UNIVERSIDADE FEDERAL FLUMINENSE

SANDERSON LINCOHN GONZAGA DE OLIVEIRA

Graph-based adaptive simplicial-mesh refinement for finite volume discretizations

Niterói

2009

UNIVERSIDADE FEDERAL FLUMINENSE

SANDERSON LINCOHN GONZAGA DE OLIVEIRA

Graph-based adaptive simplicial-mesh refinement for finite volume discretizations

Thesis submitted to the Programa de Pós-Graduação em Computação at Universidade Federal Fluminense in partial fullfillment of the requirements for the degree of Doctor in Computer Science.

Advisor:
Mauricio Kischinhevsky

Niterói

2009

Graph-based adaptive simplicial-mesh refinement for finite volume discretizations

SANDERSON LINCOHN GONZAGA DE OLIVEIRA

Tese submetida ao Programa de Pós-Graduação em Computação da Universidade Federal Fluminense como requisito parcial para obtenção de título de Doutor. Área de concentração: Modelagem Computacional.

Aprovada por:

Prof. D.Sc. Mauricio Kischinhevsky / IC-UFF (Presidente)

Prof. Ph.D. Jorge Stolfi / IC-UNICAMP

Prof. Ph.D. Carlos Antonio de Moura / IME-UERJ

Prof. D.Sc. Luiz Nélio Henderson Guedes de Oliveira / IPRJ-UERJ

Profa. D.Sc. Regina Célia Paula Leal Toledo / IC-UFF

Profa. D.Sc. José Henrique Carneiro de Araújo / IC-UFF

Prof. D.Sc. Anselmo Antunes Montenegro / IC-UFF

Niterói, 26 de janeiro de 2009.

Este trabalho é dedicado

a meu pai,

Luiz Gonzaga de Oliveira,

à doce Tatiana Maduro

e

à memória de minha mãe,

Selma Andrade de Oliveira.

Acknowledgements

I must express my gratitute to many people who helped me during this work. Trying to mention all of them would certainly lead me to an incomplete list with someone important missing. However, I am particularly grateful to Professor Mauricio Kischinhevsky for his supervision and teaching and to Tatiana Barbosa Maduro, Jacques Alves da Silva, Diego Nunes Brandão and Mauricio José Guedes for their contributions. Likewise, I acknowledge CAPES for the financial support. Similarly, I thank the entire group of professors and employees from IC-UFF. I also must express my thanks to Prof. Denise Burgarelli for sending me the program code to solve the Laplace Equation based on the quadrangular ALG.

Abstract

This work proposes an adaptive mesh refinement technique for discretizations based on triangles of any shape. It employs the Finite Volume Method in a cell-centered scheme in order to solve partial differential equations. The mesh is represented by a graph data structure. The scheme intends to reduce the computational cost to construct the refined mesh in evolutionary problems. Furthermore, it admits more flexibility to link nodes among neighbors in different levels of refinement, in comparison to a local refinement scheme which uses a tree-like data structure. A Sierpiński-like Curve is used to order the finite volumes. Volumes are refined by the 4-triangles longest-side partition scheme, allowing straightforward updating of the linked list. A specific linear approximation is used to determine the gradients of diffusive and viscous terms and the dependent variable of the partial differential equation in the control volume is solved by a specific linear interpolation. Additionally, linear-system solvers based on the minimization of functionals can be easily employed. Specifically, the Conjugate Gradient Method is used. This approach enables any polygonal shape in the initial domain. Numerical results demonstrate the efficiency and advantages of this scheme.

Keywords

1. Numerical methods
2. Numerical solution and simulation of PDEs
3. Finite Volume Method
4. Adaptive mesh refinement
5. Computational analysis
6. Space-filling curve
7. Sierpiński Curve
8. Eulerian approach
9. Eulerian-Lagrangian approach

Glossary

$2D$: two-dimensional;

$3D$: three-dimensional;

4TLSP : 4-triangles longest-side partition;

ALG : Autonomous Leaves Graph;

AMR : adaptive mesh refinement;

FVM : Finite Volume Method;

M : mass inside a volume;

PDE : Partial Differential Equation;

R : reference length used in the Reynolds number;

S : source term;

p : pressure;

t : time;

$\vec{u}$: velocity vector;

u_∞ : air velocity in the free-stream region (flat-plate Boundary Layer Problem);

$\vec{A}$: vector area;

ν : kinematic viscosity;

ϕ : density of a substance;

ρ : specific fluid mass;

Γ^ϕ : diffusive coefficient;

$\triangle$: velocity boundary-layer thickness (flat-plate Boundary Layer Problem).

Contents

List of Figures

Chapter 1

Introduction

The Finite Volume Method is a numerical method for solution of partial differential equations (PDEs). It is a method for representing and evaluating PDEs as algebraic[1] equations and values are calculated at discrete places on a meshed geometry. This results in a linear system to be computed. When one increases the number of discrete places in the discretized domain in order to improve the quality of the numerical solution, the resulting linear system is proporcionally increased. As a result, the computational cost in order to compute the resulting linear system is also increased. On the other hand, the solution may require more discrete places only where the solution and/or its derivatives change rapidly. Thus, the adaptive mesh refinement (AMR) technique is used to concentrate more discrete places in certain regions of the mesh allowing a coarse level of refinement in other regions. Increasing discrete places only in such regions of the discretized domain presents two characteristics: *i)* allows to maintain the quality being sought and; *ii)* the resulting linear system is smaller than a solution in which the discretized domain is uniformly refined. For evolutionary problems, such scheme becomes particularly important since, by the dynamic nature of such problems, migration (or occurrence) of regions of rapid solution change may happen whereas the solution in other parts is relatively stable. Thus, a *non-uniform* (*irregular* or sometimes used as *unstructured* in the finite volume field) mesh is adaptively refined in order to improve the quality of a computed solution.

[1] Appendix A presents historical descriptions of some terms, researchers, and others.

1.1 Uniform meshes × AMR technique

In relation to data structures, a mesh of a numerical method should efficiently describe those features. A mesh is called uniform if each discrete place has the same number of neighbors. In terms of data structures, it requires to deal with discrete places just through indices of multi-dimensional arrays and matrices. Using a regular mesh in order to cover the domain of the differential problem is not appropriate in certain situations because those meshes do not take into account the different scales of the phenomena being studied. Usually, regular meshes are computationally expensive because they should have a large number of discrete places in the entire discretized domain in order to provide a quality solution.

On the other hand, techniques that use AMR are, usually, computationally less expensive in certain problems than using a uniform mesh. In other words, an AMR scheme can significantly improve the computational efficiency of scientific and engineering calculations involving solutions to large gradients, complicated geometries, and other special characteristics in problems. Emphasizing, the idea is to automatically build a coarse mesh whose numerical solution provides an appropriate approximation and to construct a fine mesh where the numerical solution does not supply an appropriate approximation among cells, such as singularities, boundary layers, among others. Furthermore, it should have a smooth transition among neighbor discrete places, which have different levels of refinement. Usually, an AMR reduces the required number of discrete places to obtain an accurate numerical solution to problems that are almost smooth, and can reach a numerical solution with the desired quality in non-smooth problems.

[Rivara, 1984a] explained that multigrid methods use iterative solvers of equations to smooth either the solution or related residual functions over fine and coarse grids. This proceeds only until the functions become smooth, namely, while the iterative methods are rapidly convergent. Adaptivity of the mesh is a central feature since it relieves the user of critical decisions and allows use of all the flexibility of the a numerical method for getting a minimum number of discrete places. In this sense, the generation of the discretization (triangulation) should not be a first separate step of the solution process, but a dynamic adaptive process. In order to deal with singular solutions, the capability for managing local refinement of the discretization is indispensable, and mesh refinement algorithms that maintain the nondegeneracy of the volumes and smoothness of the mesh are certainly desirable. The techniques chosen to satisfy the different goals proposed should also fit together in an efficient way. One of the most critical decisions with this kind of software is the definition of data structure to be used to construct the sequence of discretized problems and to manage the multigrid method. This definition must necessarily consider the different algorithms to be used with it.

1.2 Tree-like × graph-based data structures

Usually, a tree-like data structure is used in order to construct regions of the mesh with more discrete places. From an initial discretization, represented by the root of a tree-like data strucutre, children nodes are created recursively, which represent the discretized domain with more discrete places. When setting up the linear system, connectivity information among neighbors is needed. Using a tree-like data structutre, this can be only obtained from a leaf node l_i seeking (up) many nodes until a certain original node in common (also one original node of the leaf node being searched) and again seeking (down) the leaf node l_j that is the neighbor of the leaf l_i.

In order to avoid this overhead, [Burgarelli, Kischinhevsky and Biezuner, 2006] proposed a graph-based data structure for representing the AMR. In this work, either neighbors can be directly connected or the steps to reach nodes among neighbors are greatly reduced when there are different levels of refinement among cells. The authors proposed a simple, yet flexible and powerful, strategy within a group of methods that deal with the AMR scheme, which provides local mesh refinement with low computational cost. The technique displays a plug-in feature that enables replacement of a single discrete place by a pack of discrete places in any region of interest. Moreover, only strict local changes in the data structure occur. Besides, low storage is achieved since the refined nodes are not stored. The authors named the graph data structure as Autonomous Leaves Graph (ALG). ALG represents a mesh composed by square-shaped control volumes and the initial discrete domain is, generally, a unit square. In [Brandao, Gonzaga and Kischinhevsky, 2008], the scheme was applied with square-shaped finite-element approximations in the Laplace Equation Problem and, similarly, in [Brandao, Gonzaga and Kischinhevsky, 2009], it was applied in the Poisson Equation Problem. Furthermore, as long as a discrete place is refined, a graph node is substituted by a sub-graph that represents the new discrete places inserted. The data structure was especially designed to minimize the number of operations needed in the AMR process. Implementation of this AMR scheme allows flexibility in reaching neighbors with different levels of refinement. Since ALG was not conceived with any particular problem or geometry, its concepts can be applied to the study of several phenomena and classes of PDEs. To summarize, there are low storage requirements and low computational cost when representing an AMR.

1.3 Triangular meshes

Triangular meshes, or triangulations, are one of the most widely used representations for geometric models. A triangulation is a 2D simplicial complex, a simple structure with convenient combinatorial properties [Velho, Figueiredo and Gomes, 1999]. Simplicial meshes can be related to general meshes. Here, the meshes are related to simplicial complex structures (see [Velho, Figueiredo and Gomes, 1999]) in non-conform meshes since those are not a concern in finite volume discretizations.

A triangular mesh is said to be distorted (or bad quality) if it has triangles in which the ratio between the radius r of the incircle and the radius R of the cincumcircle of the triangle tends to zero ($\frac{r}{R} << 1$). Thus, generating a quality mesh is crucial when solving PDEs. This present work intends to generate quality meshes during the dynamic AMR process.

1.4 A graph-based adaptive triangular mesh refinement for finite volume approximations

Considering those aspects, this work follows the concepts of the graph data structure proposed by [Burgarelli, Kischinhevsky and Biezuner, 2006]; however, in this present work, the graph data structure represents a triangular mesh. Thus, the main novelty proposed here is that this technique uses a discrete adaptive mesh comprised of volumes with any triangular shape and the mesh is represented by a graph. An irregular mesh comprised of triangular volumes can be more appropriate near physical boundary regions and features of a problem with complex geometries. In relation to the total ordering, a Hamiltonian triangulation is generated with respect to the Hamiltonian ordering found. Additionally, linear-system solvers based on the minimization of functionals can be easily employed. Moreover, this technique operates all the required elements for AMR in the Finite Volume Method. Triangular meshes are shown in the experimental tests in order to exemplify this technique.

1.5 Brief introduction to the Finite Volume Method

The Finite Volume Method (FVM) has been applied to approximate complex problems in several areas of science and engineering, especially for environmental draining, prediction of scattering of pollutants in the atmosphere, water and earth, in aerodynamic problems, draining study over several surfaces, in petroleum reservoirs, in simulations to increase efficiency of recovery of petroleum in a reservoir, among others.

As stated before, the FVM is a method for representing and evaluating PDEs as algebraic equations and values are calculated at discrete places on a meshed geometry. *Finite volume* refers to the small volume surrounding each node point on a mesh. In the FVM, volume integrals in a PDE that contain a divergence term are converted to surface integrals using the Divergence Theorem [Leithold, 1994]. In Vector Calculus, the Divergence Theorem, also known as Gauss Theorem or Ostrogradsky-Gauss Theorem, is a result that links the divergence of a vector field to the value of surface integrals of the flow defined by the field. Those terms are then evaluated as fluxes at the surfaces of each finite volume. Because the flux entering a given volume is identical to that leaving the adjacent volume, these methods are conservative. Another advantage of the FVM is that it is easily formulated to allow unstructured meshes. The method is used in many computational fluid dynamics packages [Versteeg and Malalasekera, 1995].

In other words, the FVM is naturally based on the mass conservation laws and its application is simple and convenient. Classically, it is applied to a PDE in its conservative form, that is, with conservation laws. Such equations characteristically employ a divergent operator. Each equation is located in each control volume and the Divergence Theorem converts each equation to the integral of the flux along the boundary of the control volume. Moreover, integral conservation laws state that the total size rate of a substance with density ϕ in a specific finite volume is equal to the total flux of the substance through the finite volume boundary. Namely, the key concept used during the FVM formulation is the principle of conservation of a specific physical quantity expressed by governing equations over any finite volume.

The mesh generation with a proper generation of control volumes is required. This should be carefully considered in order to obtain an optimal order of convergence. The average flux along each edge of a control volume is then approximated by a special finite-difference scheme by using data in the neighboring cells. From the code implementation point of view, FVM is easier to code than other similar numerical methods. Appendix B presents a brief introduction to finite volume approximations by integrating the PDE in the conservative form on an elementar control volume.

To summarize, the flux domain is discretized in a set of non-overlapped control vol-

umes, which can be irregular in size and shape. Namely, similarly to other methods that obtain approximated equations from balance equations, the FVM consists of integrating the differential equation in a conservative form of the control volume set.

1.5.1 Convergence analysis

The reader is referred to [Arnold, 2001] for an introduction to numerical analysis. Basically, convergence analysis of the Finite Volume Method can be obtained from the framework of the Finite Element Method (FEM). Indeed, convergence can be analyzed directly in discrete norms, or function space norms can be used by constructing either an equivalent or an asymptotically equivalent FEM. Some authors obtained stability and convergence analysis from other forms, which do not use a FEM basis [Jeon and Sheen, 2005]. For example, [Bouche, Ghidaglia and Pascal, 2005] investigated the order of convergence of the upwind FVM for solving linear steady convection problem on a bounded domain with natural boundary conditions. In order to overcome the non-consistency in the finite difference sense of the scheme, those authors introduced a sequence that they called geometric correctors. It is associated with each finite volume and each constant convection vector. Under a local quasi-uniformity condition for the triangulation and if the continuous solution is regular enough, they first established a link between the convergence of the finite volume scheme and those geometric correctors. As a result, the study of that corrector in the case of uniformly refined triangular meshes in 2D led to the proof of the optimal order of convergence for the finite volume scheme.

After this brief introduction, Chapter 2 deals with the discretization schemes used in the FVM with irregular meshes. Chapter 3 presents the numerical solution adopted. Although informally presented, the algebraic operations in Chapter 3 are somewhat technical and may be omitted on a first reading. Chapter 4 explains the 4-triangles longest-side partition of triangular volumes, and also introduces the AMR scheme as well as discusses the graph data structure in detail, including the refinement/unrefinement process. Since the characterists in Chapter 4 are the main novelties proposed, it can be considered as the core of this work. Another important feature of this work is presented in Chapter 5. It briefly explains the space-filling curve employed to number the mesh nodes and gives details of the mesh total ordering. Chapter 6 discusses the experimental tests. Finally, Chapter 7 discusses some additional remarks and presents future directions as well.

Chapter 2

A review of finite volume schemes with triangular meshes

Approaches seek appropriate forms in order to fullfil the Finite Volume Method (FVM) requirements, e.g. to adequately establish an orthogonal segment between evaluation points and the outward normal vector, which is used in the Divergence Theorem.

Generally, there are two main schemes for finite volume discretizations in triangular meshes: cell-centered and vertex-centered. Both discretizations differ in the location of the control volumes in the mesh and the flux variable:

(i) In cell-centered schemes, flux quantities are stored in the finite volume centroid themselves and the mesh has simple geometry;

(ii) In vertex-centered schemes, flux variables are stored in the mesh vertices. Consequently, control volumes are comprised of sub-finite volumes, i.e. parts of finite volumes, which the vertex belongs.

