or even inverted elements. For this reason, we have developed a simultaneous untangling and smoothing procedure to optimise the resulting mesh. Once we have constructed the adapted mesh in accordance with the geometrical characteristics of our domain, we use an adaptive local refinement of tetrahedral meshes in order to improve the numerical solution. Finally, all these techniques are applied to the generation of adapted meshes for realistic and test problems.*

1 INTRODUCTION

In Section 2 we construct a tetrahedral mesh that approximates the orography of the terrain with a given precision [1, 2]. To do so, we only have digital terrain information. Our domain is limited in its lower part by the terrain and in its upper part by a horizontal plane placed at a height at which the magnitudes under study may be considered steady. The lateral walls are formed by four vertical planes. The generated mesh could be used for numerical simulation of natural processes, such as wind field adjustment [3], fire propagation [4] and atmospheric pollution. These phenomena have the main effect on the proximities of the terrain surface. Thus node density increases in these areas accordingly.

To construct the Delaunay triangulation [5] we must define a set of points within the domain and on its boundary. These nodes will be precisely the vertices of the tetrahedra that comprise the mesh. Point generation on our domain will be done over several layers, real or fictitious, defined from the terrain up to the upper boundary. Specifically, we propose the construction of a regular triangulation of this upper boundary. Now, the refinement/derefinement algorithm [6, 7] is applied over this regular mesh to define an adaptive node distribution of the layer corresponding to the surface of the terrain. Once the node distribution is defined on the terrain and the upper boundary, we begin to distribute the nodes located between both layers. A vertical spacing function is involved in this process.

The node distribution in the domain will be the input to a three-dimensional mesh generator based on Delaunay triangulation [8]. To avoid conforming problems between mesh and orography, the tetrahedral mesh will be designed with the aid of an auxiliary parallelepiped. We start with the definition of the set of points in the real domain and its transformation to the auxiliary parallelepiped where the mesh is constructed. Next, the points are placed by the appropriate inverse transformation in their real position, keeping the mesh topology. This process may give rise to mesh tangling that will have to be solved subsequently. We should, then, apply a mesh optimisation to improve the quality of the elements in the resulting mesh. In Section 3, we introduce a method [9], so the untangling and smoothing are performed in the same stage. To do this, we shall use a modification of usual objective functions [10, 11].

In Section 4, a local refinement algorithm [12] for tetrahedral meshes, based on the 8-subtetrahedron subdivision [13, 14, 15], is summarized. We have applied this technique to improve the numerical solution of several 3-D problems and, recently, we have combined it with the smoothing procedure to improve the quality of distorted elements. Obviously, for evolution problems it is necessary to implement a derefinement algorithm as the inverse of the refinement one.

To illustrate the effectiveness of our approaches, we present in Section 5 several applications where it can be seen the validity of the proposed strategies. Finally, conclusions are presented in Section 6.

2 AUTOMATIC MESH GENERATION ADAPTED TO GEOMETRY

2.1 Adaptive Discretization of the Terrain Surface

The three-dimensional mesh generation process starts by fixing the nodes placed on the terrain surface. Their distribution must be adapted to the orography to minimise the number of required nodes. First, we construct a sequence of nested meshes $T = \{\tau_1 < \tau_2 < ... < \tau_m\}$ from a regular triangulation τ_1 of the rectangular area under consideration. The τ_j level is obtained by previous level τ_{j-1} using the 4-T Rivara algorithm [16]. All triangles of the τ_{j-1} level are divided in four sub-triangles by introducing a new node in the centres of each edge and connecting the node introduced on the longest side with the opposite vertex and with the other two introduced nodes. Thus, new nodes, edges and elements named *proper* of level j appear in the τ_j level. The number of levels m of the sequence is determined by the degree of discretization of the terrain digitalisation. In other words, the diameter of the triangulation must be approximately the spatial step of the digitalisation. In this way we ensure that the mesh is capable of obtaining all the topographic information by an interpolation of the actual heights on the mesh nodes. Finally, a new sequence $T' = \{\tau_1 < \tau'_2 < ... < \tau'_{m'}\},\ m' \leq m$, is constructed by applying the derefinement algorithm; details may be seen in [6, 7]. In this step we present the derefinement parameter ε that fixes the precision with which we intend to approximate the terrain topography. The difference in absolute value between the resulting heights at any point of the mesh $\tau'_{m'}$ and its corresponding real height will be less than ε.

This resulting two-dimensional mesh $\tau'_{m'}$ may be modified when constructing Delaunay triangulation in the three-dimensional domain, as its node position is the only information we use. We are also interested in storing the level in which every node is proper so as to proceed to the node generation inside the domain. This will be used in the proposed vertical spacing strategies.

2.2 Vertical Spacing Function

As stated above, we are interested in generating a set of points with higher density in the area close to the terrain. Thus, every node is to be placed in accordance with the following function

$$z_i = a\, i^{\alpha} + b \tag{1}$$

so that when the exponent $\alpha \geq 1$ increases, it provides a greater concentration of points near the terrain surface. The z_i height corresponds to the ith inserted point, in such a way that for $i = 0$ the height of the terrain is obtained, and for $i = n$, the height of the last introduced point. This last height must coincide with the altitude h of the upper plane that bounds the domain. In these conditions the number of points defined over the vertical is $n + 1$ and (1) becomes

$$z_i = \frac{h - z_0}{n^{\alpha}}\, i^{\alpha} + z_0 \quad ; \quad i = 0, 1, 2, ..., n \tag{2}$$

It is sometimes appropriate to define the height of a point in terms of the previous one, thus avoiding the need for storing the value of z_0

$$z_i = z_{i-1} + \frac{h - z_{i-1}}{n^\alpha - (i-1)^\alpha} \left[i^\alpha - (i-1)^\alpha \right] \quad ; \quad i = 1, 2, ..., n \tag{3}$$

In (2) or (3), once the values of α and n are fixed, the points to insert are completely defined. Nevertheless, to maintain acceptable minimum quality of the generated mesh, the distance between the first inserted point ($i = 1$) and the surface of the terrain could be fixed. This will reduce to one, either α or n, the number of degrees of freedom. Consider the value of the distance d as a determined one, such that $d = z_1 - z_0$. Using (2),

$$d = z_1 - z_0 = \frac{h - z_0}{n^\alpha} \tag{4}$$

If we fix α and set free the value of n, from (4) we obtain

$$n = \left(\frac{h - z_0}{d} \right)^{1/\alpha} \tag{5}$$

Nevertheless, in practice, n will be approximated to the closest integer number. Conversely, if we fix the value of n and set α free, we get

$$\alpha = \frac{\log \frac{h - z_0}{d}}{\log n} \tag{6}$$

In both cases, given one of the parameters, the other may be calculated by expressions (5) or (6), respectively. In this way, the point distribution on the vertical respects the distance d between z_1 and z_0. Moreover, if the distance between the last two introduced points is fixed, that is, $D = z_n - z_{n-1}$, then the α and n parameters are perfectly defined. Let us assume that α is defined by (6). For $i = n - 1$, (2) could be expressed as

$$z_{n-1} = \frac{h - z_0}{n^\alpha} (n-1)^\alpha + z_0 \tag{7}$$

and thus, by using (6),

$$\frac{\log (n-1)}{\log n} = \frac{\log \frac{h - z_0 - D}{d}}{\log \frac{h - z_0}{d}} \tag{8}$$

