Results on 3-D Adaptive Finite Element Mesh Generation

R. Montenegro, G. Montero, J.M. Escobar,
E. Rodríguez, J.M. González-Yuste
University Institute of Intelligent Systems and Numerical Applications in Engineering,
University of Las Palmas de Gran Canaria,
Edificio Instituto Polivalente, Campus Universitario de Tafira,
35017-Las Palmas de Gran Canaria, Spain.
rafa@dma.ulpgc.es, gustavo@dma.ulpgc.es, escobar@cic.teleco.ulpgc.es,
barrera@dma.ulpgc.es, josem@sinf.ulpgc.es

1. INTRODUCTION

In the simulation of processes that occur in a 3-D domain, a mesh generator capable of adapting itself to the geometrical characteristics and to the numerical solution is essential. The objective of this work is to present our recent results in these topics. Particularly, in Sect. 2 we summarize a technique able to generate a tetrahedral mesh from an *optimal* node distribution in a domain defined over complex orography. The main ideas for the construction of the mesh combine the use of a refinement/derefinement algorithm for 2-D domains and a tetrahedral mesh generator algorithm based on Delaunay triangulation. Moreover, we propose a procedure to optimise the resulting mesh. To improve the corresponding numerical solution obtained by finite element method, we present an adaptable refinement of 3-D meshes. First, we compute an error indicator for each element of the mesh attending to the current numerical solution. This point out what elements must be refined. The proposed refinement technique, based on the subdivision in 8-subtetrahedra, allows higher discretization of the selected zones. This process may be repeated until the error of the numerical solution satisfies the imposed tolerance. Several ideas about this local refinement algorithm are presented in Sect. 3. Finally, adapted meshes are shown in Sect. 4.

2. TETRAHEDRAL MESH OVER COMPLEX OROGRAPHY

Here we intend to construct a tetrahedral mesh that approximates the orography of the terrain with a given precision. 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 (Montero et al., 1998), fire propagation (Montenegro et al., 1997) 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 (George et al., 1991) 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 (Ferragut et al., 1994; Plaza et al., 1996) 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 (Escobar & Montenegro, 1996). 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. We have proposed a method, so the untangling and smoothing are performed in the same stage. To do this, we shall use a modification of the objective function proposed in (Djidjev, 2000).

3. LOCAL REFINEMENT

Data structures, used in finite element meshes, are usually based on the definition of many arrays in which the mesh information is kept: nodes, edges, faces, tetrahedra, connectivity, genealogy, etc. Generally, in FORTRAN codes the memory management consisted in overdimension the variables for preventing the mesh changes; in case of using adaptive mesh refinement, the memory needs must be estimated previously. Besides, in case of using mesh derefinement, arrays must be compacted for recovering the memory space corresponding to deleted elements. With the appearance of languages as C, part of these problems could be solved. Besides, with C++ the qualitative advance is considerable, since the concept of structure is extended to class. A class contains, besides the proper information of an element, all operations which can be carried out on this one. On the other hand, object oriented programming introduces the concept of inheritance. Any class can be defined as inherited from another, such that a new class will have all the characteristics of its ancestor and other new ones. These properties allow us to generate hierarchical classes, that is, to create complex modules from simpler ones. Other interesting concept is encapsulation. All these tools are essential for improving the programming of adaptive finite element meshes. So, an hierarchy of elements have been designed to model the different parts of a mesh: nodes, edges, faces and tetrahedra. The definition is based on a very simple class, so called *element*, with a few properties, but commons to all of them. From this class, other classes will be inherited; for instance, the characteristics of the subdivision of an element maintain links between fathers and sons in the mesh. Obviously, except nodes, the rest of the elements are divisible. Another class will be responsible for maintaining links between different elements,

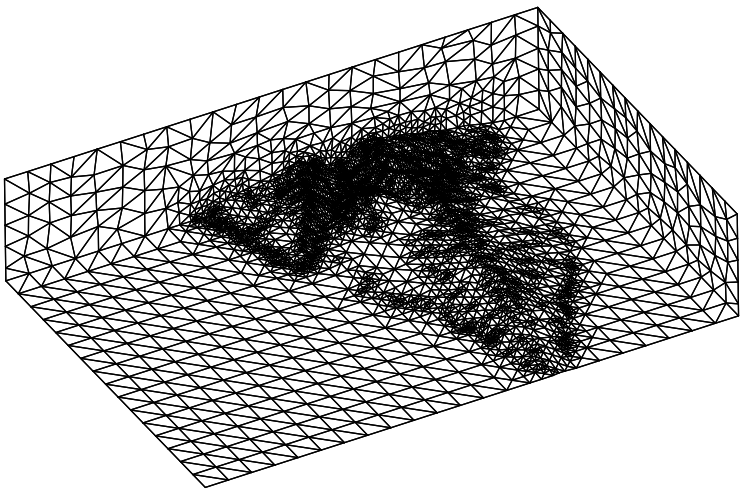


Figure 1: Resulting mesh after five steps of the optimisation process

such that the hierarchical structure of the mesh will be guaranteed every moment: nodes with edges, edges with faces and faces with tetrahedra. Due to this organisation, the neighbouring relationship can be easily determined and, thus, also the implementation of refinement and derefinement algorithms. A local refinement algorithm for tetrahedral meshes, based on the 8-subtetrahedra subdivision (Löhner & Baum, 1992; Liu & Joe 1996), have been efficiently developed with the proposed design.