2.1 A brief comparison between cell and vertex-centered control volume schemes

Vertex-centered schemes are first-order accurate on distorted grids. On Cartesian or on smooth grids, the vertex-centered schemes are second or higher-order accurate depending on the flux evaluation scheme. On the other hand, the discretization error of cell-centered schemes depends strongly on the smoothness of the grid. In general, a cell-centered scheme on triangular/tetrahedral grid leads to about two/six times more control volumes than a vertex-centered scheme. Hence, cell-centered schemes have more degrees (more unknowns) of freedom than the Median dual scheme (see 2.2.1). In addition, control volumes in cell-centered schemes are usually smaller than those in vertex-centered schemes. This suggests

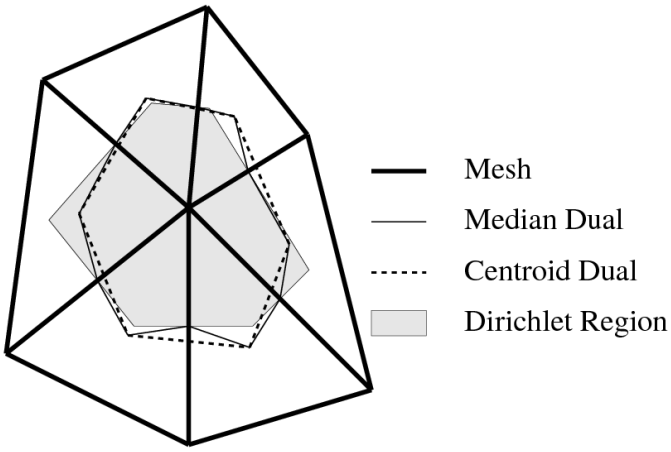

Figure 2.1: Triangulation duals.

that cell-centered schemes are more accurate than vertex-centered schemes. However, residual of cell-centered schemes results from much smaller number of fluxes compared to the Median dual scheme, which is a vertex-centered scheme (see Section 2.2.1). Boundary condition implementation in vertex-centered schemes requires additional logic in order to assure a consistent solution at boundary points, contrary to cell-centered schemes, which is simple [Yousuf, 2005]. As a result, there is no clear evidence about the better scheme.

There are publications that used both cell and vertex-centered strategies. Among these examples one finds [McManus et al., 2000], who showed a scalable strategy for the parallelization of multiphysics unstructured mesh-iterative codes on distributed-memory systems.

2.2 A review of vertex-centered control volume schemes

In general, there are two main triangular vertex-centered schemes: the Median dual scheme for general triangular meshes and Voronoi Diagrams (or the Dirichlet tessela-tion). Both create a dual mesh for determining the required quantities. In Fig. 2.1 (from [Barth, 1994]), edges and faces around the central vertex are formed by duals median segments, centroid segments, and by the Dirichlet Tesselation.

2.2.1 The Median dual scheme

One of the first formulations using triangular meshes for the generation of control vol-umes appeared in the end of 1970s and in the begining of 1980s, proposed by Baliga and Patankar [Schneider and Maliska, 2002]. The Finite Element Method (FEM) test func-tions were adapted to the finite-volume discretization. Figure 2.2a (from [Schneider and Maliska, 2002]) depicts a control volume according to this scheme.

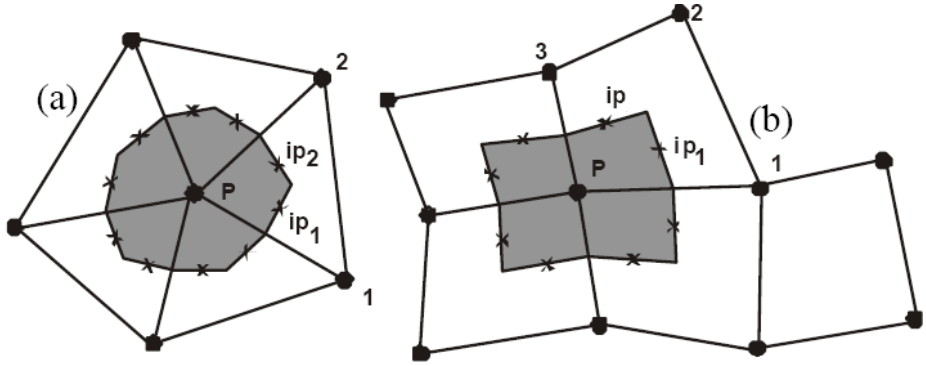

Figure 2.2: (a) A control volume of the Median dual scheme and; (b) a control volume from the scheme proposed by [Schneider and Raw, 1987].

The Median dual scheme can be used with general meshes. It plays a special role because one can show that using a specific numerical quadrature, the FEM method with linear elements and the FVM on median duals are equivalent [Barth and Ohlberger, 2004]. This enables convergence analysis directly from the FEM.

Although the Median dual scheme enables generic mesh usage, consider a Delaunay Tesselation (or triangulation), common in the FEM, where the control volumes are generated, depicted in Fig. 2.2a. The geometric center of the triangle P12 in Fig. 2.2a is joined to the mean point of its edges $\overline{P1}$ and $\overline{P2}$. Thus, the five triangular elements, similar to the element P12, form the control volume centered in P. Each triangular element contributes with two integration points, ip_1 and ip_2, in the balance conservation of the control volume centered in P.

Aiming towards maintaining the praticity in the mesh generation,
[Schneider and Raw, 1987] proposed to obtain control volumes similarly to the proposal in [Baliga and Patankar, 1980]. Their approach is based on a mesh formed by quadrangles similarly to the generalized coordinates [Schneider and Maliska, 2002]. The advance in the field given in [Schneider and Raw, 1987] was the coupling introduced in the equations in order to improve the convergence process. Figure 2.2b (from [Schneider and Maliska, 2002]) illustrates an elementar volume according to this scheme. The control volumes are obtained by joining the barycenter of the quadrangle P123 to the mean point of the edges. Quadrangular elements similar to the element P123 form the control volume centered in P. Furthermore, a local coordinate system is needed in order to implement the approximations. It provides more options to create the mesh and does not need to have a fixed number of points in the coordinate directions, what differs from the generalized coordinates [Schneider and Maliska, 2002]. Each element also contributes with two integration points, ip and ip_1, in the balance conservation of the control volume centered in P.

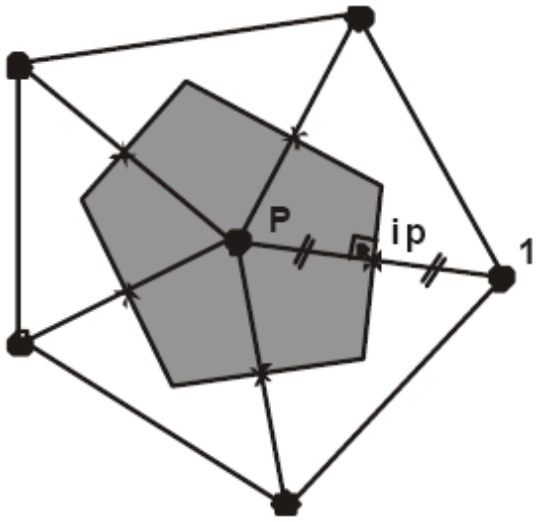

Figure 2.3: A Voronoi volume.

2.2.2 Voronoi Diagrams

Voronoi Diagrams are another important vertex-centered scheme for its simplicity in the formulation. Voronoi Diagrams take advantage that the edges that comprise a control volume are orthogonal to the segment between control volume centroids (of simplices). More precisely, Voronoi Diagrams (see Fig. 2.1) of a set of points P are convex regions in the plane, each region being the portion of the plane closer to one of the points of P than to any other point of P. Figure 2.3 (from [Schneider and Maliska, 2002]) represents a Voronoi volume. It is obtained from joinning the mean point of each edge to the adjacent triangle barycenters. Each segment contributes with one integration point ip in the balance conservation.

Voronoi Diagrams have a rich mathematical theory. According to [Barth, 1994], Voronoi Diagrams are believed to be one of the most fundamental constructs of computational geometry defined by discrete data. Voronoi Diagrams have been independently discovered in a wide variety of disciplines.

The Delaunay Tesselation of a point set is defined as the dual of the Voronoi Diagrams of the set. The 2D Delaunay Tesselation is formed by connecting two points if and only if their Voronoi regions have a common border segment. If no four or more points are cocircular, the vertices of the Voronoi Diagrams are circumcenters of the Delaunay triangles. This is true because vertices of the Voronoi represent locations are equidistant to three (or more) sites. Because edges of the Voronoi Diagrams are the *loci* of points equidistant to two sites, each edge of the Voronoi Diagrams is perpendicular to the corresponding edge of the Delaunay Tesselation. This duality extends straightforwardly to 3D [Barth, 1994].

In summary, Voronoi Diagrams use an elegant characteristic from the Delaunay Tes-

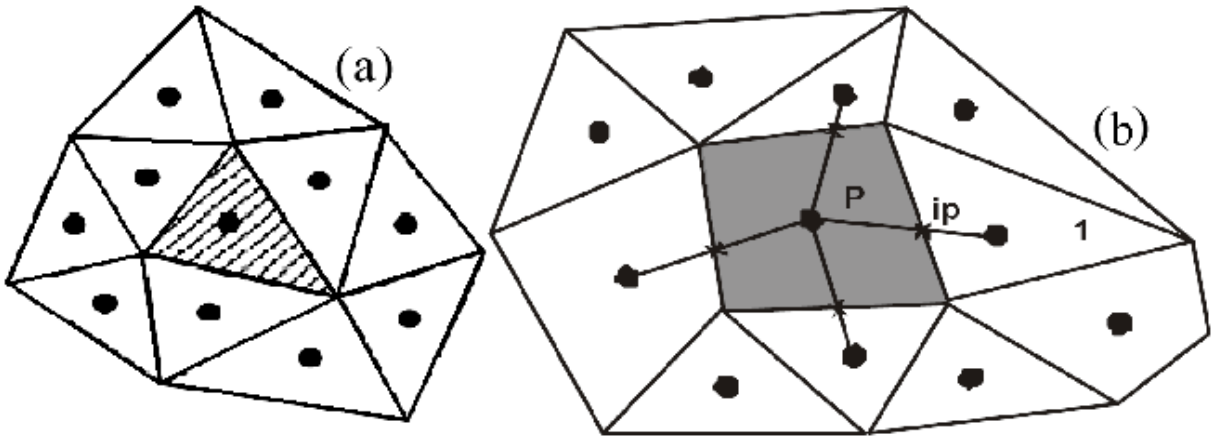

Figure 2.4: Example meshes with control volumes in the proper finite volume.

selation: Voronoi volumes have edges orthogonal to the line segment between adjacent vertices and the intersection point is on the mean point of the edge that connects the two vertices.

As examples of the wide research in Voronoi Diagrams, the following are mentioned: [Guibas and Stolfi, 1985] presented primitives for the manipulation of general subdivisions and the computation of Voronoi Diagrams; [Barth, 1994] presented aspects of unstructured grids and finite volume solvers for the Euler and Navier-Stokes Equations; [Shewchuk, 1999] presented aspects of the Delaunay mesh generation; [Ju, 2004] presented an algorithm for mesh generation; [Iske and Kaser, 2004] proposed a conservative semi-lagrangian advection on adaptive unstructured meshes; [Galante, 2006] applied paralell multigrid methods in the simulation of CFD problems.

2.3 A review of cell-centered control volume schemes

In this strategy, the control volumes are the cell themselves. For instance, Fig. 2.4a (from [Barth and Ohlberger, 2004]) sketches a triangular mesh with centroids in the proper finite volumes. Figure 2.4b (from [Schneider and Maliska, 2002]) sketches the discretization using arbitrary polygons. The finite volume approximations are performed on each edge of the polygon P. The integration point ip in Fig. 2.4a is not necessarily the mean point between the volumes P and 1 [Schneider and Maliska, 2002].

Many publications used this strategy. Examples are: [Turner and Ferguson, 1995] applied a hexagonal mesh in the numerical simulation of mass and heat transfer in porous media; a generalization of the discretization through convex arbitrary polygons was proposed in [Mathur and Murthy, 1997] employing a versatile mesh;
[Berger, Aftosmis and Melton, 1998] described their approach to computing accurate so-

lutions for time dependent fluid flows in complex geometry; [Pascal and Ghidaglia, 2001] introduced an algorithm for the discretization of second order elliptic operators in the context of finite volume schemes on unstructured meshes.

Two common techniques for simplified linear reconstruction include a Green-Gauss integration technique and the simplified least-squares technique, treated in the following sections.

2.3.1 Cell-centered Green-Gauss integration technique

A common technique for simplified linear reconstruction is the Green-Gauss linear integration technique reconstruction, where gradients are computed in specific integration points. Moreover, convective and diffusion terms are evaluated on all control volume edges.

As an example of article using this strategy, [Sachdev, Groth and Gottlieb, 2005] implemented a parallel adaptive mesh refinement (AMR) scheme for turbulent multi-phase rocket motor core flows.

2.3.2 Cell-centered least-squares schemes

The simplified least-square gradient reconstruction was proposed by Barth in 1991 [Barth and Ohlberger, 2004]. The principle of the least-squares reconstruction is to minimize the error in numerically approximating the integrals in the cell averages of the neighboring cells, which locally support the higher-order method [Bramkamp, Lamby and Muller, 2004]. An example of publication using this scheme is [Kobayashi, Pereira and Pereira, 1999], who presented a conservative finite volume second-order accurate projection method on hybrid unstructured grids in steady 2D incompressible viscous recirculating flows.

2.3.3 The Green-Gauss technique and the least-squares method

Some publications proposed hybrid approaches using the Green-Gauss technique and the least-squares method. Examples are: [Bramkamp, Ballmann and Muller, 2000] developed a flow solver employing local adaptation based on multiscale analysis on b-spline grids; [Bramkamp et al., 2003] described h-adaptive multiscale schemes for the compressible Navier-Stokes Equations with polyhedral discretization and Data Compression; [Northrup, 2004] implemented a parallel AMR scheme for predicting laminar diffusion flames; [Bramkamp, Lamby and Muller, 2004] presented an adaptive multiscale finite vol-

ume solver for unsteady and steady state flow computations; [Iaccarino and Ham, 2004] presented automatic mesh generation for LES in complex geometries.

2.4 A review of higher-order resolution schemes

First-order accurate schemes are strongly diffusive (numerical oscillations). For this reason, higher-order accurate schemes seek to avoid numerical oscillations. High-resolution schemes are used in the numerical solution of PDEs where high accuracy in the presence of shocks or discontinuities is required. They have the following properties: second or higher-order spatial accuracy is obtained in smooth parts of the solution; solutions are free from spurious oscillations or wiggles; high accuracy is obtained around shocks and discontinuities; the number of mesh points in parts of the discretized domain (containing the wave in a convective problem) is small compared with a first-order scheme with similar accuracy [Harten, 1983].

High-resolution schemes often use flux/slope limiters to limit the gradient around shocks or discontinuities. Some of those methods are:

i) Monotone Upstream centered Scheme for Conservation Laws (MUSCL) proposed by van Leer in 1979. Notice that functions between ordered sets are monotonic (or monotone, or even isotone) if they preserve the given order;

ii) Piecewise Parabolic Method (PPM) proposed by Colella and Woodward in 1981;

iii) Essentially Non-Oscillatory (ENO) scheme proposed by Harten in 1983;

vi) Quadratic reconstruction proposed by Barth in 1990;

v) H-box methods proposed by Berger and LeVeque in 1990;

vi) Frink's reconstruction proposed by Frink, Parikh and Pirzadeh in 1991;

vii) Cubic Interpolation Profile (CIP) proposed by Yabe, Aoki, Ishikawa and Wang in 1991;

viii) Weighted Essentially Non-Oscillatory (WENO) proposed by Liu, Osher and Chan in 1994;

ix) Cubic Interpolation with Volume/Area Coordinates (CIVA) proposed by Tanaka in 1999;

x) Arbitrary high order, using high-order DERivatives of polynomials (ADER) proposed by Toro, Millington and Nejad in 2001.

2.4.1 MUSCL

A general family of Total Variation Diminishing (TVD) discretizations is the Monotone Upstream-centered Scheme for Conservation Laws (MUSCL) discretization of van Leer, proposed in 1979. MUSCL is a second-order accurate extension of the scheme proposed by Godunov in 1959 [Barth and Ohlberger, 2004]. An example of publication is [Azevedo and Korzenowsk, 1999], who developed an unstructured grid solver for inert and reactive high speed flow simulations.

2.4.2 PPM

PPM is a higher-order extension of Godunov's method of a type first introduced by van Leer in his MUSCL algorithm. Examples of publications are: [Colella and Woodward, 1984] presented the PPM for Gas-Dynamical Simulations (1984); [Carpenter et al., 1989] applied the PPM to meteorological modeling.

2.4.3 ENO

ENO is also a higher-order accuracy scheme that follows the Godunov scheme dating back to Harten, who introduced the concept of ENO schemes for 1D conservation laws. Later, Harten and Chakravarthy in 1991, Abgrall in 1994, and Sonar in 1997 extended this finite volume formulation to unstructured triangular meshes. The basic idea of ENO schemes is to, firstly, select, for each control volume, a set of stencils comprising of neighbouring control volumes. Secondly, for each stencil, a recovery polynomial is computed, which interpolates the given cell averages on the control volumes in the stencil. Among the different recovery polynomials, the smoothest (i.e. least oscillatory) polynomial is finally selected, which constitutes the numerical solution of the hyperbolic conservation laws over its corresponding control volume. In this way, ENO schemes lead to finite volume discretizations of high-order space accuracy, provided that high-order reconstruction polynomials are utilized. Moreover, by the selection of smoothest polynomials, spurious oscillations can be avoided [Kaser and Iske, 2005].