From the characteristics which define the mesh, we may affirm *a priori* that $h - z_0 > D \geq d > 0$. Thus, the value of n will be bounded such that, $2 \leq n \leq \frac{h - z_0}{d}$, and the value of α cannot be less than 1. Moreover, to introduce at least one intermediate point between the terrain surface and the upper boundary of the domain, we must verify that $d + D \leq h - z_0$. If we call $k = \frac{\log \frac{h - z_0 - D}{d}}{\log \frac{h - z_0}{d}}$, it can be easily proved that $0 \leq k < 1$. So, (8) yields

$$n = 1 + n^k \tag{9}$$

If we name $g(x) = 1 + x^k$, it can be demonstrated that $g(x)$ is contractive in $\left[2, \frac{h-z_0}{d}\right]$ with Lipschitz constant $C = \frac{1}{2^{1-k}}$, and it is also bounded by

$$2 \leq g(x) \leq 1 + \left(\frac{h - z_0}{d}\right)^k \leq \frac{h - z_0}{d} \tag{10}$$

In view of the fixed point theorem, we can ensure that (9) has a unique solution which can be obtained numerically, for example, by the fixed point method, as this converges for any initial approximation chosen in the interval $\left[2, \frac{h-z_0}{d}\right]$. Nevertheless, the solution will not generally have integer values. Consequently, if its value is approximated to the closest integer number, the imposed condition with distance D will not exactly hold, but approximately.

2.3 Determination of the Set of Points

The point generation will be carried out in three stages. In the first, we define a regular two-dimensional mesh τ_1 for the upper boundary of the domain with the required density of points. Second, the mesh τ_1 will be globally refined and subsequently derefined to obtain a two-dimensional mesh $\tau'_{m'}$ capable of fitting itself to the topography of the terrain. This last mesh defines the appropriate node distribution over the terrain surface. Next, we generate the set of points distributed between the upper boundary and the terrain surface. In order to do this, some points will be placed over the vertical of each node P of the terrain mesh $\tau'_{m'}$, attending to the vertical spacing function and to level j $(1 \leq j \leq m')$ where P is proper. The vertical spacing function will be determined by the strategy used to define the following parameters: the topographic height z_0 of P; the altitude h of the upper boundary; the maximum possible number of points $n + 1$ in the vertical of P, including both P and the corresponding upper boundary point, if there is one; the degree of the spacing function α; the distance between the two first generated points $d = z_1 - z_0$; and the distance between the two last generated points $D = z_n - z_{n-1}$. Thus, the height of the ith point generated over the vertical of P is given by (2) for $i = 1, 2, ..., n - 1$.

Regardless of the defined vertical spacing function, we shall use level j where P is proper to determine the definitive number of points generated over the vertical of P excluding the terrain and the upper boundary. We shall discriminate among the following cases:

1. If $j = 1$, that is, if node P is proper of the initial mesh τ_1, nodes are generated from (2) for $i = 1, 2, ..., n - 1$.
2. If $2 \leq j \leq m' - 1$, we generate nodes for $i = 1, 2, ..., min(m' - j, n - 1)$.
3. If $j = m'$, that is, node P is proper of the finest level $\tau'_{m'}$, then any new node is generated.

This process has its justification, as mesh $\tau'_{m'}$ corresponds to the finest level of the sequence of nested meshes $T' = \{\tau_1 < \tau'_2 < ... < \tau'_{m'}\}$, obtained by the refinement/derefinement algorithm. Thus the number of introduced points decreases smoothly with altitude, and they are also efficiently distributed in order to build the three-dimensional mesh in the domain.

We set out a particular strategy where values of α and n are automatically determined for every point P of $\tau'_{m'}$, according to the size of the elements closest to the terrain and to the upper boundary of the domain. First, the value of d for each point P is established as the average of the side lengths of the triangles that share P in the mesh $\tau'_{m'}$. A unique value of D is then fixed according to the desired distance between the last point that would be theoretically generated over the different verticals and the upper boundary. This distance is directly determined according to the size of the elements of the regular mesh τ_1. Once d and D are obtained, for every point P of $\tau'_{m'}$, their corresponding value of n is calculated by solving (9). Finally, the vertical spacing function is determined when obtaining the value of α by (6). This strategy approximately respects both the required distances between the terrain surface and the first layer and the imposed distance between the last virtual layer and the upper boundary.

2.4 Three-dimensional Mesh Generation

Once the set of points has been defined, it will be necessary to build a three-dimensional mesh able to connect the points in an appropriate way and which conforms with the domain boundary, i.e., a mesh that respects every established boundary.

Although Delaunay triangulation is suitable to generate finite element meshes with a high regularity degree for a given set of points, this does not occur in the problem of conformity with the boundary, as it generates a mesh of the convex hull of the set of points. It may be thus impossible to recover the domain boundary from the faces and edges generated by the triangulation. To avoid this, we have two different sorts of techniques: *conforming Delaunay triangulation* [17] and *constrained Delaunay triangulation* [5]. The first alternative is inadequate for our purpose, as we wish the resulting mesh to contain certain predetermined points. Moreover, given the terrain surface complexity, this strategy would imply a high computational cost. The second alternative could provide another solution, but it requires quite complex algorithms to recover the domain boundary.

To build the three-dimensional Delaunay triangulation of the domain points, we start by resetting them in an auxiliary parallelepiped, so that every point of the terrain surface is on the original coordinates x, y, but at an altitude equal to the minimum terrain height, z_{min}. In the upper plane of the parallelepiped we set the nodes of level τ_1 of the mesh sequence that defines the terrain surface at altitude h. Generally, the remaining points also keep their coordinates x, y, but their heights are obtained by replacing their corresponding z_0 by z_{min} in (2). The triangulation of this set of points is done using a variant of Watson incremental algorithm [8] that effectively solves the problems derived from the round-off errors made when working with floating coma numbers.

Once the triangulation is built in the parallelepiped, the final mesh is obtained by re-establishing its original heights. This latter process can be understood as a compression of the global mesh defined in the parallelepiped, such that its lowest plane becomes the terrain surface. In this way, conformity is ensured.

Sometimes when re-establishing the position of each point to its real height, poor quality, or even *inverted* elements may occur. For inverted elements, their volume V_e, evaluated as the Jacobian determinant $|J_e|$ associated with the map from reference tetrahedron to the physical one e, becomes negative. For this reason, we need a procedure to untangle and smooth the resulting mesh, as analysed in the following section.

We must also take into account the possibility of getting a high quality mesh by smoothing algorithms, based on movements of nodes around their initial positions, depends on the *topological quality* of the mesh. It is understood that this quality is high when every *node valence*, i.e., the number of nodes connected to it, approaches the valence corresponding to a regular mesh formed by *quasi-equilateral* tetrahedra.