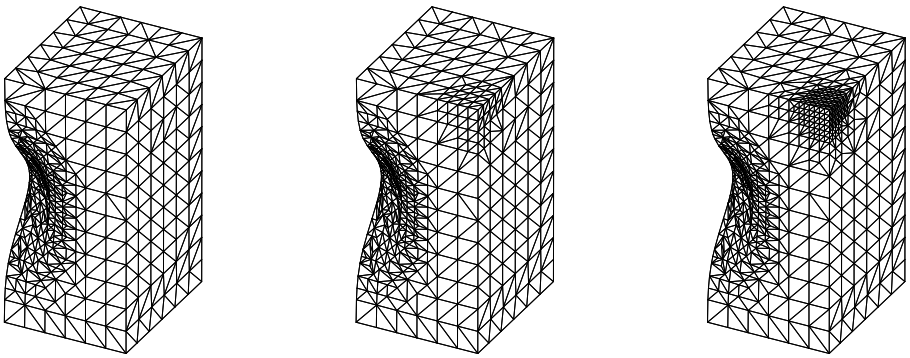
4. APPLICATIONS

As a practical application of the mesh generator, we have considered a rectangular area in the south of La Palma (Canary Islands) of $45.6 \times 31.2\ km$, where extreme heights vary from 0 to 2279 m. The upper boundary of the domain has been placed at $h = 9\ km$. To define the topography we used a digitalisation of the area where heights were defined over a grid with a spacing step of 200 m in directions x and y. The adapted mesh approximates the terrain surface with an error less than $\varepsilon = 40\ m$. The result obtained is shown in Fig. 1. In this case, the mesh has 57193 tetrahedra and 11841 nodes. To avoid inverted tetrahedra, our proposed technique has been efficiently applied. Moreover, the worst quality measure (Freitag & Knupp, 1999) of the optimised mesh tetrahedra is about 0.2. We note that the number of parameters necessary to define the resulting mesh is quite low, as well as the computational cost; only a few seconds of CPU time, on a Pentium III processor, are necessary for contructing the mesh before its optimisation.

Relative to the local refinement algorithm, an application is shown in Fig. 2. In this case, we have defined an error indicator in funcion of the distance from tetrahedra to a corner of the domain.

ACKNOWLEDGEMENT

Partially supported by MCYT, Spain. Grant contract: REN2001-0925-C03-02/CLI.



(a) (b) (c)

Figure 2: Application of the refinement algorithm; (a) initial mesh, (b) and (c) resulting meshes after 1 and 2 refinement steps, respectively

REFERENCES

1. Djidjev, H.N. (2000). Force-directed methods for smoothing unstructured triangular and tetrahedral meshes, *Tech. Report*, Dep. of Computer Science, Univ. of Warwick, Coventry, UK. Available from http://www.andrew.cmu.edu/user/sowen/topics/new.html.
2. Escobar,J.M., & Montenegro, R. (1996). Several aspects of three-dimensional Delaunay triangulation, *Advances in Engineering Software*, v. 27, 1/2, pp. 27-39.
3. Ferragut, L., Montenegro, R., & Plaza, A. (1994). Efficient refinement/derefinement algorithm of nested meshes to solve evolution problems, *Comm. Num. Meth. Eng.*, v. 10, pp. 403-412.
4. Freitag, L.A., & Knupp, P.M. (1999). Tetrahedral element shape optimization via the jacobian determinant and condition number, in *Proceedings of the Eighth International Meshing Roundtable*, Sandia National Laboratories, pp. 247-258.
5. George, P.L., Hecht, F., & Saltel, E. (1991). Automatic mesh generation with specified boundary, *Comp. Meth. Appl. Mech. Eng.*, v. 92, pp. 269-288.
6. Liu, A., & Joe, B. (1996). Quality local refinement of tetrahedral meshes based on 8-subtetrahedron subdivision, *Mathematics of Computations*, v. 65, 215, pp. 1183-1200.
7. Löhner, R., & Baum, J.D. (1992). Adaptive h-refinement on 3D unstructured grids for transient problems, *Int. J. Num. Meth. Fluids*, v. 14, pp. 1407-1419.
8. Montenegro, R., Plaza, A., Ferragut, L., & Asensio, I. (1997). Application of a nonlinear evolution model to fire propagation, *Nonlinear Analysis, Th., Meth. & App.*, v. 30, 5, pp. 2873-2882.
9. Montero, G., Montenegro, R., & Escobar, J.M. (1998). A 3-D diagnostic model for wind field adjustment, *J. of Wind Eng. and Ind. Aerodynamics*, v. 74-76, pp. 249-261.
10. Plaza, A., Montenegro, R., & Ferragut, L. (1996). An improved derefinement algorithm of nested meshes, *Advances in Engineering Software*, v.27, 1/2, pp. 51-57.