2.4.4 WENO

In the more sophisticated WENO approach, the whole stencil set is used in order to construct, for a corresponding control volume, a weighted sum of reconstruction polynomials, each belonging to one stencil. Moreover, the weights are determined by a specific oscillation indicator, which measures the oscillation behaviour of each reconstruction

polynomial. WENO schemes show, in comparison with ENO schemes, superior convergence to steady-state solutions and higher-order accuracy, especially in smooth regions and around extrema of the solution. Friedrich in 1998, Hu and Shu in 1999, constructed WENO schemes on unstructured meshes [Kaser and Iske, 2005]. Examples of publications are: [Furst, 2006] applied a weighted least-square scheme for compressible flows; [Kaser and Iske, 2005] applied ADER schemes on adaptive triangular meshes for scalar conservation laws; [Pantano et al., 2007] presented a low numerical dissipation patch-based AMR scheme for large-eddy simulation of compressible flows.

2.4.5 Quadratic reconstruction

Quadratic reconstruction was firstly proposed by Barth in 1990 to vertex-centered control volumes [Barth and Ohlberger, 2004]. Due to a simple FVM scheme based on quadratic reconstruction would have great appeal, Delanaye and Essers (1997) developed a particular form of quadratic reconstruction for the cell-centered scheme, which is computationally more efficient than the Barth's method [Yousuf, 2005]. However, quadratic reconstruction is not an easy task on irregular meshes and still there are some problems: (i) ill conditioning of the quadratic reconstruction, which can occur on distorted (or "bad") meshes; (ii) ill conditioning of the quadratic reconstruction that can occur near physical boundaries due to the larger stencil used in quadratic reconstruction; and (iii) usually it requires curved elements on physical boundaries, otherwise the accuracy significantly deteriorates. Also, the issue of monotonicity via *limiting* becomes less robust. Examples of publications are: [Barth, 1993] presented developments in high-order k-exact reconstruction on unstructured meshes; [Yousuf, 2005] presented a 2D compressible viscous-flow solver on unstructured meshes with linear and quadratic reconstruction of convective fluxes.

2.4.6 *H*-box methods

The basic idea of the so-called h-box methods is to approximate the numerical fluxes at the interfaces of a small cell based on initial values specific over regions of length h, i.e. of the length of a regular grid cell. If this is done appropriately, the resulting method remains stable for time steps based on a CFL number appropriate for the regular part of the grid (Berger, Helzel, and LeVeque, 2002). Examples of publications are: [Berger, Helzel and LeVeque, 2003] presented the approximation of hyperbolic conservation laws; [Helzel, Berger and LeVeque, 2005] considered a high-resolution rotated grid method for conservation laws with embedded geometries.

2.4.7 Frinck's reconstruction

Frink's reconstruction is an extension for 3D problems of the scheme proposed by Barth and Jaspersen in 1989 [Barth and Ohlberger, 2004]. Examples of publications are:
[Frink, 1994] improved the scheme by a pseudo-Laplacian weighted averaging algorithm; [Frink, 1996] presented an assesment of the method for predicting 3D turbulent viscous flows; [Pandya and Frink, 2004] presented an agglomeration multigrid Euler/Navier-Stokes solver.

2.4.8 CIP

The Cubic Interpolation Profile Method improves the accuracy of the numerical solution by using the spatial derivatives as prognostic variables. It is a third-order scheme in terms of spatial accuracy. CIP reconstructs advection by cubic interpolation based on a rectangular parallepiped mesh [Tanaka, Yamasaki and Tagushi, 2004]. Examples of publications are: [Ozawa and Tanahashi, 2005] solved the Boltzmann equation for the spatial derivatives as in the CIP method; [Tanaka, Yamasaki and Tagushi, 2004] performed the discretization by the FVM combined with the CIP for universal capability to handling compressible and incompressible fluid flows.

2.4.9 CIVA

Since CIP was implemented in an unstructured and fixed (Eulerian) rectangular mesh-based finite volume solver, [Tanaka, 1999] developed the Cubic-interpolation with Volume/Area Coordinates, a highly accurate interpolation method for meshfree flow simulations. Moreover, CIVA is an extension of the CIP to a triangular or tetrahedral mesh system. In CIVA, interpolation is based on n-simplex directions (triangle and tetrahedron), where n means the spatial dimension of Euclidian space. The method was originally developed for meshfree approaches, such as the Lagrangian (particle) approach and the gridless method in order to improve the accuracy and stability. Barycentric coordinates are used for the formulation of the CIVA, instead of Cartesian coordinates. Barycentric coordinates are called volume coordinates in 3D and area coordinates in 2D. Through CIVA, achieving highly accurate interpolation based on n-simplex directions as in the case of meshfree approaches [Tanaka, Yamasaki and Tagushi, 2004] is possible. Examples of articles are: [Ozawa and Tanahashi, 2005] applied CIVA for the Discrete Boltzmann Equation; [Tanaka, Yamasaki and Tagushi, 2004] presented an accurate and robust fluid analysis using CIVA.

2.4.10 ADER

Toro, Millington and Nejad proposed an explicit one-step finite volume scheme in 2001. It is an arbitrary high-order scheme, using high-order derivatives of polynomials. The finite volume discretization of ADER combines high-order polynomial reconstruction from cell averages with high-order flux evaluation. The latter is done by solving Generalized Riemann Problems across the cell interfaces, i.e. boundaries of adjacent control volumes. Therefore, the finite-volume ADER scheme of the work can be viewed as a generalization of the classical first-order Godunov scheme to arbitrary high orders [Kaser and Iske, 2005].

2.4.11 Other schemes

There are publications that seek second-order accuracy for specific classes of problems. Other publications use both approaches for vertex and cell-centered control volumes. Examples are: [Kim and Choi, 2000] presented a second-order time-accurate numerical solution to unsteady incompressible flow problems; [McManus et al., 2000] presented a scalable strategy for the parallelization of multiphysics irregular mesh-iterative codes on distributed-memory systems; [Bertolazzi and Manzini, 2008] presented polynomial reconstructions and limiting strategies in finite volume approximations;
[Barth and Ohlberger, 2004] presented a complete FVM foundation and analysis of some of the earlier cited schemes. Besides, there are several other publications that use the FVM with irregular meshes either applying to specific problems or proposing specific solutions.

Chapter 3

Description of a discrete formulation based on the Finite Volume Method

The main objetive of this work is to produce a computational analysis of implementing the concepts of the Autonomous Leaves Graph in a triangular mesh. Thus, choosing an appropriate scheme for numerical integration in order to introduce those concepts in triangular meshes is critical.

Higher-order accurate schemes seek to avoid numerical oscillation; however, they are computationally expensive, such as Frinck's reconstruction, PPM and ENO/WENO schemes, as well as in the generation of a dual mesh of the quadratic reconstruction. MUSCL demands to include some non-linearity, namely flux limiters. The design and implementation of limiters and especially multidimensional limiters is an active field of research. Simplified least-square gradient reconstruction requires solving small linear systems to each average cell, such in the FEM.

Using a Median dual or centroid dual mesh and a Green-Gauss integration around the control volume is equivalent to a Galerkin discretization of the gradient on linear elements which is known to be linearity preserving. Median dual may be computationally more expensive than other schemes because the dual-mesh generation and also because the computation of small linear systems required for each control volume (elementar systems), in the same form that the Finite Element Method (FEM) does. Besides, producing ghost triangular volumes in the border for determination through Median dual scheme would require evaluation of the dependent variable of the partial differential equation (PDE) also in the vertices that early existed. This would increase the linear system in n unknowns, where n is the number of the triangular volumes in the mesh.

Voronoi Diagrams are attractive for its simplicity in evaluating the dependent variable and its gradients on the control volume edges; though, it requires the strict dual-mesh Delaunay Tesselation and hybrid meshes are not allowed. Besides, points may be determined

outside of the volumes [Barth, 1994].

Determining the PDE dependent variable in the triangular volume centroids has an advantage similar to that of Median dual, that is, using finite volume centroids for generating control volumes, one can assure that the mesh will not have exterior points in the mesh. In addition, general meshes can be used. Differently from control volumes created, for instance, from Voronoi Diagrams, whose centroids are circumcenters and hence, the centroids can be exterior to the triangular volumes (because it requires Delaunay Tesselation).

Although the simplified least-square gradient reconstruction gives better results than the Green-Gauss reconstruction for distorted meshes, the simplified least-square gradient reconstruction is computationally more expensive than Green-Gauss reconstruction.

Employing a linear integration in a cell-centered approach demands an extra computational effort in order to evaluate the flux on the vertices of the edge between control volumes. Though, this present approach, which is mainly characterized by a diamond cell, determinates the gradients, namely, diffusion and viscous terms, as easy as a vertex-centered dual scheme; however, without generating that dual mesh. Therefore, seeking low computational cost, this work tailors a scheme based on the work of [Schneider and Maliska, 2002].

Since the original ALG [Burgarelli, Kischinhevsky and Biezuner, 2006] was proposed in a cell-centered scheme, adapting it to a vertex-centered approach would demand an approach similar to the proposed by [Brandao, Gonzaga and Kischinhevsky, 2008] and [Brandao, Gonzaga and Kischinhevsky, 2009], i.e. an extra linked list is implemented in order to number the mesh vertices, increasing the computational effort. In other words, the data structure to number the volumes would have to be adapted to a vertex-centered scheme, whereas it is straightforward in a cell-centered scheme. For that purpose, this work implements a cell-centered approach tailoring the vertex-center approach of [Schneider and Maliska, 2002], shown in the next section.

To summarize, taking in account the above characteristics, this work determines the PDE dependent variable at the centroids of the proper triangular volumes, namely, using a cell-centered scheme. This work determines the barycenter as the centroid of the triangle. The barycenter is the gravity and mass center of the triangle.

3.1 A finite volume approximation

The discretization by elements, based on the FEM, eases the modeling of the approximated equations and normalizes the computational implementation, resulting in a better

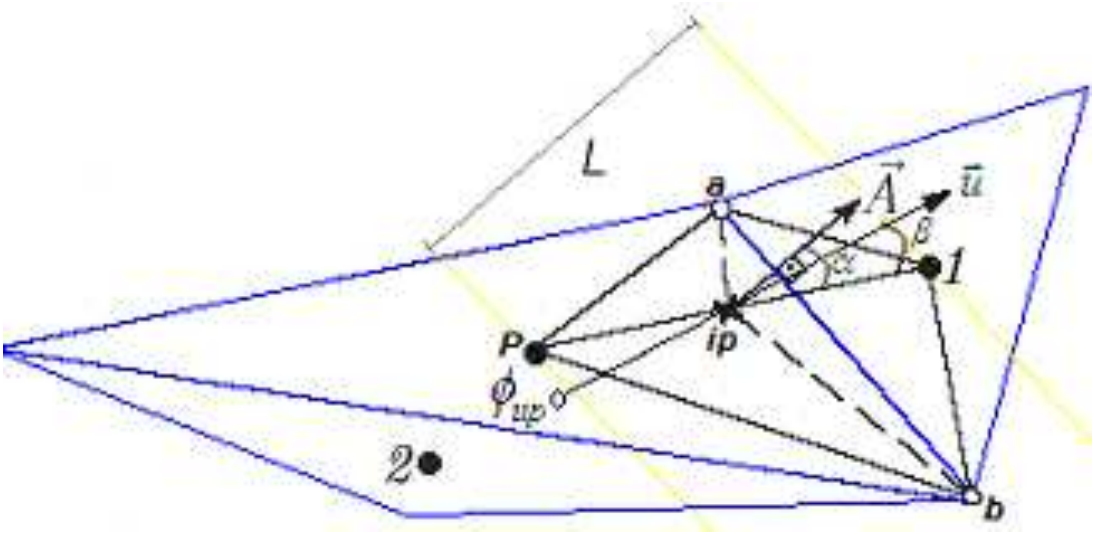

Figure 3.1: Three neighbor volumes, whose centroids are P, *1* and *2*, whereas *ip* is the integration point between volumes P and *1*.

accuracy in the solution. The arbitrary discretization of [Schneider and Maliska, 2002] consists of obtaining the control volume from a set of diamond elements limited by two centers of volumes and one area, where the equations are integrated. The element $Pa1b$ in Fig. 3.1 links the volumes centered in P and *1* through the segment $\overline{P1}$.

Figure 3.1 sketches three volumes, whose centroids are indicated by points P, *1* and *2*. The edge between volumes P and *1* is limited by points a and b. The integration point between those volumes is indicated by point *ip*, and the vector area is indicated by $\vec{A}$.

Thus, each element $Pa1b$ has one integration point. The position of the integration point in each element is fundamental in order to minimize the numerical error introduced by the approximation. Points a and b of Fig. 3.1 determine the effective area where flow is changed in element $Pa1b$, and also is the addition of the vector areas of the segments $\overline{aip}$ and $\overline{ipb}$, indicated by the vector $\vec{A}$ over the integration point *ip*. Vector $\vec{A}$ is not, necessarily, parallel to $\overline{P1}$. Such area can be applied at an integration point *ip* that is not in the intersection of the segments $\overline{ab}$ and $\overline{P1}$.

According to [Schneider and Maliska, 2002], the effetive area, where the flux is changed, does not depend on the position of the integration point. Consequently, this value can be $\frac{1}{2}Pa1b$ when using source terms involving a sub-volume quantity, i.e. the computation of sub-volumes is not needed. Thus, the integration point is the mean point of the segment $\overline{P1}$.

[Schneider and Maliska, 2002] showed a formulation applied to an evolutionary convective-diffusive problem, whose velocity field $\vec{u}$ is known and the concentration ϕ evolves

$$\rho\frac{\partial\phi}{\partial t} + \vec{u}\rho\nabla\cdot\phi = \Gamma^{\phi}\nabla^2\phi + S^{\phi}, \tag{3.1}$$

where Γ^{ϕ} is the diffusion coefficient, S^{ϕ} is the source term, ρ is the specific fluid mass and t is time.

Following the Finite Volume Method basic formulation for non-uniform meshes, the integration is performed over the volume named P. Applying the Divergence Theorem, with the source term lineariazed, numerical integration in time and space yield

$$M_P^n\phi_P^n - M_P^{n-1}\phi_P^{n-1} + \triangle t\sum_{ip=1}^{3}[\rho(\vec{u}\cdot\vec{A})_{ip}\phi_{ip} - \Gamma^{\phi}(\overrightarrow{\nabla\phi_P}.\overrightarrow{A})_{ip} - (S_P\phi_{ip} + S_C)\frac{\triangle V_{Pa1b}}{2}] = 0, \tag{3.2}$$

where $M_P = \rho\triangle V_P$ is the mass contained in the control volume, $\vec{A}$ is the vector area of each face and $\triangle V_P$ is the volume area. When the quantity parcels of all elements are added and boundary conditions applied, there is a conservative algebraic equation of volume P, connected to its neighbors. Applying this scheme to all volumes results in a system of algebraic equations. When it is solved, the values ϕ in all centroids that comprise the discretization are obtained. The reader is referred to [Schneider and Maliska, 2002] for details.