Our domain mesh keeps the topological quality of the triangulation obtained in the parallelepiped and an appropriate smoothing would thus lead to high quality meshes.

3 MESH OPTIMISATION WITH IMPROVED OBJECTIVE FUNCTIONS

In finite element simulation the mesh quality is a crucial aspect for good numerical behaviour of the method. In a first stage, some automatic 3-D mesh generator constructs meshes with poor quality and, in special cases, for example when node movement is required, inverted elements may appear. So, it is necessary to develop a procedure that optimises the pre-existing mesh. This process must be able to smooth and untangle the mesh.

The most usual techniques to improve the quality of a *valid* mesh, that is, one that does not have inverted elements, are based upon local smoothing. In short, these techniques consist of finding the new positions that the mesh nodes must hold, in such a way that they optimise an objective function. Such a function is based on a certain measurement of the quality of the *local submesh,* $N(v)$, formed by the set of tetrahedra connected to the *free node* v. As it is a local optimisation process, we can not guarantee that the final mesh is globally optimum. Nevertheless, after repeating this process several times for all the nodes of the current mesh, quite satisfactory results can be achieved. Usually, objective functions are appropriate to improve the quality of a valid mesh, but they do not work properly when there are inverted elements. This is because they present singularities (barriers) when any tetrahedron of $N(v)$ changes the sign of its Jacobian determinant. To avoid this problem we can proceed as Freitag et al in [18, 19], where an optimisation method consisting of two stages is proposed. In the first one, the possible inverted elements are untangled by an algorithm that maximises their negative Jacobian determinants [19]; in the second, the resulting mesh from the first stage is smoothed using another objective function based on a quality metric of the tetrahedra of $N(v)$ [18]. One of these objective functions are presented in Section 3.1. After the untangling procedure, the mesh has a very poor quality because the technique has no motivation to create good-quality elements. As remarked in [18], it is not possible to apply a gradient-based algorithm to optimise the objective function because it is not continuous all over $\mathbb{R}^3$, making it necessary to use other non-standard approaches.

In Section 3.2 we propose an alternative to this procedure, such that the untangling and smoothing are carried out in the same stage. For this purpose, we use a suitable modification of the objective function such that it is regular all over $\mathbb{R}^3$. When a feasible region (subset of $\mathbb{R}^3$ where v could be placed, being $N(v)$ a valid submesh) exists, the minima of the original and modified objective functions are very close and, when this region does not exist, the minimum of the modified objective function is located in such a way that it tends to untangle $N(v)$. The latter occurs, for example, when the fixed boundary of $N(v)$ is tangled. With this approach, we can use any standard and efficient unconstrained optimisation method to find the minimum of the modified objective function, see for example [20].

In this work we have applied the proposed modification to one objective function derived from an *algebraic mesh quality metric* studied in [10], but it would also be possible to apply it to other objective functions which have barriers like those presented in [21].

3.1 Objective Functions

Several *tetrahedron shape measures* [22] could be used to construct an objective function. Nevertheless those obtained by algebraic operations are specially indicated for our purpose because they can be computed very efficiently. The above mentioned algebraic mesh quality metric and the corresponding objective function are shown in this Section.

Let T be a tetrahedral element in the physical space whose vertices are given by $\mathbf{x}_k = (x_k, y_k, z_k)^T \in \mathbb{R}^3$, $k = 0, 1, 2, 3$ and T_R be the reference tetrahedron with vertices $\mathbf{u}_0 = (0,0,0)^T$, $\mathbf{u}_1 = (1,0,0)^T$, $\mathbf{u}_2 = (0,1,0)^T$ and $\mathbf{u}_3 = (0,0,1)^T$. If we choose $\mathbf{x}_0$ as the translation vector, the affine map that takes T_R to T is $\mathbf{x} = A\mathbf{u} + \mathbf{x}_0$, where A is the Jacobian matrix of the affine map referenced to node $\mathbf{x}_0$, and expressed as $A = (\mathbf{x}_1 - \mathbf{x}_0, \mathbf{x}_2 - \mathbf{x}_0, \mathbf{x}_3 - \mathbf{x}_0)$.

Let now T_I be an equilateral tetrahedron with all its edges of length one and vertices located at $\mathbf{v}_0 = (0,0,0)^T$, $\mathbf{v}_1 = (1,0,0)^T$, $\mathbf{v}_2 = (1/2, \sqrt{3}/2, 0)^T$, $\mathbf{v}_3 = \left(1/2, \sqrt{3}/6, \sqrt{2}/\sqrt{3}\right)^T$. Let $\mathbf{v} = W\mathbf{u}$ be the linear map that takes T_R to T_I, being $W = (\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3)$ its Jacobian matrix.

Therefore, the affine map that takes T_I to T is given by $\mathbf{x} = AW^{-1}\mathbf{v} + \mathbf{x}_0$, and its Jacobian matrix is $S = AW^{-1}$. This weighted matrix S is independent of the node chosen as reference; it is said to be *node invariant* [10]. We can use matrix norms, determinant or trace of S to construct algebraic quality measures of T. For example, the Frobenius norm of S, defined by $|S| = \sqrt{\operatorname{tr}(S^T S)}$, is specially indicated because it is easily computable. Thus, it is shown in [10] that $q = \frac{3\sigma^{\frac{2}{3}}}{|S|^2}$ is an algebraic quality measure of T, where $\sigma = \det(S)$. The maximum value of these quality measures is the unity and it corresponds to equilateral tetrahedron. Besides, any flat tetrahedron has quality measure zero. We can derive an optimisation function from this quality measure.

Thus, let $\mathbf{x} = (x, y, z)^T$ be the free node position of v, and let S_m be the weighted Jacobian matrix of the m-th tetrahedron of $N(v)$. We define the objective function of $\mathbf{x}$, associated to an m-th tetrahedron as

$$\eta_m = \frac{|S_m|^2}{3\sigma_m^{\frac{2}{3}}} \tag{11}$$

Then, the corresponding objective function for $N(v)$ can be constructed by using the p-norm of $(\eta_1, \eta_2, \ldots, \eta_M)$ as

$$|K_\eta|_p(\mathbf{x}) = \left[\sum_{m=1}^{M} \eta_m^p(\mathbf{x})\right]^{\frac{1}{p}} \tag{12}$$

where M is the number of tetrahedra in $N(v)$. The objective function $|K_\eta|_1$ was deduced and used in [23] for smoothing and adapting of 2-D meshes. The same function was introduced in [11], for both 2 and 3-D mesh smoothing, as a result of a force-directed method. Finally, this function, among others, is studied and compared in [21]. We note that the cited authors only use this objective function for smoothing valid meshes.