3.1.1 Linear interpolation of the convective term

An interpolation function evaluates the value of a generic property ϕ on the control volume edge. Differencing schemes for the linearization of convective terms are employed, i.e. the discretization of convected quantities. Early attempts to solve advection-diffusion problems applied the Central Differencing scheme (CDS). However, for problems with predominant convection, solutions exhibited non-physical behavior. Those issues initiated the development of a multitude of differencing schemes. Some of these are:

i) CDS is the most straightforward discretization of the convected variable, since it simply follows the linear interpolation idea. In terms of a Taylor-series expansion, CDS is second-order accurate, but it is rarely used due to its conditional stability [Madsen, 1998]. Oscillations in numerical solutions lead to upstream propagation of any disturbance. In Fig. 3.1, through a Taylor-series expansion around ip integration point, ϕ_P and ϕ_1 values can be calculated by

$$\phi_P = \phi_{ip} - \frac{\partial\phi}{\partial\overline{n}}|_{ip}(\frac{L}{2}) + \frac{\partial^2\phi}{\partial\overline{n}^2}|_{ip}\frac{(\frac{L}{2})^2}{2} - ... + ... \ , \tag{3.3}$$

$$\phi_1 = \phi_{ip} + \frac{\partial \phi}{\partial \overline{n}}|_{ip}(\frac{L}{2}) + \frac{\partial^2 \phi}{\partial \overline{n}^2}|_{ip}\frac{(\frac{L}{2})^2}{2} + ... + ... \ . \tag{3.4}$$

In CDS, the value ϕ_{ip} is given by adding Eqs. 3.3 and 3.4, assuming second-order accuracy

$$\phi_{ip} = \frac{\phi_P + \phi_1}{2}. \tag{3.5}$$

ii) Upwind Differencing Scheme (UDS) is a well-known remedy for the difficulties encountered in CDS. It consists of setting the cell-edge value equal to the nearest cell-center value in the upstream direction. UDS is only first-order accurate but still an improvement over CDS since it avoids disturbances in the upstream propagation of the flux. The low accuracy of the relatively crude UDS is often interpreted as causing excessive numerical diffusion, and makes it quite common to apply higher-order upwind schemes including more upstream points for the interpolation of ϕ_{ip}. In UDS, the value of ϕ_{ip} is given by adding Eqs. 3.3 and 3.4, assuming first-order accuracy:

$$\phi_{ip} = \phi_P \ for \ cos\beta > 0, \tag{3.6}$$

$$\phi_{ip} = \phi_1 \ for \ cos\beta < 0, \tag{3.7}$$

where β is the angle of the velocity vector $\vec{u}$ with segment $\overline{P1}$ in Fig. 3.1.

iii) Weighted Upwind Differencing Scheme is a combination of CDS and UDS using weights;

iv) Hybrid Differencing Scheme is also a combination of CDS and UDS, which was suggested by Spalding in 1972 [Madsen, 1998];

v) A quick scheme was proposed in 1979 [Versteeg and Malalasekera, 1995]. It performs the interpolation by fitting a parabola through the two upstream points and the downstream point nearest to the edge;

vi) Higher-order accurate schemes for interpolating ϕ in the interface of the control volume use information from two or more points in the upstream direction. An example is Skew Upwind Differencing Scheme, which interpolates values on the edges using two points of the flow. [Schneider and Maliska, 2002] cited

$$\phi_{ip} = \frac{1 + cos\ \beta}{2}\phi_P + \frac{1 - cos\ \beta}{2}\phi_1, \tag{3.8}$$

where β is the angle between segment $\overline{P1}$ and $\vec{u}$.

The literature still presents other schemes and for each case, options should be carefully studied in order to choose the best one. Usually, one implements some schemes for choosing the scheme that fits the PDE being approximated. In this work, the UDS was applied.

3.1.2 Treatment of the diffusion term

The dot product between the gradient vector of the PDE dependent variable and the vector area of Eq. 3.2 was adopted from the proposal presented in [Schneider and Maliska, 2002]. The authors determined an expression to the gradient of ϕ, described in the Cartesian coordinates (x, y)

$$\phi(x, y) = N_P \Phi_P + N_1 \Phi_1, \tag{3.9}$$

where Φ_P and Φ_1 are values of ϕ stored in the respective vertices of the element $Pa1b$, and N_P and N_1 are weighted (shape) functions from a coordinate transformation. Equation 3.9 is written in two coordinates in order to easily obtain the derivative expressions of ϕ in x and y to posteriorly compose the gradient vector. Therefore, Eq. 3.9 is a 2D interpolation function with only two points for its construction.

[Schneider and Maliska, 2002] built an equation of plane in the form of a 2D slope between values Φ_P and Φ_1. This function presents maximum gradient in the direction of the vector area and null gradient in the direction of the segment $\overline{ab}$ in Fig. 3.1. The third point is obtained, for example, passing the slope perpendicularly through the vector area. Such function yields

$$N_P = \frac{(y_b - y_a)x + (x_a - x_b)y + (x_b - x_a)y_1 + (y_a - y_b)x_1}{LM\ cos\ \alpha}, \tag{3.10}$$

$$N_1 = \frac{(y_a - y_b)x + (x_b - x_a)y + (y_b - y_a)x_P + (x_a - x_b)y_P}{LM\ cos\ \alpha}, \tag{3.11}$$

where

$$LM\ cos\ \alpha = (x_1 - x_P)(y_a - y_b) + (y_1 - y_P)(x_b - x_a) \tag{3.12}$$

where M is the area, α is the angle between $\vec{A}$ and $\overline{P1}$, and L is the distance of points P and 1 in the direction of the segment $\overline{ab}$, represented in Fig. 3.1.

Afterwards, in order to obtain the gradient vector of ϕ, Eq. 3.9 is derived with respect to *(x,y)*

$$\overrightarrow{\nabla\phi} = \overrightarrow{\nabla N_P}\Phi_P + \overrightarrow{\nabla N_1}\Phi_1. \tag{3.13}$$

Substituting Eqs. 3.10 and 3.11 in Eq. 3.13 results in

$$\overrightarrow{\nabla\phi} = (y_a - y_b, x_b - x_a)\frac{(\Phi_1 - \Phi_P)}{LM\ cos\ \alpha}. \tag{3.14}$$

Although Eqs. 3.10 and 3.11, and consequently Eq. 3.14, are different from the ones presented in [Schneider and Maliska, 2002], the idea is from those authors and here the third point of the equation of plane is obtained passing the slope perpendicularly through the vector area. Thus, the dot product of the gradient by the vector area (its module is equal to the elemental area) of Eq. 3.2 is

$$\overrightarrow{\nabla\phi}\cdot\vec{A} = \frac{M}{L\ cos\ \alpha}(\Phi_1 - \Phi_P). \tag{3.15}$$

The scheme presented by [Schneider and Maliska, 2002] can be simplified. From Eq. 3.12, one can write

$$D = (x_1 - x_P)(y_a - y_b) + (y_1 - y_P)(x_b - x_a) = LM\ cos\ \alpha \tag{3.16}$$

or

$$M = \frac{D}{L\ cos\ \alpha}. \tag{3.17}$$

Substituting $L = l_{\overline{P1}}\ cos\ \alpha$, where $l_{\overline{P1}}$ is the distance between points P and *1* (see Fig. 3.1), and substituting Eq. 3.17 in Eq. 3.15 yield

$$\overrightarrow{\nabla\phi}\cdot\vec{A} = \frac{D(\Phi_1 - \Phi_P)}{(l_{\overline{P1}}\ cos^2\alpha)^2}. \tag{3.18}$$

Since point *1* is on the right side of point P, i.e. $x_1 > x_P$ if they are at the same horizontal coordinate, vector $\vec{c} = (x_1 - x_P, y_1 - y_P)$ is generated from segment $\overline{P1}$. In addition, since $cos\ \alpha$ is given by the inner product between $\vec{c} = (x_c, y_c)$ and $\vec{A} = (x_A, y_A) = (y_b - y_a, x_a - x_b)$ and the distance of a segment is given by the Euclidian norm, calculations of square root are eliminated in

$$\overrightarrow{\nabla\phi} \cdot \vec{A} = \frac{(x_1 - x_P)(y_a - y_b) + (y_1 - y_P)(x_b - x_a)}{[(x_1 - x_P)^2 + (y_1 - y_P)^2](\frac{(x_c x_A + y_c y_A)^2}{(x_c^2 + y_c^2)(x_A^2 + y_A^2)})^2}(\Phi_1 - \Phi_P). \tag{3.19}$$

According to [Schneider and Maliska, 2002], the calculated flux is subjected to two errors: one of order L^2 because of the linear approximation of the derivative (as usual, L^2 is a space of 2-power integrable functions, and corresponding sequence spaces [Moura, 2002]); and another error involving the cosines of the angles between the real flux and the vector area with the segment $\overline{P1}$, which compromises the conditioning of the matrix associated with the resulting linear system. However, this present work intends to improve the mesh quality by the adopted refinement scheme.

Chapter 4

A graph-based adaptive mesh refinement technique with triangular cell-centered volumes

Using a graph for representing a triangular mesh is not a novelty. [Arkin et al., 1994] considered several issues related to paths on the dual graph of general triangular meshes. [Speckmann and Snoeyink, 1997] built triangle strips based on the dual graph of a static triangulation. [Velho, Figueiredo and Gomes, 1999] also described a dual graph for triangular mesh in order to build triangle strips by heuristics. These authors considered several issues related to paths on the dual graph of general triangular meshes. Besides, [Velho, Figueiredo and Gomes, 1999] proposed an adaptive mesh refinement (AMR) scheme using a dual graph in an approach similar to the one showed in Fig. 4.1d in order to accelerate rendering and produce better results in geometry compression. They divided the triangle in five ways, depending on the edge to be refined. On the other hand, this present work applies a 4-triangles longest-side partition (4TLSP) scheme following the Autonomous Leaves Graph (ALG) concepts. The reader is referred to [Burgarelli, Kischinhevsky and Biezuner, 2006] for a further reading about ALG.

To summarize, this present work proposes a graph-based AMR scheme that provides all the requirements for solution of partial differential equations (PDEs) based on the Finite Volume Method (FVM) with cell-centered triangular volumes through an adapted FVM scheme ([Schneider and Maliska, 2002]) in order to reduce the computational cost in solving PDEs.

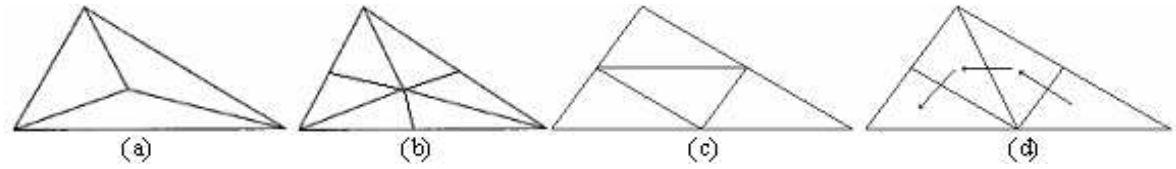

Figure 4.1: (a) Refinement by simple centroid insertion; (b) Refinement by centroid insertion and adding midpoints of the three edges; (c) Refinement by adding the three midpoints; and (d) Refinement by successive bisections.

4.1 The 4-triangles longest-side partition of volumes

In relation to triangle partition, the essential is that no angle be too close to 0 or π. In other words, triangulations which minimize the maximun angle (or maximize the minimum angle) are more desirable.

There are several possibilities of refining a triangle. For instance, the *intuitive* centroid insertions in Figs. 4.1a-b rapidly deteriorate the quality of triangulations (especially along boundaries), even when global refinement is performed [Rivara and Iribarren, 1996]. Besides, Fig. 4.1a is not compliant with the ALG scheme of refinement because it does not divide an edge into two new segments. Another option, shown in Fig. 4.1c, is to insert a triangle formed by adding the three midpoints generating four new triangles. This option maintains the quality of the original triangle. Moreover, the scheme showed in Fig. 4.1c could be used with initial equilateral triangles. However, [Velho, Figueiredo and Gomes, 1999] revealed that the approach showed in Fig. 4.1c does not produce a generalized sequential triangulation (see Chapter 5), whereas the approach showed in Fig. 4.1d does. Other options in order to refine triangles were presented by [Márquez et al., 2008], where the triangle-partition schemes presented do not generate triangle strip sequences and the 7T-Delaunay partition scheme is $O(N\ lg\ N)$.

4.1.1 The 4-triangles longest-side partition scheme

An interface orthogonal to the segment between two centroids facilitates finite volume approximations. Namely, it improves its accuracy and reduces the computational effort. In fact, an approach that improves the quality of triangles must be sought. A fourth option is to divide the triangle through successive bisections, as shown in Fig. 4.1d. This scheme enables the use of any shape of triangles. Thus, bisection divides the triangular area exactly in half.

Since in this work the triangle partition of Fig. 4.1d was considered the most appropriate, a review of the literature was performed related to it. Fortunately, there is a number of works in this field. Rosenberg and Stenger demonstrated in 1975 that the angles of

a bisected triangle t_j do not go toward zero as $j \to \infty$ [Rivara, 1984b]. [Stynes, 1980] demonstrated that the shape regularity of the triangles is improved as the method proceeds, i.e. the domain tends to be covered by triangles which are approximately equilateral in a certain sense. As long as triangles are bisected only across their longest edge, one can bound the maximum and minimum angles of the resulting triangles independently of the number of times the resulting triangles are bisected. In other words, the author proved that the largest resulting angle is bounded away from *0* and π. Furthermore, some angles of the next refinement phase tend to go toward $\frac{\pi}{3}$ as $k \to \infty$, where k is the number of refinements. Namely, asymptotically, the refined triangles tend to be equilateral. Hence, the edges tend to be orthogonal to the segment between neighbor centroids facilitating the finite volume resolution, improving its accuracy and reducing the computational effort.

In relation to the 4-triangles longest-side partition (4TLSP), each resulting triangle has one-fourth of the area of the original triangulation [Rivara, 1984a].

Definition. The 4TLSP is obtained by joining the midpoint of the longest side with the opposite vertex and the midpoints of the two remaining sides [Rivara and Iribarren, 1996] (see Fig. 4.1d).

The 4TLSP was firstly proposed by [Rivara, 1984b]. [Rivara, 1984c] demonstrated that provided a triangle and its descendants are repeatedly bisected across their longest edges, all angles in subsequent refined triangulations are greater than or equal to one-half the smallest angle in the original triangulation. Moreover, the author adapted the works of Rosenberg and Stenger from 1975 and [Stynes, 1980]. [Rivara, 1984a] used the 4TLSP scheme or other two ones in order to conform finite-element meshes. By means of simple geometrical arguments, [Rivara and Iribarren, 1996] showed that the iterative partition of obtuse triangles systematically improves the triangles while they remain obtuse in the following sense: the sequence of smallest angles monotonically increases while the sequence of largest angles monotonically decreases in an amount (at least) equal to the smallest angle of each iteration. This is true for at least two triangles since there are two classes of similar triangles. Besides, [Rivara and Iribarren, 1996] claimed that pure longest-side bisection-based algorithms are the most suitable ones in two important contexts: (1) in the adaptive refinement/unrefinement of triangulations such as needed in complex time-dependent problems; (2) in the practical use of multigrid numerical methods over irregular meshes (since the algorithms guarantee the construction of nested triangulations). Moreover, [Rivara and Iribarren, 1996] defined that, since the first 4TLSP of any triangle introduces new sides parallel to the sides of the original one, the following results hold:

Propositions. (a) The first 4TLSP of any triangle produces two triangles similar to original one and two (potentially) new similar triangles. (b) The iterative 4TLSP of any

triangle introduces (at most) one new distinct (up to similarity) triangle for iteration.

A survey of partition schemes was presented by [Jones and Plassmann, 1997], including the 4TLSP scheme. Those researches are in the context of finite-element meshes. On the other hand, [Velho, Figueiredo and Gomes, 1999] presented the 4TLSP scheme as one of five ways of partition schemes in the context of rendering and compression of images. In addition, these authors explained that the partition showed in Fig. 4.1d is a better choice than the others in relation to accelerated rendering and geometric compression. Notice that the refinement process of Fig. 4.1d produces a triangular sequence that follows the path of a Sierpinski-like space-filling Curve, which is a fractal curve.

To summarize, the scheme in Fig. 4.1d locally introduces new smaller triangles that are similar to the original triangle, but improved. Among those options, in the context of this present work, the partition scheme showed in Fig. 4.1d is the most suitable for defining a sequential refinement scheme. Therefore, this work implements the 4TLSP scheme.

4.2 A graph-based AMR representation of triangular cell-centered volumes

In this present work, a graph explicitly provides all the relations among volumes. The nodes correspond to the triangular volumes and two nodes are connected if their associated triangles have a common edge. Thus, when a triangular volume is refined, an original volume node is substituted by a new sub-graph comprised of a pack with four volume nodes and three transition nodes. Transition nodes indicate the refinement level of the volume in relation to their neighbor volumes. Such nested refinement process allows an unrefinement that may be needed, i.e. provided a new pack is created, the previous stages can be easily reached. This refinement process is depicted in Fig. 4.2. Suppose that the single graph node represents the triangular volume on the leftmost side of Fig. 4.2. This volume is refined and the single graph node is substituted by the sub-graph shown in the rightmost side of Fig. 4.2. In other words, the rightmost sub-graph is a pack that regenerates the original volume node sketched on its left-hand side.

The graph on the right-hand side of Fig. 4.2 illustrates two types of linked nodes: volume nodes (black circles) and transition nodes (white circles). Volume nodes represent each triangular volume. Graph node pointers are depicted by lines in Fig. 4.2. Volume nodes point to transition nodes by pairs. By its turn, each transition node also points to volume nodes in a sub-graph pack. Each node has three pointers. The pointer that is not used in transition nodes in boundary receives a null pointer.

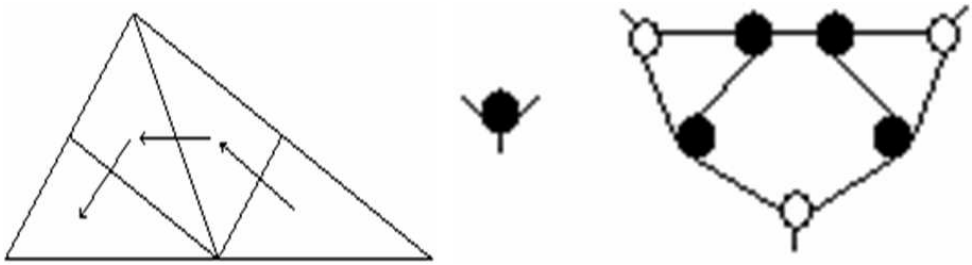

Figure 4.2: A triangular volume refined, the single graph node that represented the original triangular volume and on the mostright side a sub-graph created after a refinement of the triangular volume, respectively. The opposite is performed in the unrefinement process.