Although this optimisation function is smooth in those points where $N(v)$ is a valid submesh, it becomes discontinuous when the volume of any tetrahedron of $N(v)$ goes to zero. It is due to the fact that η_m approaches infinity when σ_m tends to zero and its numerator is bounded below. In fact, it is possible to prove that $|S_m|$ reaches its minimum, with strictly positive value, when v is placed in the geometric centre of the fixed face of the m-th tetrahedron. The positions where v must be located to get $N(v)$ to be valid, i.e., the feasible region, is the interior of the polyhedral set P defined as $P = \bigcap_{m=1}^{M} H_m$, where H_m are the half-spaces defined by $\sigma_m(\mathbf{x}) \geqslant 0$. This set can occasionally be empty, for example, when the fixed boundary of $N(v)$ is tangled. In this situation, function $|K_\eta|_p$ stops being useful as optimisation function. On the other hand, when the feasible region exists, that is *int* $P \neq \emptyset$, the objective function tends to infinity as v approaches the boundary of P. Due to these singularities, a barrier is formed which avoids reaching the appropriate minimum by using gradient-based algorithms, when these start from a free node outside the feasible region. In other words, with these algorithms we can not optimise a tangled mesh $N(v)$ with the above objective function.

3.2 Modified Objective Functions

We propose a modification in the previous objective function (12), so that the barrier associated with its singularities will be eliminated and the new function will be smooth all over $\mathbb{R}^3$. An essential requirement is that the minima of the original and modified functions are nearly identical when *int* $P \neq \emptyset$. Our modification consists of substituting σ in (12) by the positive and increasing function

$$h(\sigma) = \frac{1}{2}(\sigma + \sqrt{\sigma^2 + 4\delta^2}) \tag{13}$$

being the parameter $\delta = h(0)$. We represent in Figure 1 the function $h(\sigma)$. Thus, the new objective function here proposed is given by

$$\left|K_\eta^*\right|_p(\mathbf{x}) = \left[\sum_{m=1}^{M} \left(\eta_m^*\right)^p(\mathbf{x})\right]^{\frac{1}{p}} \tag{14}$$

where

$$\eta_m^* = \frac{\left|S_m\right|^2}{3h^{\frac{2}{3}}\left(\sigma_m\right)} \tag{15}$$

is the modified objective function for the m-th tetrahedron.

The behaviour of $h(\sigma)$ in function of δ parameter is such that, $\lim\limits_{\delta \to 0} h(\sigma) = \sigma$, $\forall \sigma \geq 0$ and $\lim\limits_{\delta \to 0} h(\sigma) = 0$, $\forall \sigma \leq 0$. Thus, if $int\ P \neq \emptyset$, then $\forall \mathbf{x} \in int\ P$ we have $\sigma_m(\mathbf{x}) > 0$, for $m = 1, 2, \ldots, M$ and, as smaller values of δ are chosen, $h(\sigma_m)$ behaves very much as σ_m, so that, the original objective function and its corresponding modified version are very close in the feasible region. Particularly, in the feasible region, as $\delta \to 0$, function $\left|K_\eta^*\right|_p$ converges pointwise to $\left|K_\eta\right|_p$. Besides, by considering that $\forall \sigma > 0$, $\lim\limits_{\delta \to 0} h'(\sigma) = 1$ and $\lim\limits_{\delta \to 0} h^{(n)}(\sigma) = 0$, for $n \geq 2$, it is easy to prove that the derivatives of this objective function verify the same property of convergence. As a result of these considerations, it may be concluded that the positions of v that minimise original and modified objective functions are nearly identical when δ is *small*. Actually, the value of δ is selected in terms of point v under consideration, making it as small as possible and in such a way that the evaluation of the minimum of modified functions does not present any computational problem. Suppose that $int\ P = \emptyset$,

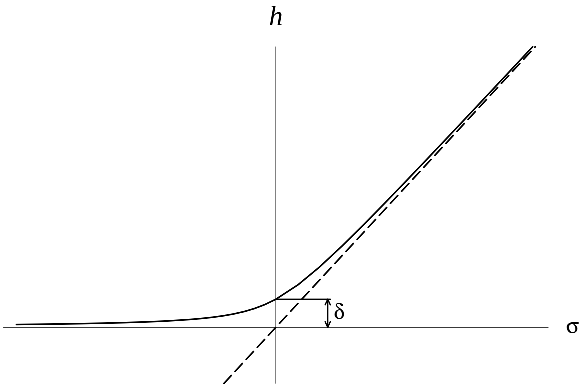


Figure 1: Representation of function $h(\sigma)$.

then the original objective function, $\left|K_\eta\right|_p$, is not suitable for our purpose because it is not correctly defined. Nevertheless, modified function is well defined and tends to solve the tangle. We can reason it from a qualitative point of view by considering that the dominant terms in $\left|K_\eta^*\right|_p$ are those associated to the tetrahedra with more negative values of σ and, therefore, the minimisation of these terms imply the increase of these values. It must be remarked that $h(\sigma)$ is an increasing function and $\left|K_\eta^*\right|_p$ tends to ∞ when the volume of any tetrahedron of $N(v)$ tends to $-\infty$, since $\lim_{\sigma\to-\infty} h(\sigma)=0$.

In conclusion, by using the modified objective function, we can untangle the mesh and, at the same time, improve its quality. More details about this mesh optimisation procedure can be seen in reference [9].

4 LOCAL MESH REFINEMENT ALGORITHM

We propose a local refinement algorithm [12] based on the 8-subtetrahedron subdivision developed in [13]. Consider an initial triangulation τ_1 of the domain given by a set of n_1 tetrahedra $t_1^1,\ t_2^1,...,t_{n_1}^1$. Our goal is to build a sequence of m levels of nested meshes $T=\{\tau_1<\tau_2<...<\tau_m\}$, such that the level τ_{j+1} is obtained from a local refinement of the previous level τ_j. The error indicator ϵ_i^j will be associated to the element $t_i^j\in\tau_j$. Once the error indicator ϵ_i^j is computed, such element must be refined if $\epsilon_i^j\geq\theta\epsilon_{\max}^j$, being $\theta\in[0,1]$ the refinement parameter and $\epsilon_{\max}^j$ the maximal value of the error indicators of the elements of τ_j. From a constructive point of view, initially we shall obtain τ_2 from the initial mesh τ_1, attending to the following considerations:

a) *8-subtetrahedron subdivision.* A tetrahedron $t_i^1\in\tau_1$ is called of *type I* if $\epsilon_i^1\geq\gamma\epsilon_{\max}^1$. Later, this set of tetrahedra will be subdivided into 8 subtetrahedra as Figure 2(a) shows; 6 new nodes are introduced in the middle point of its edges and each one of its faces are subdivided into four subtriangles following the division proposed by Bank [24]. Thus, four subtetrahedra are determinated from the four vertices of t_i^1 and the new edges. The other four subtetrahedra are obtained by joining the two nearest opposite vertices of the octohedron which result inside t_i^1. This simple strategy is that proposed in [13] or in [15], in opposite to others based on afin transformations to a reference tetrahedron, as that analysed in [14] which ensures the quality of the resulting tetrahedra. However, similar results were obtained by Bornemann et al. [15] with both strategies in their numerical experiments. On the other hand, for Lohner and Baum [13], this choice produces the lowest number of distorted tetrahedra in the refined mesh. Evidently, the best of the three existing options for the subdivision of the inner octohedron may be determined by analysing the quality of its four subtetrahedra, but this would augment the computational cost of the algorithm.