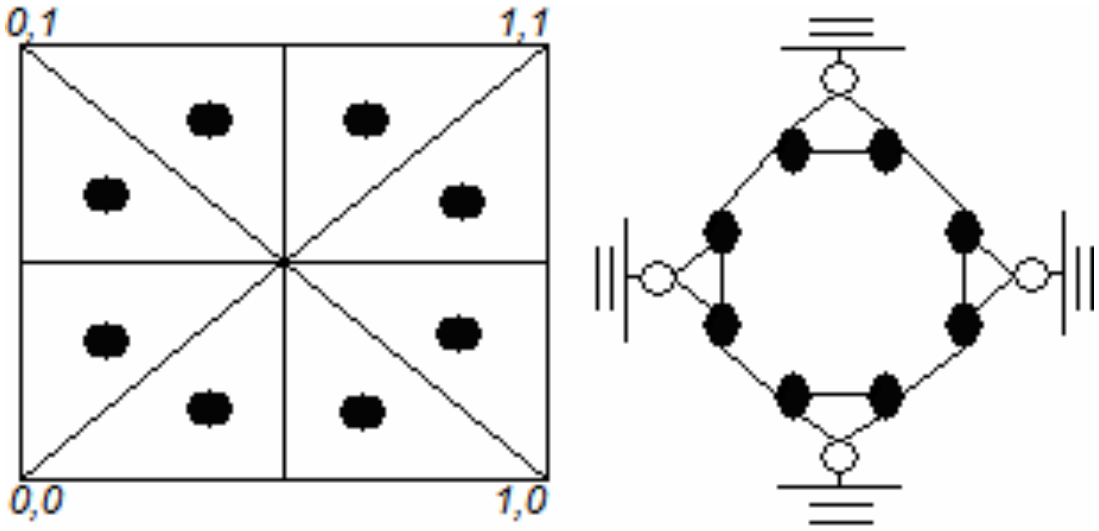

Figure 4.3: Unit square as the problem domain; and links of the graph data structure - nodes represent the refinement of level 0.

Figure 4.3 depicts an example of an initial discretization. Figure 4.3 also depicts a graph that represents this initial discretization. The barycenter of the triangle is represented by a black dot in Fig. 4.3. White circles in the graph of Fig. 4.3 link to the boundary of the domain.

Original nodes are deleted in the local refinement of each triangular volume for memory savings. Only volume nodes that represent the four new triangular volumes, which are the generated children, and the three required transition nodes in the pack are stored. In addition, the level of the volume nodes of the new pack is increased of 1.

In Fig. 4.4, a volume is refined and the resulting graph is shown on its right-hand side. Boundaries of the domain in Fig. 4.4 are omitted for clarity.

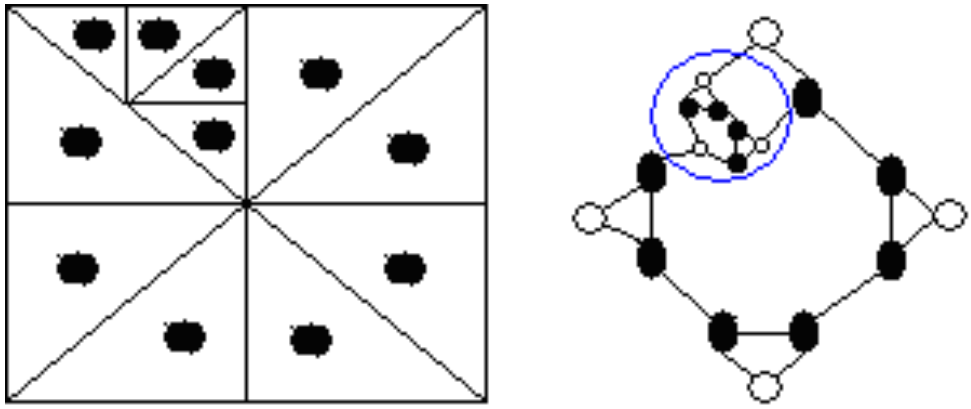

Figure 4.4: Refinement example with triangular volumes; and graph with transition and volume nodes that forms the scheme of this triangular volume refinement, respectively.

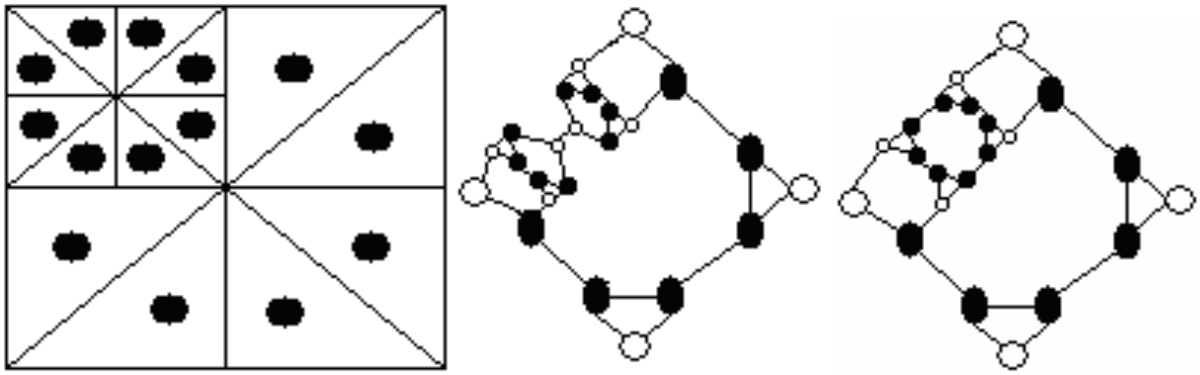

Figure 4.5: Refinement of another volume depicted in Fig. 4.4, its graph representation, and graph simplification, respectively.

4.3 Graph simplification for volumes of the same level

Figure 4.5 illustrates another refinement of the discretized domain of Fig. 4.4, its graph representation, and the simplification of the graph, respectively. More precisely, since volumes have the same refinement level, namely, their vertices are coincident; the mostright hand-side graph of Fig. 4.5 illustrates its simplification.

4.4 Generalization of the refinement scheme

During time evolution, the mesh is refined and a linear system must be solved each time step. As long as the difference in scale of flux between two neighbor volumes is large, one could choose to refine the volume in more than four new volumes. Table I describes the options.

Table I: Number of transition and mesh nodes in a pack.			
n	L	Transition nodes	Volume nodes
1	+1	3	4
2	+2	9	16
3	+3	21	64
4	+4	45	256
5	+5	93	1024
...	...	...	...
n	+n	$3 \cdot \sum_{i=0}^{n-1} 2^i$	4^n

In Table I, column n means in how much levels (L) the pack is refined. For example, the experimental tests presented in this work show *n=1*. The last line of Table I generalizes the proposal. Namely, one can choose n ($\{n|n \in N^*\}$) in order to refine a volume in 4^n volumes in the new pack. Hence, the pack will have $3 \cdot \sum_{i=0}^{n-1} 2^i$ new transition nodes, where the sum until $n-1$ is because the levels are nested. Moreover, the sum of 2^i is because each edge is bisected and one transition node links two volumes. In addition, the sum is multiplied by three due to the number of edges of (the original volume: level 0) a triangle. There will be

$$\begin{cases} 3 \cdot 2^{n-1} \text{ transition nodes in level } +n, \\ 3 \cdot 2^{n-2} \text{ transition nodes in level } +(n\text{-}1), \\ \quad \ldots \\ 3 \cdot 2^0 \text{ transition nodes in level } +1. \end{cases}$$

Figure 4.6 shows an example with *n=2*. Hence, there are sixteen (4^2) volumes in the new pack, sixteen volume nodes and nine ($3 \cdot (1+2)$) transition nodes in the new sub-graph, being three ($3 \cdot 2^0$) transition nodes of level *n+1* and six ($3 \cdot 2^1$) transition nodes of level *n+2*.

Observing the new pack in Fig. 4.6, one can notice that columns (and lines) and lines have odd number of volumes. Clearly, the sum of the number of volumes of the columns (or the lines) results in the number of volumes of the pack. Consequently, one can write $[\forall n \in N|n \; even \wedge 2^n = \sum_{i=1}^{2^{\frac{n}{2}}}(2i-1)]$ since

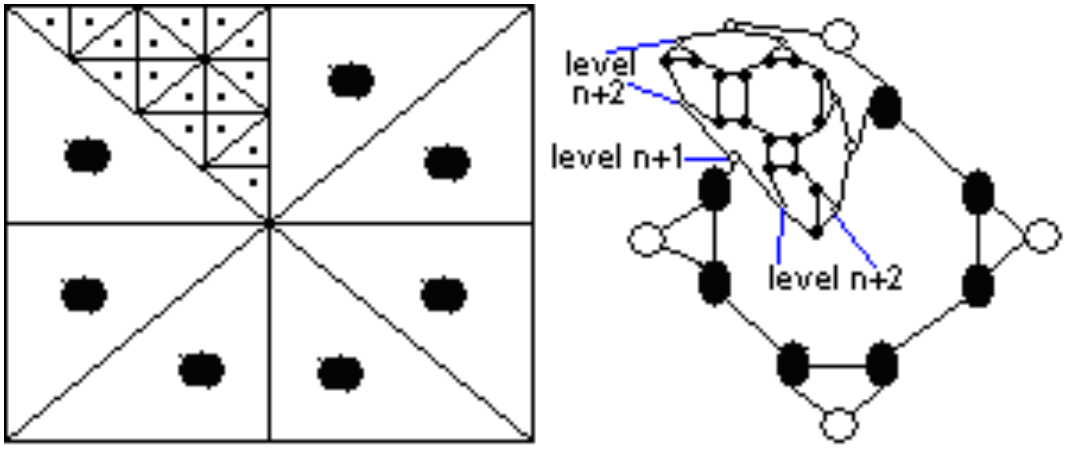

Figure 4.6: Volume refined directly in sixteen new volumes, and its graph representation, respectively.

$$\begin{cases} 2^0 = 1 \\ 2^2 = 1+3 \\ 2^4 = 1+3+5+7 \\ 2^6 = 1+3+5+7+9+11+13+15 \\ 2^8 = 1+3+5+7+9+11+13+15+\ldots+31 \\ \ldots \\ 2^n = \sum_{i=1}^{2^{\frac{n}{2}}}(2i-1). \end{cases}$$

This theorem is proved in Appendix C.

4.5 Unrefinement process

Figure 4.7 highlights the pack selected to be unrefined. All nodes in a common pack have an attribute which indicates that they belong to a common pack.

Figure 4.8 shows the pack marked in Fig. 4.7 replaced by a single volume node. The transition nodes may be regenerated if necessary. The required steps for unrefining a pack are: (a) filling the merged volume node with the new data, for instance, its spatial coordinates, level, and physical constants related to the problem, decreasing the refinement level etc.; (b) connection among the resulting volume node with its neighbors; (c) connection between each neighboring node with the resulting volume node; (d) release the memory space of the eliminated nodes.

Some aspects must be considered. Provided that the neighbor nodes are of the same

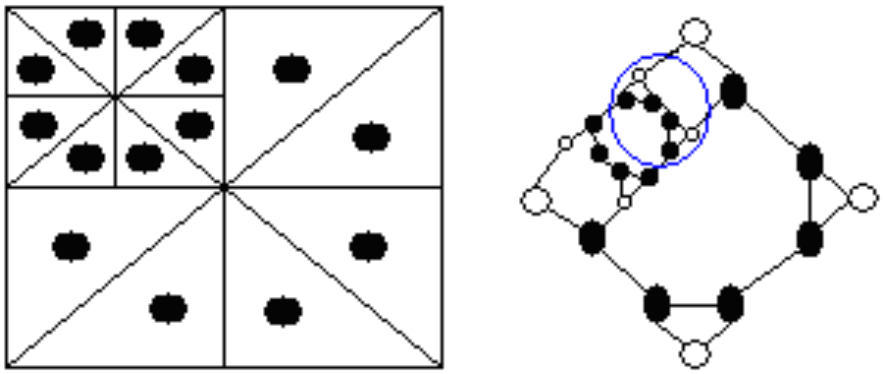

Figure 4.7: A discretized domain and its graph representation showing a pack selected to be unrefined.

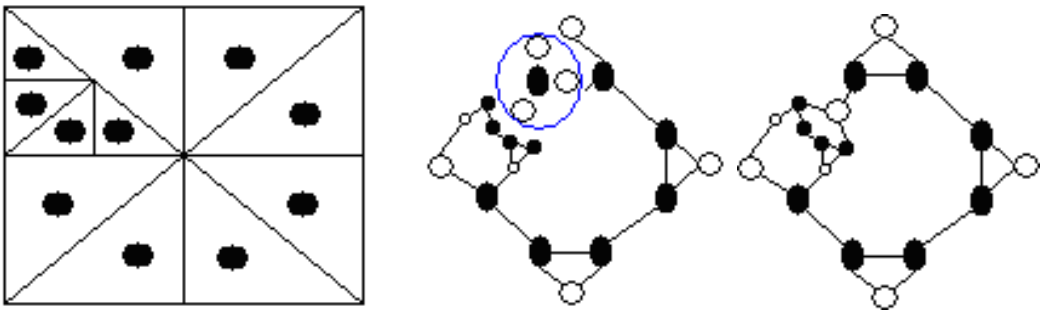

Figure 4.8: Final configuration of the discretization, pack collapse, and final configuration of the graph, respectively.

level of the resulting volume, pointers of the resulting volume node and its neighbors directly point to each other. Otherwise, transition nodes are created in order to link neighbors with different levels of refinement. There will be transition nodes from the level difference between the resulting volume node and its neighbor nodes. The most left-hand side of Fig. 4.8 presents the final configuration of the discretization followed by the intermediate graph representation and, finally, the most right-hand side graph of Fig. 4.8 is the final graph configuration of this example. Note that generating transition nodes between the resulting volume node and its neighbor nodes that have the same refinement level (refinement level 0 in this example) is not demanded. On the other hand, in this example, creating one transition node in order to link the pack with refinement level 1 was required.

Although this work does not implement the tetrahedral scheme, it is introduced in Appendix D, including a 3D Sierpiński-like Curve in order to number the tetrahedra. Finally, a general algorithm in order to solve a PDE is shown in Algorithm 4.1 (see Chapter 6 for details of the information system modeling).

Algorithm 4.1: Solving a PDE.

---

begin

boolean: continueRefining ← true;

initialize grid;

initialize ϕ with initial conditions;

FOR $t \leftarrow$ initial step time UNTIL $t < t_{final}$ STEP Δt

    WHILE (continueRefining) { refinement iterations }

        { initializes entries of the grid discretization matrix }

        { determines values of all entries in the matrix and the independent vector b }

        { each line of the coefficient matrix is a linked list Element of the volume node }

        { b_i is an attribute of the volume node }

        set discretization matrix (t);

        Conjugate Gradient Method;

        release memories of the Element linked lists;

        continueRefining ← RefineGrid ();

    end-WHILE;

    unrefinement;

end-FOR;

draw Sierpinski-like Curve;

end.

Chapter 5

Mesh total ordering

Mesh volumes should have some ordering so that each volume is related to each line of the resulting linear system in an implicit formulation. Thus, creating a direct strip is appropriate. [Velho, Figueiredo and Gomes, 1999] described a dual graph for triangular mesh in order to build triangle strips by heuristics. Those authors considered several issues related to paths on the dual graph of general triangular meshes.

A triangular sequence designates a sequential triangulation, as well as its generalizations. A triangular sequence defines a total-order relation on the set of triangles of a mesh. Dividing a triangular mesh in a collection of sub-triangulations so that each new pack is a triangular sequence is always possible [Velho, Figueiredo and Gomes, 1999].

[Velho, Figueiredo and Gomes, 1999] also explained that a generalized sequential triangulation, or Hamiltonian triangulation, is a triangulation in which there is an ordering $T_1, \ldots, T_N$ of all its triangles so that two consecutive triangles T_j and T_{j+1} share an edge. A path in a graph is said to be Hamiltonian if it visits all nodes in the graph exactly once. They also claimed that each triangle in a Hamiltonian triangulation has an entry and an exit edges with respect to the Hamiltonian ordering. The knowledge of these edges completely characterizes the sequence. Geometrically, a generalized triangular sequence can be represented by drawing an oriented path on the mesh domain that visits each triangle crossing its entry and exit edges, called a sequential path (see Fig. 4.1d). Likewise, the scheme proposed here always generates a generalized sequential triangulation.

Finite volumes of the mesh must be numbered so that the corresponding linear system has each row matching a specific finite volume. Thus, similarly to
[Velho, Figueiredo and Gomes, 1999], this present work builds the triangle strip while creating a triangulation. This work proposes a dynamic Sierpiński-like space-filling Curve for total ordering of a graph-based triangular mesh.