Once the *type I* tetrahedral subdivision is defined, we can find neighbouring tetrahedra which may have 6, 5, ..., 1 or 0 new nodes introduced in their edges that must be taken into account in order to ensure the mesh conformity. In the following we analyse each one of these cases. We must remark that in this process we only mark the edges of the tetrahedra of τ_1 in which a new node has been introduced. The corresponding tetrahedron is classified depending on the number of marked edges. In other words, until the conformity of τ_2 is not ensured by marking edges, this new mesh will not be defined.

b) *Tetrahedra with 6 new nodes.* Those tetrahedra that have marked their 6 edges for conformity reason, are included in the set of *type I* tetrahedra.

c) *Tetrahedra with 5 new nodes.* Those tetrahedra with 5 marked edges are also included in the set of *type I* tetrahedra. Previously, the edge without new node must be marked.

d) *Tetrahedra with 4 new nodes.* In this case, we mark the two free edges and it is considered as *type I*.

Proceeding as in (b), (c) and (d), we improve the mesh quality and simplify the algorithm considerably due to the global refinement defined in (a) by the error indicator. One may think that this procedure can augment the refined region, but we must take into account that only 1 or 2 new nodes are introduced from a total of 6. Note that this proportion is less or equal to that arising in the 2-D refinement with the 4-T Rivara algorithm, see for example References [16, 6], in which the probability of finding a new node introduced in the longest edge of a triangle is 1/3. This fact is accentuated in the proposed algorithm [25] as its generalization in 3-D.

e) *Tetrahedra with 3 new nodes.* In this case, we must distinguish two situations:

e.1) If the 3 marked edges are not located on the same face, then we mark the others and the tetrahedron is introduced in the set of *type I* tetrahedra. Here, we can make the previous consideration too, if we compare this step with other algorithms based on the bisection by the longer edge.

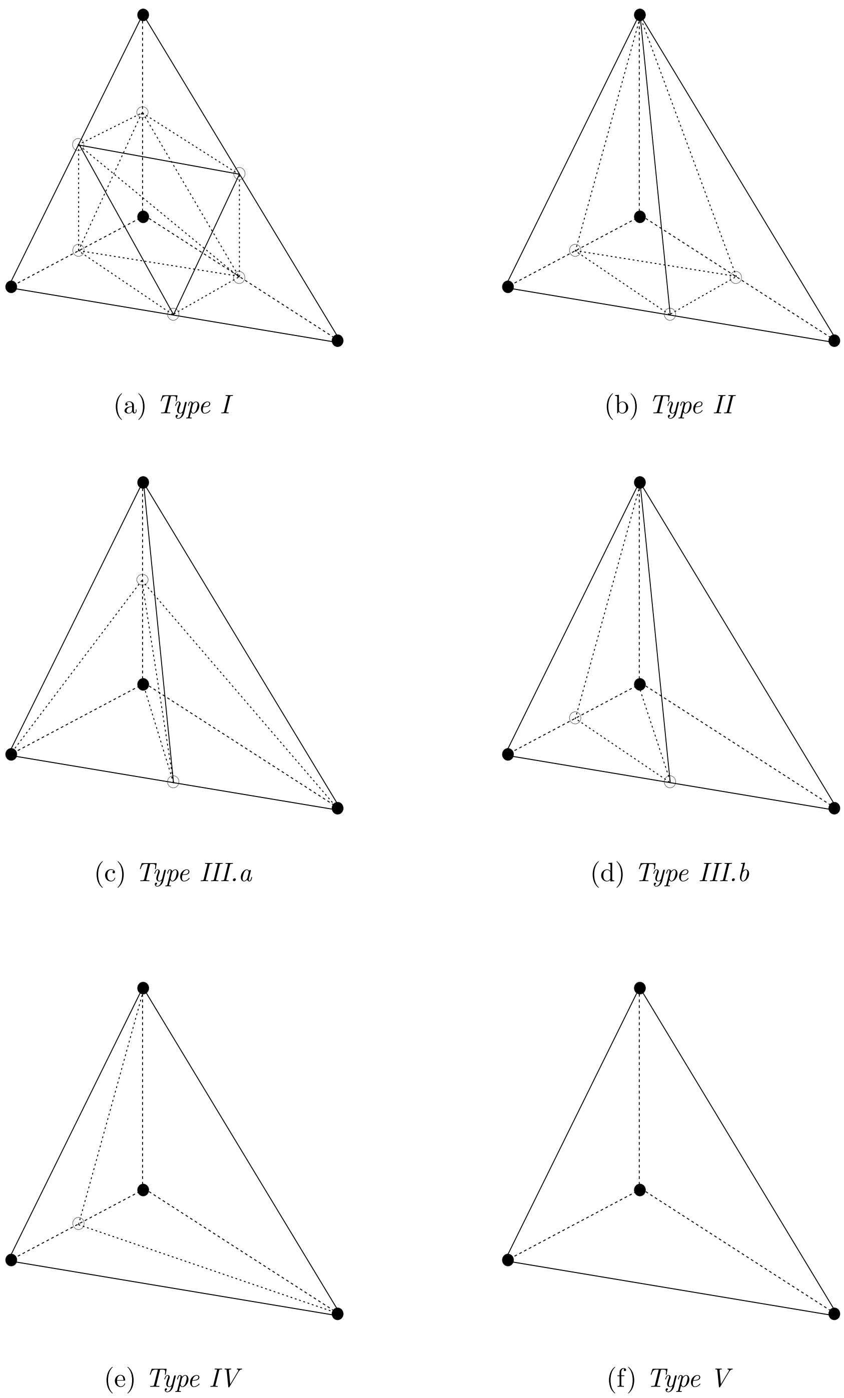
In the following cases, we shall not mark any edge, i.e., any new node will not be introduced in a tetrahedron for conformity. We shall subdivide them creating subtetrahedra which will be called *transient subtetrahedra*.

e.2) If the 3 marked edges are located on the same face of the tetrahedron, then 4 transient subtetrahedra are created as Figure 2(b) shows. New edges are created by connecting the 3 new nodes one another and these with the vertex opposite to the face containing them. The tetrahedra of τ_1 with these characteristics will be inserted in the set of *type II* tetrahedra.

f) *Tetrahedra with 2 new nodes.* Also here, we shall distinguish two situations:

f.1) If the two marked edges are not located on the same face, then 4 transient subtetrahedra will be constructed from the edges conecting both new nodes and these with the vertices opposite to the two faces which contain each one of them. This tetrahedra are called *type III.a*; see Figure 2(c).

f.2) If the two marked edges are located on the same face, then 3 transient subtetrahedra are generated as Figure 2(d) shows. The face determinated by both marked edges is divided into 3 subtriangles, connecting the new node located in the longest edge with the vertex opposite and with the another new node, such that these three subtriangles and the vertex opposite to the face which contains them define three new subtetrahedra. We remark that from the two possible choices, the longest marked edge is fixed as reference in order to take advantage in some cases of the properties of the bisection by the longest edge. These tetrahedra are called *type III.b*.



(a) *Type I* (b) *Type II*

(c) *Type III.a* (d) *Type III.b*

(e) *Type IV* (f) *Type V*

Figure 2: Subdivision classification of a tetrahedron in function of the new nodes (white circles).

g) *Tetrahedra with 1 new node.* Two transient subtetrahedra will be created as we can see in Figure 1(e). The new node is connected to the other two which are not located in the marked edge. This set of tetrahedra is called *type IV*.

h) *Tetrahedra without new node.* These tetrahedra of τ_1 are not divided and they will be inherit by the refined mesh τ_2. We call them *type V* tetrahedra; see Figure 2(f).

This classification process of the tetrahedra of τ_1 is carried out by marking their edges simply. The mesh conformity is ensured in a local level analysing the neighbourhood between the tetrahedra which contain a marked edge by an expansion process that starts in the *type I* tetrahedra of paragraph (a). Thus, when the run along this set of *type I* tetrahedra is over, the resulting mesh is conformal and locally refined.

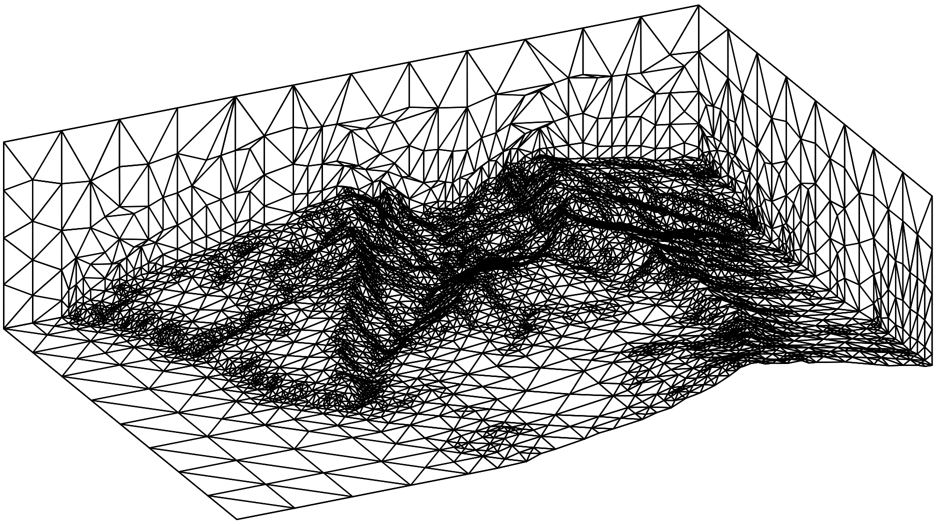
Moreover, this is a low computational cost process, since the local expansion stops when we find tetrahedra whose edges have not to be marked. Implementations details in C++ are discussed in [12].

Generally, when we want to refine the level τ_j in which there already exist transient tetrahedra, we shall perform it in the same way as from τ_1 to τ_2, except for the following variation: if an edge of any transient tetrahedron must be marked, due to the error indicator or even to conformity reasons, then all the transient tetrahedra are eliminated from their parent (deleting process), all the parent edges are marked and this tetrahedron is introduced into the set of *type I* tetrahedra. We must remark that it will be only necessary to define a variable which determines if a tetrahedron is transient or not.

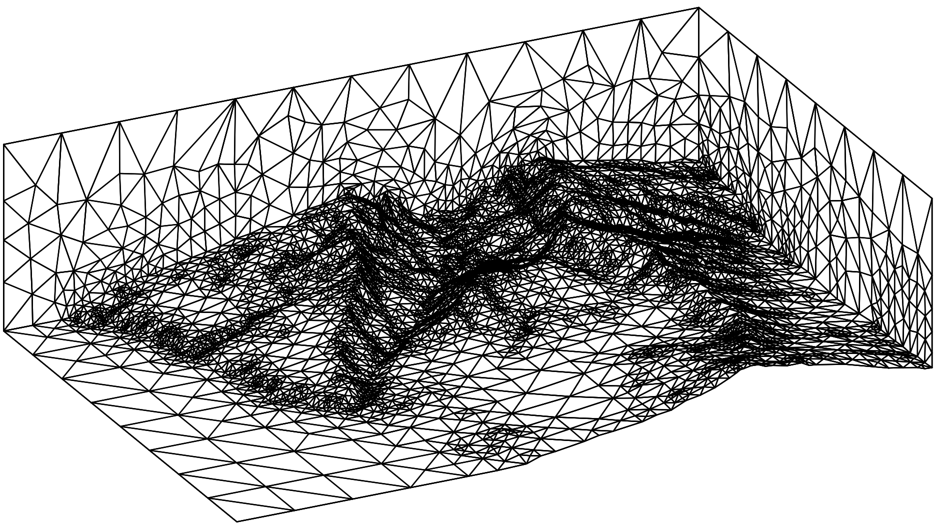
5 APPLICATIONS

As a practical application of our mesh generator and the optimisation procedure we have taken under consideration a rectangular area in Isla de La Palma (Canary Islands) of $22.8 \times 15.6\ km$, where extreme heights vary from 0 to 2279 m. The upper boundary of the domain has been placed at $h = 6\ km$. To define the topography we had a digitalisation of the area where heights were defined over a grid with a spacing step of 200 m in directions x and y. Starting from a uniform 2-D mesh τ_1 of the rectangular area with a size of elements about $2 \times 2\ km$, six global refinements were made using Rivara 4-T algorithm [16]. Once the data were interpolated on this refined mesh, the derefinement algorithm developed in [6] and [7] with a derefinement parameter of $\varepsilon = 25\ m$ was used. Thus, the adapted mesh nears the terrain surface with an error less than that value. The node distribution of τ_1 is the one considered on the upper boundary of the domain.

The meshes have 81068 tetrahedra and 16504 nodes, with a maximum valence of 36, see Fig. 3. The initial tangled mesh has 574 inverted tetrahedra with an average quality measure $\overline{q}_\kappa = 0.626$, see reference [9] and Fig. 4. The node distribution is hardly modified after ten steps of the optimisation process using $\left|K_\eta^*\right|_1$. All the inverted elements disappear in the first step of this process and average quality measure increases to $\overline{q}_\kappa = 0.706$. This measure tends to stagnate quickly; in the 5-th step we obtain $\overline{q}_\kappa = 0.732$ and in the 10-th step $\overline{q}_\kappa = 0.734$. After this optimisation process, the worst quality measure of the optimised mesh tetrahedra is 0.112.



(a)



(b)

Figure 3: Rectangular area of Isla de La Palma (Canary Islands): (a) initial mesh with 574 inverted tetrahedra and (b) resulting valid mesh after ten steps of the optimisation process.