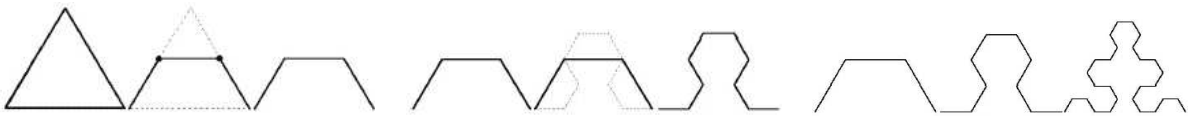

Figure 5.1: Generator process of the Sierpiński Curve through equilateral triangles divided into 4, 10 and 22 triangles, respectively.

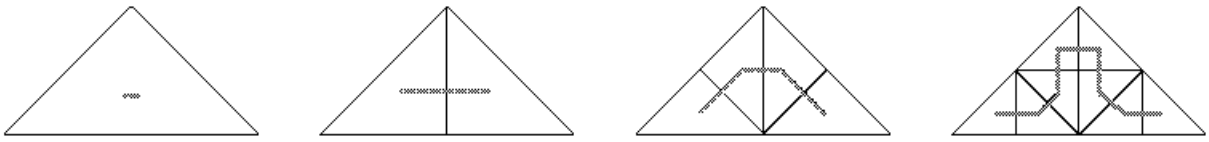

Figure 5.2: Sierpiński Curve generation.

5.1 Sierpiński Curve

A curve (with endpoints) is a continuous function whose domain is the unit interval $[0, 1]^2$. Moreover, the limit of the sequence given by curves of order $1, 2, \ldots$, is the curve that passes through each point in the unit square $[0; 1]^2$, or in a closed continuous surface. Furthermore, space-filling curves are curves whose ranges contain the entire 2D unit square (or the 3D unit cube).

The Sierpiński Curve is a space-filling curve proposed by [Sierpiński, 1912] and complemented in [Sierpiński, 1916]. Moreover, Sierpiński Curves are a recursively defined sequence of continuous closed plane fractal curves, which in the limit $n \to \infty$ completely fill the unit square: thus, their limit curve is an example of a space-filling curve. Sierpiński Curves in uniform successive refinement in equilateral triangles are depicted in Fig. 5.1.

Figure 5.2 depicts another example of Sierpiński Curve generation. As long as the domain shape is the left-hand side triangle of Fig. 5.2, and also the triangles are uniformly refined, the third level of refinement and ordering is such as the right-hand side curve of Fig. 5.2. Figure 5.3 depicts Sierpiński Curve ordering scalene triangles.

Consider the 2D Laplace Equation with equal prescribed boundary condition in all sides of the unit square. Let the Dirichlet Problem given by

$$\nabla^2 \phi = 0 \; in \; \Omega, \; \phi = f \; on \; \partial\Omega, \tag{5.1}$$

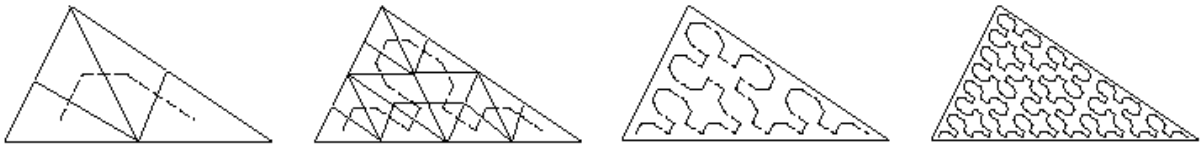

Figure 5.3: Sierpiński Curve ordering scalene triangles.

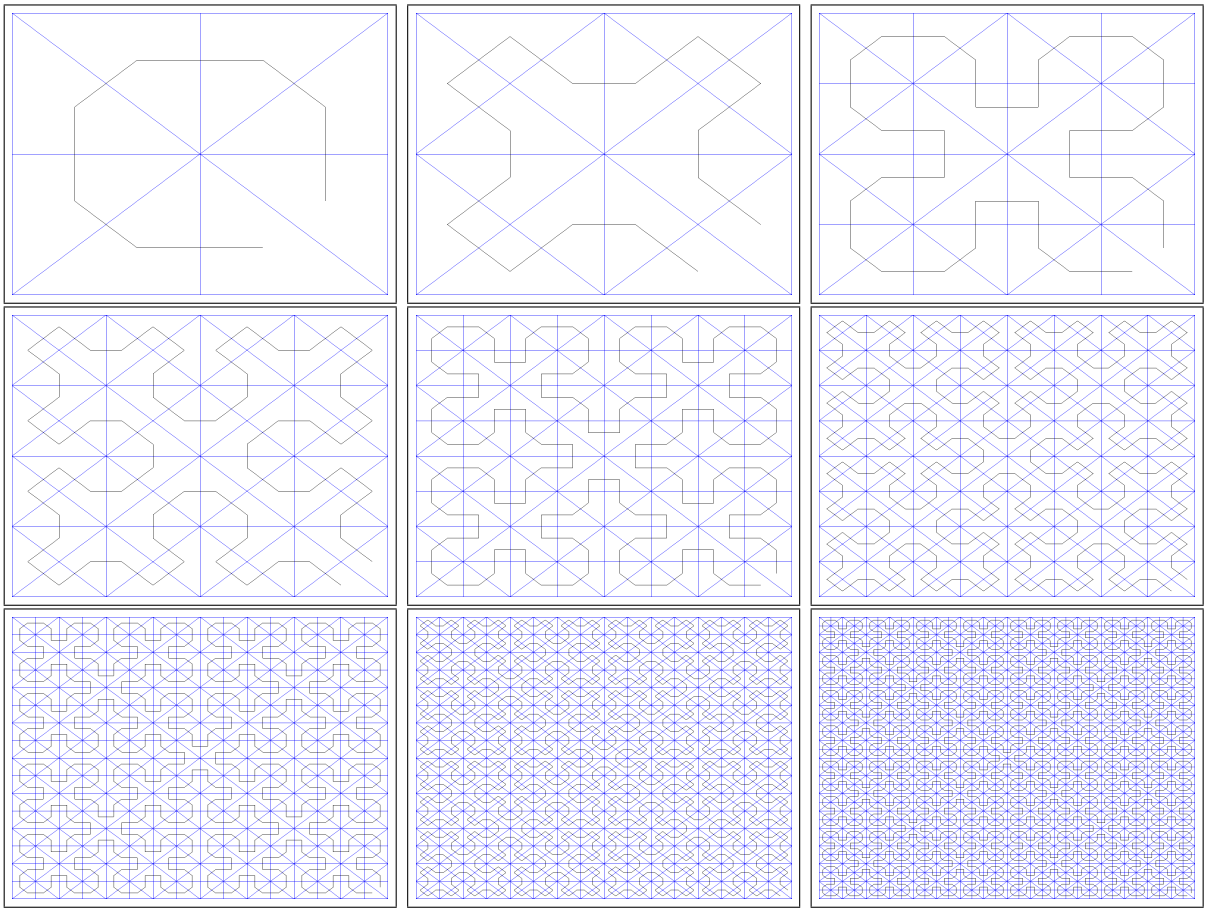

Figure 5.4: Sierpiński Curves in a unit square with 2^3, 2^3, ..., 2^{11} volumes.

where ϕ is the dependent variable of the EDP, Ω is a limited domain in $\mathbb{R}^2$, and f is a defined smooth function on boundary $\partial\Omega$.

In 1902, Jacques Salomon Hadamard (1865-1963) claimed that, for a mathematical problem correspond to reality, the following basic conditions should be satisfied: the solution must exist (existence); the solution must be determined by data of a unique form (unicity); and the solution must depend on data of continual form (stability). Problem 5.1 is well-posed in the sense of Hadamard according to [Zachmanoglou and Thoe, 1986].

Consider now the approximation of 5.1 in a unit square. Following the Finite Volume Method basic formulation for irregular meshes gives

$$\oint \nabla\phi \cdot \vec{n} d(\partial v_i) = 0, \tag{5.2}$$

where $\vec{n}$ is the normal outward vector of control volume v_i and ∂v_i represents the boundary of the control volume v_i. After numerical integration, the numerical solution of equation 5.2 is illustrated in Fig. 5.4 with constant boundary conditions. Moreover, through setting f as a constant function, Fig. 5.4 illustrates nine levels of the Sierpiński Curve.

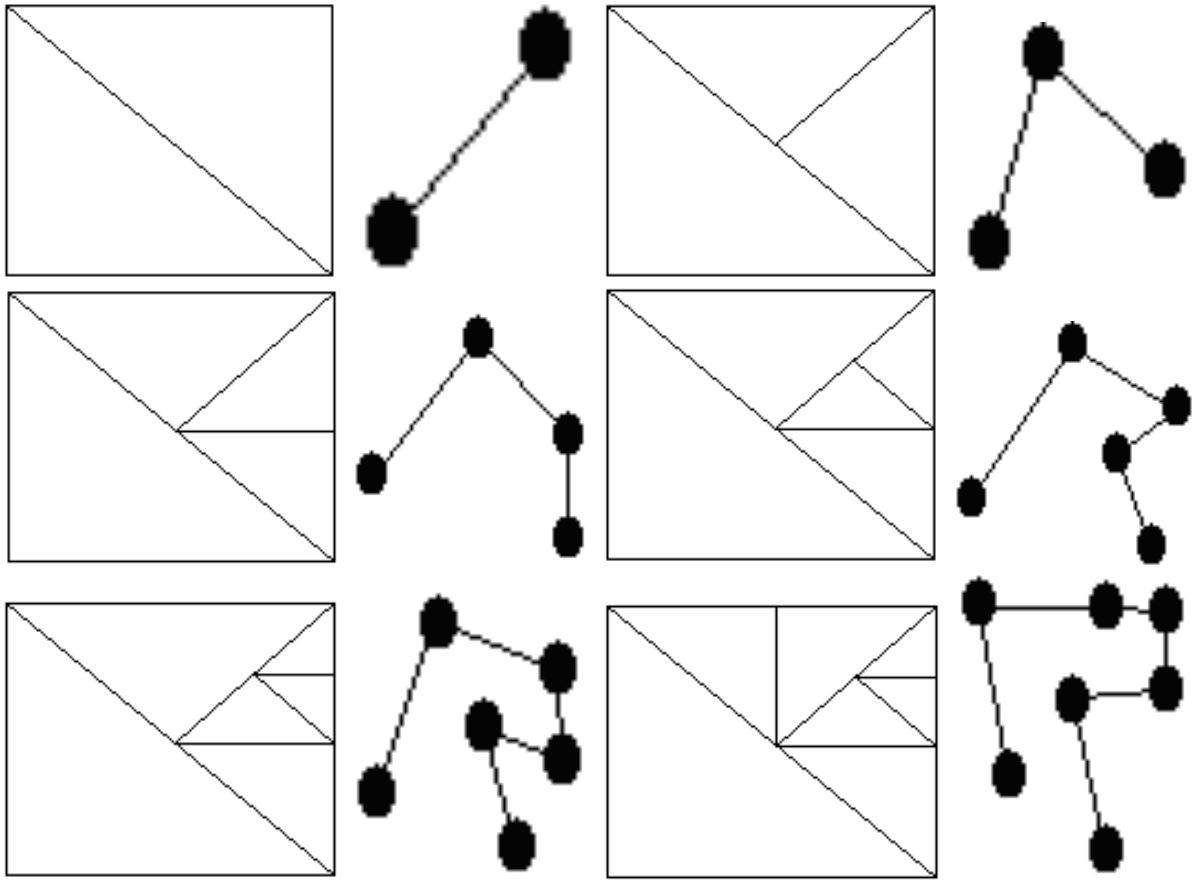

Figure 5.5: Successive adaptive refinements with volumes ordered by a Sierpiński-like Curve on the right-hand side of each discretization.

5.2 Sierpiński-like Curve for the total-order relation on the triangular volumes of the mesh

In [Gonzaga and Kischinhevsky, 2008], a non-regular Sierpiński Curve for total ordering of volumes of a discretized mesh was introduced. It is generated by a double-linked list. Exemplifying, Fig. 5.5 sketches successive adaptive refinements by volume bisection in a quadrangular domain with triangular volumes ordered by a Sierpiński-like Curve. This scheme allows straightforward update of the linked list for mesh node ordering. This means that, in the sense of data structures, this adaptive refinement scheme of triangular volumes enables straightforward update of the linked list in the modification of volume nodes in the adaptive refinement process during the evolution of the partial differential equation.

In summary, since triangular volumes are adaptively refined, this work uses a Sierpiński-like Curve. The modification is that volumes are adaptively refined by sucessive bisections.

Chapter 6

Experimental tests

Tests show numerical results in the Laplace (elliptic problem), Heat Conduction (parabolic problem), Wave (hyperbolic problem) Equations and flat-plate Boundary Layer Problem performed in an Intel® Celeron^{TM} 2.6GHz (approximated) processor with 256MB DDR SDRAM (plus 360MB virtual memory) and XGA display integrated with 32MB VRAM video memory.

In all tests accomplished, the linear systems presented sparse, symmetric and positive-definite coefficient matrices. It is very likely that this scheme always produces matrices with those properties.

Linear-system solvers based on the minimization of functionals were easily employed in this work. Specifically, the Conjugate Gradient Method [Hestenes and Stiefel, 1952] was employed. This method is one of the most proeminent iterative methods for solving sparse systems of linear equations. Notice that a pre-conditioner was not used.

Before showing the experimental tests, the modeling information is briefly presented as well as the refinement criteria adopted.

6.1 Information system modeling using UML

These object-oriented projects were coded in C++ language using OpenGL [OpenGL, 1993]. The main classes of the modeling are described in Fig. 6.1. The Grid object is composed by at least eight Volume objects. The Volume objects are strongly related to the Grid one. Namely, if the Grid object is deleted, so are all the Volume instances. Afterwards, a Volume instance has a constant number of Side objects. In the codes of this work, each Volume instance has three Side objects. Moreover, although the Side was implemented as a class, the objects are parts of the Volume object. Namely, the Side objects are allocated when the Volume objects are allocated. However, the code is prepared to be implemented

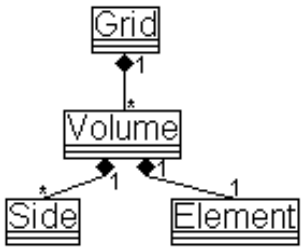

Figure 6.1: Simplified diagram class of this work.

with general polygonal volumes. Furthermore, if one codes volumes with n edges, the volume will have n Side objects.

A Volume class is also composed by one Element class. Moreover, an Element class is strongly related to the volume it belongs to: if the Volume object is deleted, so is the associated Element object.

The Grid class is the main one of the implementation. In this class, the data structures, the mesh node numbering, the adaptive mesh refinement (AMR) algorithm and the Conjugate Gradient Method calling are programmed. Next, the Volume class is responsible for implementing the volume nodes. It stores the connectivity information of the mesh nodes, and in its objects the Sierpinski-like Curve is implemented, among other functions. Subsequently, a Side object stores geometrical information about the edge (that it represents) of the volume which it belongs to. Finally, in the Element class, a simple linked list is implemented. In this linked list, the non-null coefficients of a line (related to the specific volume) of the resulting linear system are stored.

6.2 Refinement criteria

The refinement criteria is based on the gradient of the edge between two neighbor volumes. Namely, provided that the gradient of the flux in an edge is greater than a refinement quota established by the user, both volumes are refined. The choice of the user is based on the requirements of a larger or smaller refinement in certain regions of the domain where there is larger or smaller variation in the solution behavior. Since a large variation in certain regions may occur in the analytical solution, a large difference of fluxes among neighboring volumes does not necessarily mean that there is a large variation in the difference between the approximation and the analytical solution. Therefore, those criteria chosen by the user overestimate the number of refinements needed. Similarly to [Burgarelli, Kischinhevsky and Biezuner, 2006], an error estimator, tailored to specific applications, can be used in order to improve the performance of the algorithm.

Since this present approach tailors the work proposed in [Schneider and Maliska, 2002],

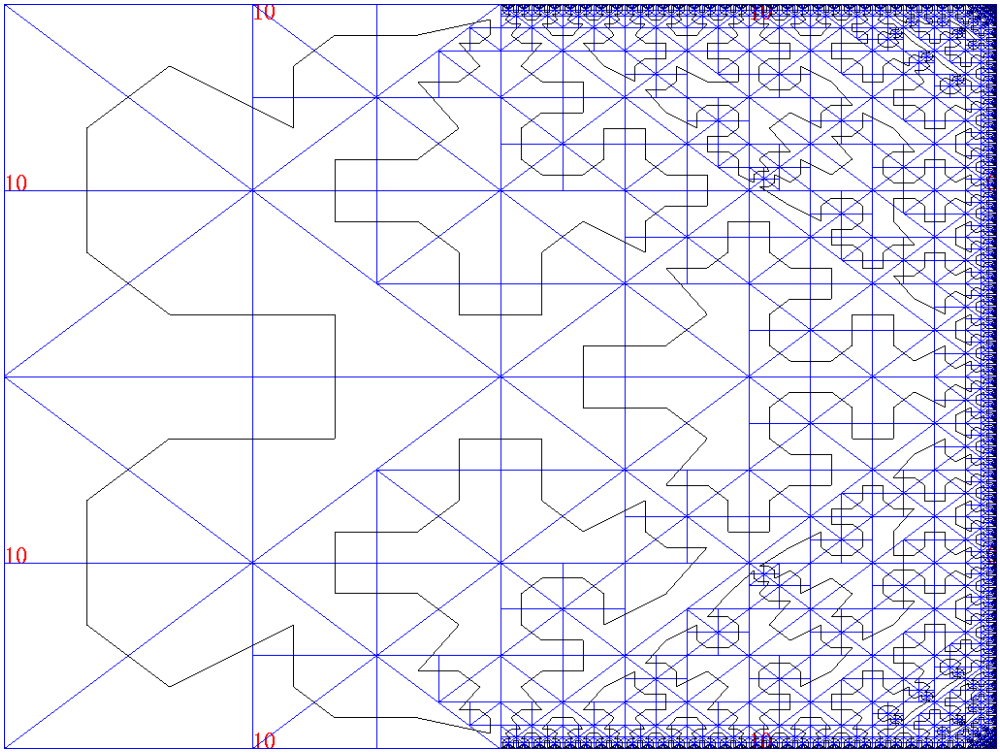

Figure 6.2: Sierpinski-like Curve of the AMR scheme applied to the Laplace Equation.

this choice could present a less homogeneous distribution of the edge sizes and the sizes of the proper control volumes. For this reason, seeking a smooth (or graded) transition among neighbor volumes in the experimental tests, a limit of the difference of refinement levels between neighbor volumes was implemented. That was performed in order to avoid numerical oscillations, though the scheme and the graph data structure allow any difference in the level of refinements between neighbor volumes. In other words, the areas of neighboring triangles do not differ dramatically.

6.3 Laplace Equation (elliptic problem)

[Gonzaga et al., 2008] approximated the problem given in 5.1, where on top, bottom, and left sides of the unit square present prescribed boundary condition with a unique constant f. On the other hand, right side of the unit square presents a different value for f. Moreover, Fig. 6.2 illustrates the Sierpinski-like Curve of this AMR scheme. Furthermore, Fig. 6.2 shows the final mesh configuration after eight time steps using 1.5 in the refinement criteria. The mesh is comprised of 4421 volumes, whose Conjugate Gradient Method converges after 729 iterations.

6.4 Heat Conduction Equation (parabolic problem)

Consider the 2D Heat Conduction Equation with continuous initial and boundary data

$$\begin{aligned} \phi_t &= \nabla^2\phi \\ \phi(u,0) &= f(u), u \in \Omega, f(u) \equiv 0, \\ \phi(u,t) &= g(u), u(u) \in \Omega, t \geq 0, \end{aligned} \tag{6.1}$$

where the bounded $\Omega \subset \mathbb{R}^2$, thus, $u = (x, y)$, and f is a smooth function limited in Ω. Problem 6.1 is well-defined in the sense of Hadamard [Zachmanoglou and Thoe, 1986]. Therefore, consider the approximation of 6.1 in a unit square. Consider now the Finite Volume Method (FVM) basic implicit formulation for irregular meshes

$$M_P^{k+1}\phi_P^{k+1} - \Delta t \oint_{\partial\Omega} \nabla\phi^{k+1} \cdot \vec{n} d(\partial\Omega) = M_P^k \phi_P^k, \tag{6.2}$$

where M represents the area of $\triangle P$ and $\Delta t = 0.1$.

Boundary conditions on top, bottom and left sides of the unit square have a unique prescribed boundary value f and right side has a different value. Figure 6.3 illustrates the final mesh configuration after ten time steps ($\Delta t = 0.1$) using 1.0 in the refinement criteria and seven maximum refinement levels for each volume. The mesh is comprised of 5285 volumes, whose Conjugate Gradient Method converges in 578 iterations.

Figure 6.4 illustrates a test with boundary condition $x - y$ and the refinement criteria is 0.05. Moreover, Fig. 6.4 shows the final discretizations after ten time steps ($\Delta t = 0.1$) with six maximum levels of refinement for each volume making up 6110 volumes whose Conjugate Gradient Method converges in 10 rounds comprising from 520 to 181 iterations.

Figure 6.5 illustrates a test with boundary condition $x^2 - y^2$, 0.07 as the refinement criteria and ten time steps where $\Delta t = 0.1$. Moreover, Fig. 6.5 shows the final discretizations with seven maximum levels of refinement for each volume making up 4472 volumes whose Conjugate Gradient Method converges in 10 rounds comprising from 640 to 278 iterations.

6.4.1 Performance of the refinement process

[Burgarelli, Kischinhevsky and Biezuner, 2006] showed those boundary conditions of the approximation to the Heat Conduction Equation in order to demonstrate the performance

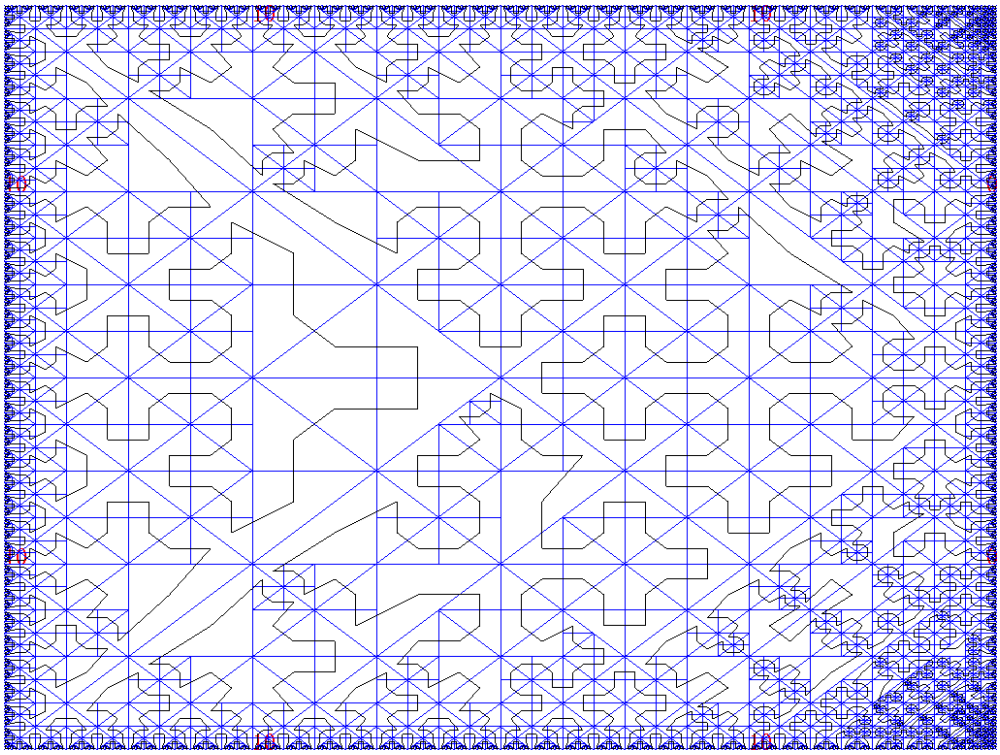

Figure 6.3: Final mesh configuration of an AMR approximation to the Heat Conduction Equation.

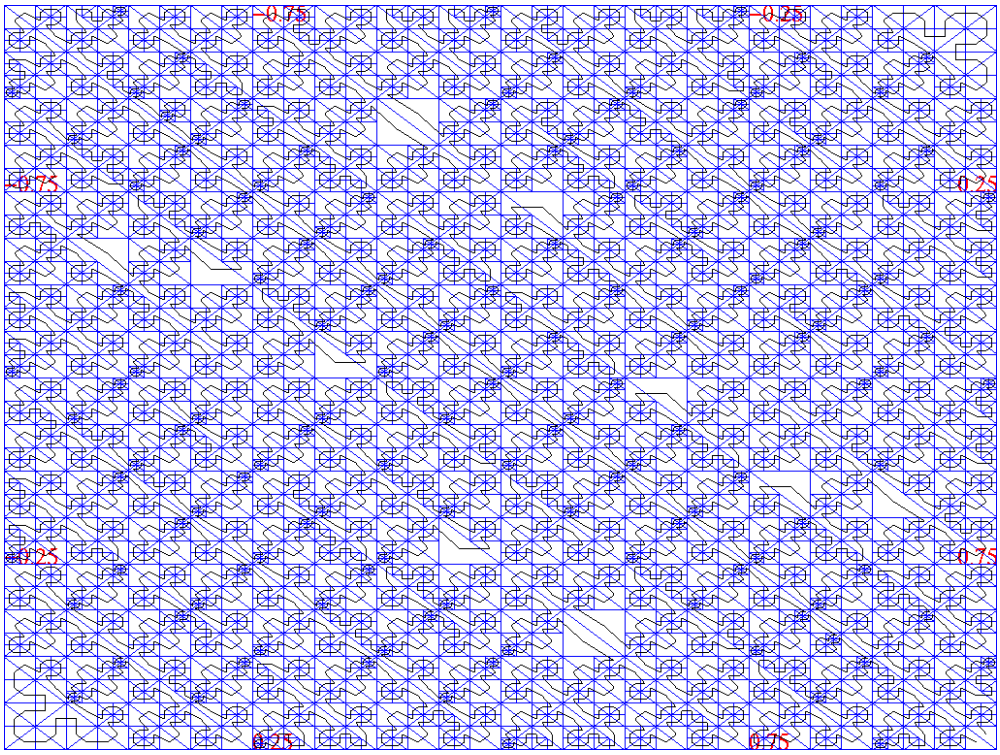

Figure 6.4: Final time step of an AMR approximation to the Heat Conduction Equation with boundary condition: $x - y$.

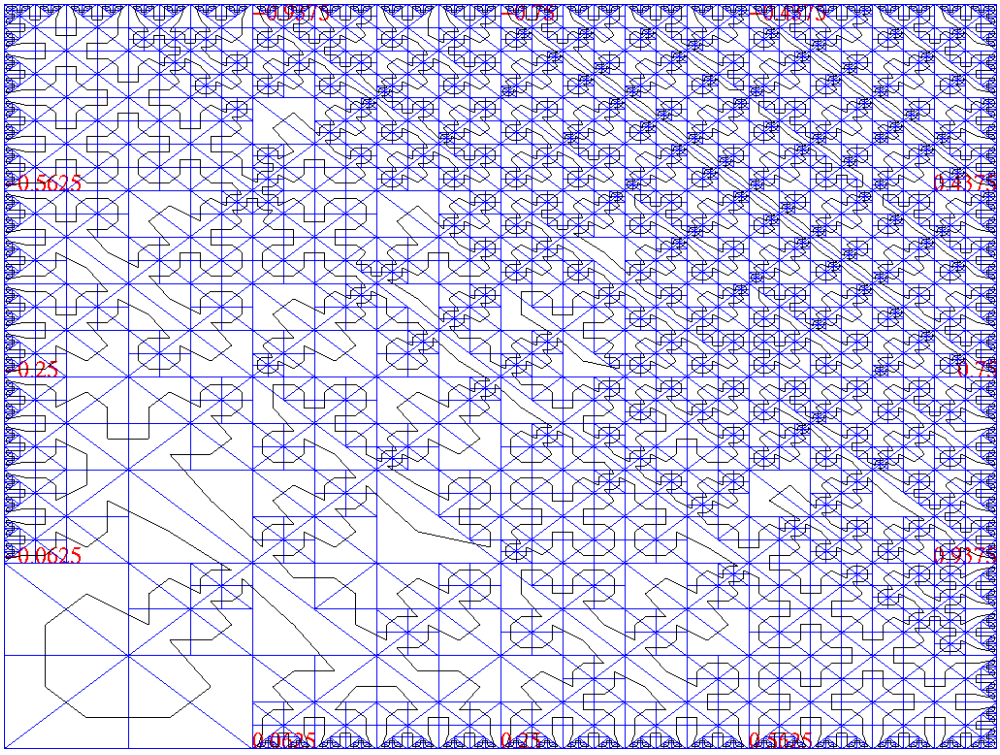

Figure 6.5: AMR approximation of the Heat Conduction Equation with boundary condition: $x^2 - y^2$.

of their scheme. Thus, the same boundary conditions were applied here in order to compare the results.

Appropriate refinement criteria was chosen so that the number N of volumes is similar to the ones presented in [Burgarelli, Kischinhevsky and Biezuner, 2006]. In Table II, *Condition* column means the boundary condition of the test; L is the maximum refinement level reached; N is the number of volumes attained in the grid; *RefTime* is the total time (normalized) spent in the refinement process; *Time* column is the total time (rounded in the first digit) spent in numerical computations; and $\% = 100\frac{\text{RefTime}}{\text{Time}}$ (rounded in the second digit). Times are given in seconds. The first N, *RefTime*, *Time* and % are related to the results showed in [Burgarelli, Kischinhevsky and Biezuner, 2006], where Autonomous Leaves Graph (ALG) was presented; whereas the second group is related to the work presented here.

In this group of tests, criteria used for refinement are based on the flux across the interface of neighboring volumes. More precisely, in order to produce similar conditions among tests, if the difference of the absolute value of the flux in an edge is larger than a refinement quota chosen by the user, the implementation refines both volumes. The choice of the user is based on his requirements of a greater or lesser refinement in certain regions of the domain where there is greater or lesser variation in the behavior of the solution, respectively. These criteria overestimate the number of refinements needed, since a large

difference of fluxes does not necessarily mean that there is a large variation in the difference between the approximate and exact solutions, for this variation can already occur in the exact solution. The tests were performed choosing $\Delta t = 0.1$ with the program running until $t = 10\Delta t$. The initial conditions were set as $f(x, 0) = 0$. In the level L, the minimum volume size attained is 2^{-L}.

Results are summarized in Table II where comparisons between the results from [Burgarelli, Kischinhevsky and Biezuner, 2006] and this present work can be carried out since the number of volumes is similar in the simulations. Moreover, Table II shows that most of the computing time is spent in the core of the computation, leaving very low processing time to refinement stages. Furthermore, the higher resolution of results is required, the more the computational cost in refinement stages decreases. Notice that since the hardware that [Burgarelli, Kischinhevsky and Biezuner, 2006] obtained their results is not known, the comparison is performed in percentual as described.

Table II: Numerical results for the Heat Conduction Equation.

Tests		Quadrangular ALG				Triangular AMR scheme			
Condition	L	N	*RefTime*	*Time*	%	N	*RefTime*	*Time*	%
$g(x, y) = 10$	5	856	$6.8 \cdot 10^{-3}$	$5.6 \cdot 10^{-2}$	12.1	845	$1.3 \cdot 10^{-2}$	1.5	0.89
	6	3016	$1.9 \cdot 10^{-2}$	0.8	2.4	2906	$4.1 \cdot 10^{-2}$	13.1	0.32
	7	3580	$3.5 \cdot 10^{-2}$	2.5	1.4	3755	$8.6 \cdot 10^{-2}$	29.6	0.29
$g(x, y) = x - y$	5	934	$7.5 \cdot 10^{-3}$	0.2	3.8	956	$8.1 \cdot 10^{-3}$	1.5	0.53
	6	4090	$5.0 \cdot 10^{-2}$	3.2	1.6	4028	$4.5 \cdot 10^{-2}$	21.8	0.23
	7	15682	$1.2 \cdot 10^{-1}$	21.6	0.6	15734	$2.1 \cdot 10^{-1}$	285.5	0.07
$g(x, y) = x^2 - y^2$	6	3529	$3.4 \cdot 10^{-2}$	4.7	0.7	3431	$7.7 \cdot 10^{-3}$	37.5	0.02
	7	12442	$1.2 \cdot 10^{-1}$	28.6	0.4	12479	$2.9 \cdot 10^{-2}$	282.9	0.01

According to Table II, the computing time spent in the refinement stage of this present work is smaller than ALG. This is mainly because these points:

i) Difference in the complexity of the algorithms that implement the space-filling curves in both approaches. In the original quadrangular ALG,
[Burgarelli, Kischinhevsky and Biezuner, 2006] tailored the Hilbert Curve for this issue. Moreover, the authors proposed a Modified Hilbert Curve (MHC) in order to number the volumes. In this present work, a Sierpiński-like Curve is employed;

ii) Searching connectivity information in order to adjust the pointers of the graph nodes. The original quadrangular ALG scans each graph node from (a set of) four pointers whereas this present approach scans each graph node from (a set of) three pointers.

6.4.1.1 Complexity analysis of the space-filling curves: MHC and Sierpiński-like Curve

i) Complexity analysis of the MHC

The MHC algorithm requires to find the local shape of the Hilbert Curve for each local refinement. Namely, the local curve can have four shapes: $\sqsubset$, $\sqsupset$, $\sqcap$ and $\sqcup$. Algorithm 6.1 evaluates which local shape the new pack is.