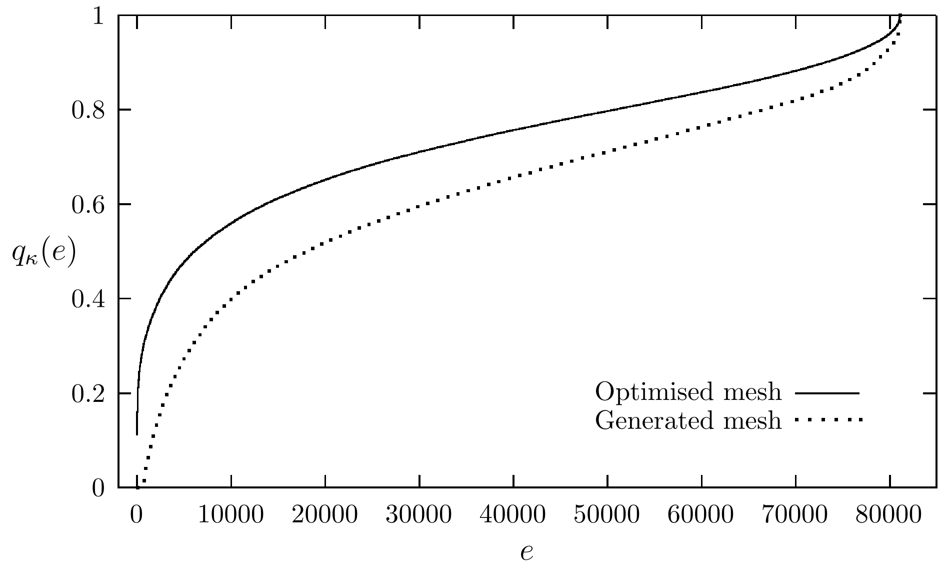


Figure 4: Quality curves of the generated mesh and the resulting mesh after ten steps of the optimisation process. Function $q_\kappa(e)$ is a quality measure for tetrahedron e.

We remark that the number of parameters necessary to define the resulting mesh is quite low, as well as the computational cost.

If we want to simulate the atmospheric pollution produced by a chimney of a test power-station placed in the region of Isla de La Palma, we have to introduce the geometry of the chimney in the topography and to apply the 3-D mesh generator with a few modifications. Let us consider a chimney of a heigth of 200 m over the terrain and with a diameter of 25 m at its top and 50 m at its bottom. In this application we have fixed the upper boundary of the domain at $h = 9$ km. The mesh must be able to detect the details of the chimney. So, if we decide a size of elements about 2×2 m in the chimney, starting from the uniform 2-D mesh τ_1 of the rectangular area with a size of elements about 2×2 km, we should have to make ten global refinements using Rivara 4-T algorithm [16]. For memory reasons, this procedure should be impracticable. Therefore, our strategy was to make only five global refinements over τ_1 and, after, to refine five times only the elements included in the chimney.

In this case, we applied the derefinement algorithm with a parameter $\varepsilon = 40$ m considering that nodes included in the chimney could not be eliminated. The mesh have been constructed by using the strategy 1 with $\alpha = 2$ and we ditributed the nodes over 11 layers ($n = 10$), including the terrain and the upper boundary of the domain; see reference [2]. The resulting mesh have 197836 tetrahedra and 30424 nodes, with a maximum valence of 25. We have 5947 nodes over the terrain and only 117 nodes over the upper bounday of the domain. The initial tangled mesh has 56 inverted tetrahedra. Fig. 5 shows four details of the mesh near the chimney before applying the optimisation process.

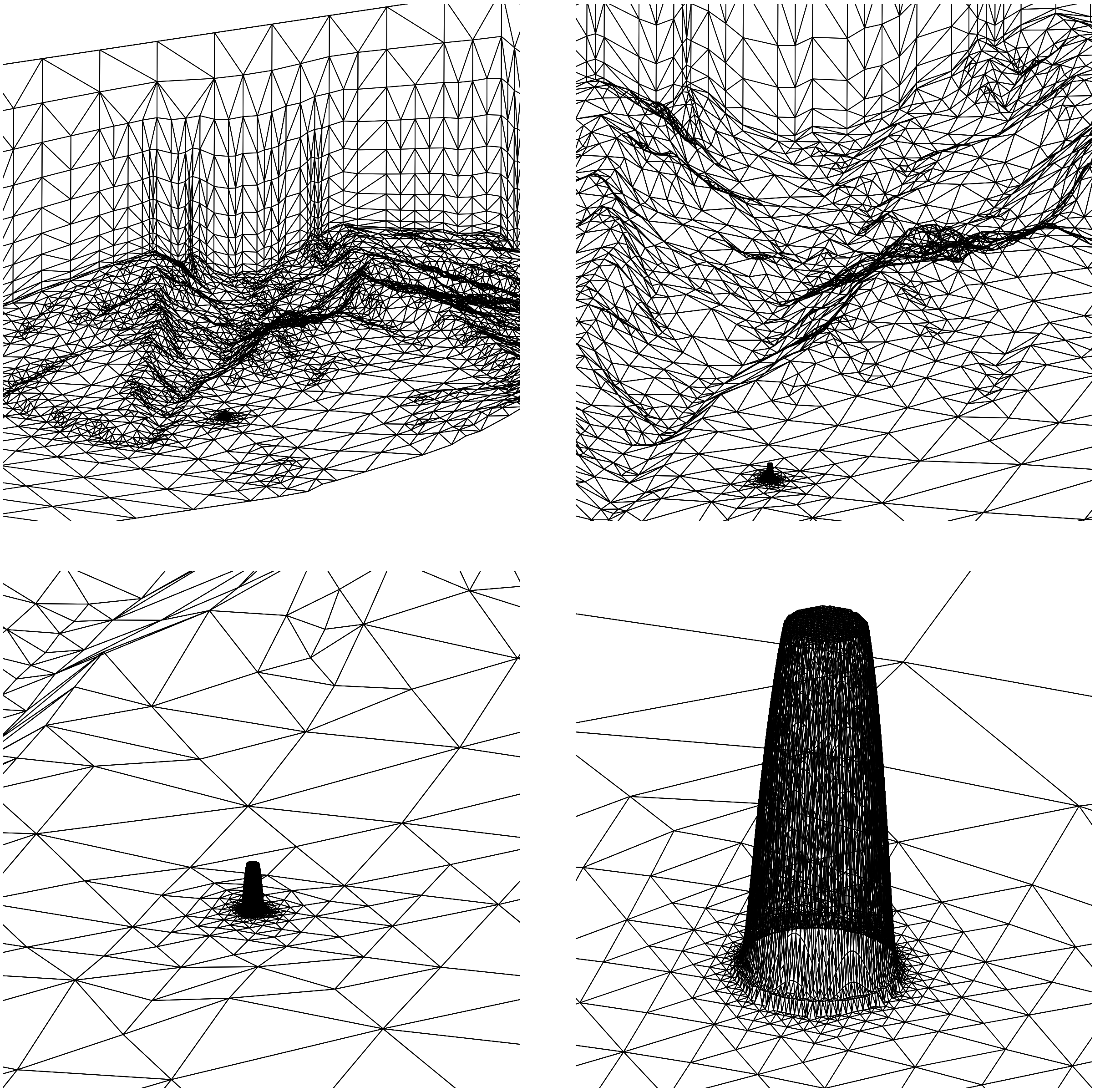


Figure 5: Four details of a 3-D mesh of Isla de La Palma including the discretization of a chimney.

6 CONCLUSIONS

We have presented efficient results in automatic and adaptive 3-D mesh generation for environmental problems. So, we can discretize a very complex domain with a minimal user intervention and low computational cost.

A promising field of study would combine the 3-D refinement/derefinement of nested meshes with node movement, where the ideas presented here could be introduced. Good recent results have been obtained in [26] and [27] using these techniques, for determining the shape and size of the elements in anisotropic problems. We have introduced several results [28] to improve the mesh quality by combining smoothing techniques and local refinement.

ACKNOWLEDGEMENT

This work has been supported by the Spanish Government's Science and Technology Department (Ministerio de Ciencia y Tecnología) and FEDER, grant contract: REN2001-0925-C03-02/CLI.

REFERENCES

[1] R. Montenegro, G. Montero, J.M. Escobar, E. Rodríguez and J.M. González-Yuste. Tetrahedral mesh generation for environmental problems over complex terrains. *Lecture Notes in Computer Science*, **2329**, 335–344, 2002.

[2] R. Montenegro, G. Montero, J.M. Escobar and E. Rodríguez. Efficient strategies for adaptive 3-D mesh generation over complex orography. *Neural, Parallel & Scientific Computation*, **10**, 1, 57–76, 2002.

[3] G. Montero, R. Montenegro and J.M. Escobar. A 3-D diagnostic model for wind field adjustment. *J. of Wind Eng. and Ind. Aerodynamics*, **74-76**, 249–261, 1998.

[4] R. Montenegro, A. Plaza, L. Ferragut and I. Asensio. Application of a nonlinear evolution model to fire propagation. *Nonlinear Analysis, Th., Meth. & App.*, **30**, 5, 2873–2882, 1997.

[5] P.L. George, F. Hecht, and E. Saltel. Automatic mesh generation with specified boundary. *Comp. Meth. Appl. Mech. Eng.*, **92**, pp. 269–288, 1991.

[6] L. Ferragut, R. Montenegro and A. Plaza. Efficient refinement/derefinement algorithm of nested meshes to solve evolution problems. *Comm. Num. Meth. Eng.*, **10**, 403–412, 1994.

[7] A. Plaza, R. Montenegro and L. Ferragut. An improved derefinement algorithm of nested meshes. *Advances in Engineering Software*, **27**, 1/2, 51–57, 1996.

[8] J.M. Escobar and R. Montenegro. Several aspects of three-dimensional Delaunay triangulation. *Advances in Engineering Software*, **27**, 1/2, 27–39, 1996.

[9] J.M. Escobar, E. Rodríguez, R. Montenegro, G. Montero and J.M. González-Yuste. Simultaneous untangling and smoothing of tetrahedral meshes. *Comp. Meth. Appl. Mech. Eng.*, in press, 2003.

[10] P.M. Knupp. Algebraic mesh quality metrics. *SIAM. J. Sci. Comput.*, **23**, 193–218, 2001.

[11] H.N. Djidjev. Force-directed methods for smoothing unstructured triangular and tetrahedral meshes. *Tech. Report*, Department of Computer Science, Univ. of Warwick, Coventry, UK, (2000). Available from http://www.andrew.cmu.edu/user.

[12] J.M. González-Yuste, R. Montenegro, J.M. Escobar, G. Montero and E. Rodríguez. An object oriented method for tetrahedral mesh refinement. *Proc. The Third International Conference on Engineering Computational Technology*, Civil-Comp Press, in CD-ROM, Paper 72, 1–18, 2002.

[13] R. Lohner and J.D. Baum. Adaptive h-refinement on 3D unstructured grids for transient problems. *Int. J. Num. Meth. Fluids*, **14**, 1407–1419, 1992.

[14] A. Liu and B. Joe. Quality local refinement of tetrahedral meshes based on 8-subtetrahedron subdivision. *Mathematics of Comput.*, **65**, 215, 1183–1200, 1996.

[15] F. Bornemann, B. Erdmann and R. Kornhuber. Adaptive multilevel methods in three space dimensions. *Int. J. Numer. Meth. Eng.*, **36**, 3187–3203, 1993.

[16] M.C. Rivara, A grid generator based on 4-triangles conforming. Mesh-refinement algorithms. *Int. J. Numer. Meth. Eng.*, **24**, 1343–1354, 1987.

[17] M. Murphy, D.M. Mount and C.W. Gable. A point-placement strategy for conforming Delaunay tetrahedralization. *Symposium on Discrete Algorithms*, 67–74, 2000.

[18] L.A. Freitag and P.M. Knupp. Tetrahedral mesh improvement via optimization of the element condition number. *Int. J. Numer. Meth. Eng.*, **53**, 1377–1391, 2002.

[19] L.A. Freitag and P. Plassmann. Local optimization-based simplicial mesh untangling and improvement. *Int. J. Numer. Meth. Eng.*, **49**, 109–125, 2000.

[20] M.S. Bazaraa, H.D. Sherali and C.M. Shetty. *Nonlinear programing: Theory and algorithms*, John Wiley and Sons, Inc., 1993.

[21] P.M. Knupp. Achieving finite element mesh quality via optimization of the jacobian matrix norm and associated quantities. Part II - A frame work for volume mesh optimization and the condition number of the jacobian matrix. *Int. J. Numer. Meth. Eng.*, **48**, 1165–1185, 2000.

[22] J. Dompierre, P. Labbé, F. Guibault and R. Camarero. Proposal of benchmarks for 3D unstructured tetrahedral mesh optimization. *Proc. 7th International Meshing Roundtable*, Sandia National Laboratories, 459–478, 1998.

[23] R.E. Bank and R.K. Smith. Mesh smoothing using a posteriori error estimates. *SIAM J. Numer. Anal.*, **34**, 979–997, 1997.

[24] R.E. Bank, A.H. Sherman and A. Weiser. Refinement algorithms and data structures for regular local mesh refinement. *Scientific Computing IMACS*, North-Holland, 3–17, 1983.

[25] A. Plaza and G.F. Carey. Local refinement of simplicial grids based on the skeleton. *Appl. Numer. Math.*, **32**, 195–218, 2000.

[26] C.C. Pain, A.P. Umpleby, C.R.E. de Oliveira and A.J.H. Goddard. Tetrahedral mesh optimization for steady-state and transient finite element calculations. *Comput. Meth. Appl. Mech. Eng.*, **190**, 3771–3796, 2001.

[27] A. Tam, D. Ait-Ali-Yahia, M.P. Robichaud, A. Moore, V. Kozel and W.G. Habashi. Anisotropic mesh adaptation for 3-D flows on structured and unstructured grids. *Comput. Meth. Appl. Mech. Eng.*, **189**, 1205–1230, 2000.

[28] J.M. Escobar, R. Montenegro, G. Montero, E. Rodríguez and J.M. González-Yuste. Improvement of mesh quality by combining smoothing techniques and local refinement. *Proc. The Ninth International Conference on Civil and Structural Engineering Computing*, in press, 2003.