Algorithm 6.1: Evaluate the Hilbert Shape.

```
HilbertShape (integer: volHilbertCoordinate, integer: volRefinementLevel): integer
begin
integer: i ← 0, j, k;
matrix hilbertTable[4:4]: integer;
hilbertTable[0][0] ← 1; hilbertTable[0][1] ← 0;
hilbertTable[0][2] ← 0; hilbertTable[0][3] ← 3;
hilbertTable[1][0] ← 0; hilbertTable[1][1] ← 1;
hilbertTable[1][2] ← 1; hilbertTable[1][3] ← 2;
hilbertTable[2][0] ← 3; hilbertTable[2][1] ← 2;
hilbertTable[2][2] ← 2; hilbertTable[2][3] ← 1;
hilbertTable[3][0] ← 2; hilbertTable[3][1] ← 3;
hilbertTable[3][2] ← 3; hilbertTable[3][3] ← 0;
FOR k ← 1 UNTIL volRefinementLevel STEP 1 DO
    {In computing, the modulo operation finds the
     remainder of division of one number by another.}
    j ← volHilbertCoordinate modulo 4;
    i ← hilbertTable[i][j];
    volHilbertCoordinate ← vollHilbertCoordinate / 4;
end-FOR;
RETURN i;
end-HilbertShapeNumber.

...
integer: numberOfHilbertShape;
```

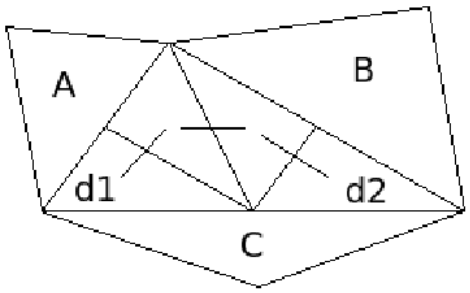

Figure 6.6: Volume ordering in a local triangular refinement.

numberOfHilbertShape ← HilbertShape (volHilbertCoordinate, volRefinementLevel+1);
IF (numberOfHilbertShape = 0) { Shape: ⊏ }
{ 11 changes + 1 condition }
ELSE IF (numberOfHilbertShape = 1) { Shape: ⊔ }
{ 11 changes + 1 condition }
ELSE IF (numberOfHilbertShape = 2) { Shape: ⊐ }
{ 11 changes + 1 condition }
ELSE IF (numberOfHilbertShape = 3) { Shape: ⊓ }
{ 11 changes + 1 condition }
...

Analysing Algorithm 6.1, the core of the computation is the instruction FOR. The loop in the instruction FOR performs *volRefinementLevel* times and it defines the order of growth of the running time of the algorithm. Therefore, MHC $\in$ *O(L)*, where L is the level of refinements of the refined pack. Moreover, since the instruction FOR runs *volRefinementLevel* times, it defines the order of growth of the running time of the algorithm. CHM could store in each node the *volRefinementLevel* already evaluated in order to improve its efficiency.

ii) Complexity analysis of the Sierpiński-like_Curve

Figure 6.6 shows a local refinement of a triangle. Since the local Sierpiński-like Curve orders the new pack starting (or finishing in) from volume d1 and finishing (or starting from) in volume d2, the curve can link: A $\rightarrow$ d1 $\rightarrow \cdots$ d2 $\rightarrow$ B; A $\rightarrow$ d1 $\rightarrow \cdots$ d2 $\rightarrow$ C; or B $\rightarrow$ d2 $\rightarrow \cdots$ d1 $\rightarrow$ C; and the path from B/C to A/B, respectively, since it is implemented by a double-linked list.

Algorithm 6.2 presents a solution of the local Sierpiński-like Curve.

Algorithm 6.2: Evaluate the local Sierpiński-like_Curve.

```
...
FaceNumber ← 3; { constant value }
boolean: first ← true; { previous → cell2 → cell4 → cell3 → cell1 → next }
integer: i, j;
{ Cell2 contains the conectivity information of the local double-linked list:
   each node contains pointers to the previous and the next nodes }
IF (Cell2->previous = NULL) { isn't it the first volume? }
    FOR i ← 1 UNTIL FaceNumber STEP 1 DO { for each edge of Cell1 }
        { for each edge of the previous node of the linked list }
        FOR j ← 1 UNTIL FaceNumber STEP 1 DO
            { compares each pair of coordinates of Cell1 with
              each pair of coordinates of the previous node of the list }
            IF Cell1->edge[i](x,y) = Cell2->previous->edge[j](x,y)
                first ← false; { previous → cell1 → cell4 → cell3 → cell2 → next }
                BREAK; { jump to the next instruction after loops }
            end-IF;
        end-FOR;
    end-FOR;
end-IF;
...
```

Analysing Algorithm 6.2, the core of the computation are two nested instructions FOR and they define the order of growth of the running time of the algorithm. Since *FaceNumber* is a constant value, the limits in the instructions FORs are constant.

Now, consider another solution in Algorithm 6.3, which verifies the consistency of the Sierpiński-like Curve in each local refinement.

Algorithm 6.3: Evaluate the local Sierpiński-like_Curve verifying its consistency.

```
...
FaceNumber ← 3; { constant value }
boolean: first ← true; { previous → cell2 → cell4 → cell3 → cell1 → next }
integer: i, j, m, n;
{ Cell2 contains the conectivity information of the local double-linked list:
   each node contains pointers to the previous and the next nodes }
IF (Cell2->previous = NULL) { isn't it the first volume? }
    FOR i ← 1 UNTIL FaceNumber STEP 1 DO { for each edge of Cell1 }
        { for each edge of the previous node of the linked list }
        FOR j ← 1 UNTIL FaceNumber STEP 1 DO
            { compares each pair of coordinates of Cell1 with
              each pair of coordinates of the previous node of the list }
            IF Cell1->edge[i](x,y) = Cell2->previous->edge[j](x,y)
                IF (SubCell2->next = NULL) { is it the last volume? }
                    first ← false; { previous → cell1 → cell4 → cell3 → cell2 → next }
                    BREAK; { jump to the next instruction after loops }
                end-IF;
                FOR m ← 1 UNTIL FaceNumber STEP 1 DO
                    FOR n ← 1 UNTIL FaceNumber STEP 1 DO
                        { compares coordinates from the 2nd cell to the next one }
                        IF Cell2->edge[m](x,y) = Cell2->next->edge[n](x,y)
                            first ← false; { previous→cell1→cell4→cell3→cell2→next}
                            BREAK; { jump to the next instruction after loops }
                        end-IF;
                    end-FOR;
                end-FOR;
            end-IF;
        end-FOR;
    end-FOR;
end-IF;
```

...

Algorithm 6.2 or 6.3 is followed by Algorithm 6.4.

Algorithm 6.4: Sierpiński-like_Curve links the nodes of the new pack.

...

IF (first = true) { update the double-linked list }

IF (Cell2->next = true) Cell2->next->previous ← Cell1;

Cell1->next ← Cell2->next;

Cell3->next ← Cell1;

Cell4->next ← Cell3;

Cell2->next ← Cell4;

Cell1->previous ← Cell3;

Cell3->previous ← Cell4;

Cell4->previous ← Cell2;

ELSE

Cell2->previous->next ← Cell1;

Cell1->next ← Cell3;

Cell3->next ← Cell4;

Cell4->next ← Cell2;

Cell1->previous ← Cell2->previous;

Cell3->previous ← Cell1;

Cell4->previous ← Cell3;

Cell2->previous ← Cell4;

end-IF.

...

Since the implementation was coded in order to use volumes with n edges, *FaceNumber* denotes the number of edges of the volume. Even employing the Algorithm 6.3 in which there are four nested intructions FOR, in a superior quota, there are 3^4 conditions, i.e. the limits of the loops are constant values that do not depend on any input of the algorithm (or the level of the refinement). Therefore, Sierpiński-like_Curve $\in$ *O(1)*.

Emphasizing, Ω defines an inferior quota for the growth of the procedure. Necessarily, CHM performs the main loop *volRefinementLevel* times. Therefore, CHM $\in \Omega(L)$. On the other hand, Sierpiński-like_Curve may find the node in one round even in the Algorithm 6.3. Thus, Algorithm 6.2 or 6.3 may *break* a loop if it finds the node being searched. Therefore, Sierpiński-like_Curve $\in$ *$\Omega(1)$*.

$$\textit{Corollary.}\ \text{CHM} \in \Theta(L) \wedge \text{Sierpiński-like\_Curve} \in \Theta(1). \tag{6.3}$$

Proof. Straightforwardly, using [Cormen et al., 2001]

$$\textit{Theorem.}\ ((\forall f(n), g(n))(f(n) \in \Theta(g(n)) \Leftrightarrow f(n) \in O(g(n)) \wedge f(n) \in \Omega(g(n)))).\square \tag{6.4}$$

6.4.1.2 Quadrangular volumes × Triangular volumes

In the local refinement, each volume node is scanned in order to adjust the pointers of the new pack in relation to the original node. Each volume node is scanned from (a set of) four pointers in the quadrangular original ALG. On the other hand, each volume node is scanned from (a set of) three pointers in this present work. At a first sight, this could not contribute significantly; however, with a large number of volumes (nodes) refined after several time steps, this may contribute to the difference of performance presented in Table II.

In summary, there are two main differences: i) complexity of the space-filling curve algorithms; and ii) number of pointers when searching conectivity information for adjusting the pointers of the graph nodes. Those differences in the approaches may be relevant if applied to several of nodes after n time steps.

6.5 Wave Equation (hyperbolic problem)

The 2D first-order Wave Equation is

$$\frac{\partial \phi}{\partial t} + c_1 \frac{\partial \phi}{\partial x} + c_2 \frac{\partial \phi}{\partial y} = 0. \tag{6.5}$$

Consider two waves moving along opposite directions with the coordinated axes. Namely, one wave moves diagonally from (0,0) to (1,1) and the other moves from (1,1) to (0,0). In particular, let $c_1 = c_2 = \frac{\sqrt{2}}{2}$ and $\phi = \phi_1(x, y, t) + \phi_2(x, y, t)$, where ϕ_1 satisfies

$$\frac{\partial \phi_1}{\partial t} + c_1 \frac{\partial \phi_1}{\partial x} + c_2 \frac{\partial \phi_1}{\partial y} = 0, \tag{6.6}$$

with initial condition given by

$$\phi_1(x, y, 0) = \begin{cases} 1 - 16[(x - 0.25)^2 + (y - 0.25)^2] & \text{if } \phi_1(x, y, 0) \geq 0, \\ 0 & \text{if } \phi_1(x, y, 0) < 0 \end{cases} \tag{6.7}$$

and ϕ_2 satisfies

$$\frac{\partial \phi_2}{\partial t} + c_1 \frac{\partial \phi_2}{\partial x} + c_2 \frac{\partial \phi_2}{\partial y} = 0, \tag{6.8}$$

with initial condition given by

$$\phi_2(x, y, 0) = \begin{cases} 1 - 16[(x - 0.75)^2 + (y - 0.75)^2] & \text{if } \phi_2(x, y, 0) \geq 0, \\ 0 & \text{if } \phi_2(x, y, 0) < 0 \end{cases} . \tag{6.9}$$

In the previous tests of this chapter, the Eulerian approach was employed. Briefly, it means dots in the (inertial) space. In relation to numerical modeling, explicit methods are subjected to CFL stability condition. Moreover, although implicit approaches (such as used in this work) are unconditionally stable, depending on the flux velocity in a Δt too large and small Δx, the scheme may not be able to physically describe the reallity, and numerical errors may occur when a Eulerian approach is used. Therefore, the FVM for advection problems are subjected to CFL stability condition.

CFL stability condition imposes that the maximum stable time step for the entire mesh depends on the minimum control volume area. Avoiding CFL stability condition is especially important in an AMR scheme because there are few control volumes very small, which force a reduced time step for the whole mesh. The numerical stability requeriment for numerical methods using the Eulerian approach for solving hyperbolic partial differential equations (PDEs) is the CFL condition

$$\left| c \frac{\Delta t}{\Delta x} \right| \leq 1 \tag{6.10}$$

and $c\frac{\Delta t}{\Delta x}$ is called the *Courant number*. An approach subjected to CFL condition is conditionally stable.

Another approach in order to build the mesh is the Lagrangian approach. It is de-

fined as forming a mesh scheme that in each time step the moviment of the parcels of the (non-inertial) fluid is accompanied. Moreover, in such mesh scheme, a certain magnitude of flux is in function of time and the element of the fluid. Afterwards, the main idea of the Eulerian-Lagrangian approach is to reconcile the advantages of both approaches, i.e. the regular distribution of the particles in time step $t + \Delta t$ is maintained, computation is facilitated and it enables larger time steps than the other approaches since the Eulerian-Lagrangian approach is unconditionally stable because the domain of the dependency of the numerical solution always include the dependency of the exact solution [Fonseca, 2005].

Thus, a Eulerian-Lagrangian approach is employed in this test. The applied procedure to find the value in the volume that contains the coordinates of the foot of a characteristic line, i.e. the intersection point of a characteristic line with the computational domain, in the previous instant is the Modified Method of Characteristics [Douglas and Russel, 1982]. In the linear approach, if (x_P, y_P) denote the coordinates of the current volume, the coordinates (x_C, y_C) of the foot of the characteristic line that intersects this volume are obtained by $x_C = x_P - c_1\Delta t$ and $y_C = y_P - c_2\Delta t$.

After calculating the coordinates of the foot of a characteristic line, one needs to find the volume that corresponds to these coordinates. The starting point is the volume, which in the previous time step meets the characteristic line. The neighbors are scanned and the walking direction is chosen through comparing the horizontal and vertical distances between the barycenter of the current volume and the foot of the characteristic line. The direction of the largest distance is taken as the walking direction, moving thus to the contiguous volume, which then becomes the current volume for walking, and the process continues until the volume that contains the foot of the characteristic line is reached. Algorithm 6.5 presents the basic algorithm for solving this problem through dynamic refinements and unrefinements.

Algorithm 6.5: A Wave Equation solution.

---

begin

initialize ϕ with initial conditions;

initial refinement;

store ϕ_{old};

WHILE $(t < t_{final})$

Modified Method of Characteristics;

store u_{old};

refine/unrefine;

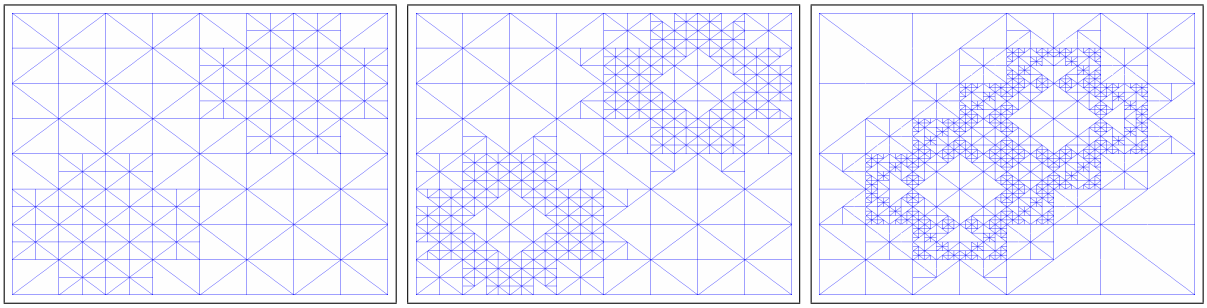

Figure 6.7: AMR approximation of the Wave Equation through the Modified Method of Characteristics.

$t = t + \Delta t$;

end-WHILE;

end.

The first discretized domain in Fig. 6.7 shows the mesh refined for this problem at the instant $t = 0$ with 272 volumes. The second discretized domain in Fig. 6.7 illustrates the same pattern with more refinements comprising of 800 volumes. Afterwards, the program scans the entire mesh trying detecting if there is some unrefinement that fulfills the required tolerance for the problem. Some regions are then unrefined, e.g. where the solution pack has zero value. The third discretized domain in Fig. 6.7 displays a posterior stage of this adaptive refinement and unrefinement for a certain instant of time when the two waves overlap.

6.6 Flat-plate Boundary Layer Problem

In Physics and Fluid Mechanics, the boundary layer is the fluid layer in the immediate vicinity of a bounding surface. It is important in many aerodynamic problems. This section presents a numerical simulation of the 2D laminar Boundary Layer Problem considering a steady incompressible flow with no-slip condition on the surface. The numerical solution for the flat-plate Boundary Layer Problem is compared to its analytical solution, namely Blasius Solution. Figure 6.8, tailored from [Anderson, Tannehill and Pletcher, 1984], sketches the flat-plate boundary-layer process.

Let $\varepsilon = \frac{\triangle}{R}$, where $\triangle$ is the velocity boundary-layer thickness and R is the reference length used in the Reynolds number [Anderson, Tannehill and Pletcher, 1984]. Using a magnitude analysis order, no term in the y-momentum equation is larger than ε in the estimated magnitude and the well-known governing Navier-Stokes Equations of viscous fluid flow can be greatly simplified within the boundary layer. Notably, the PDE characteristic

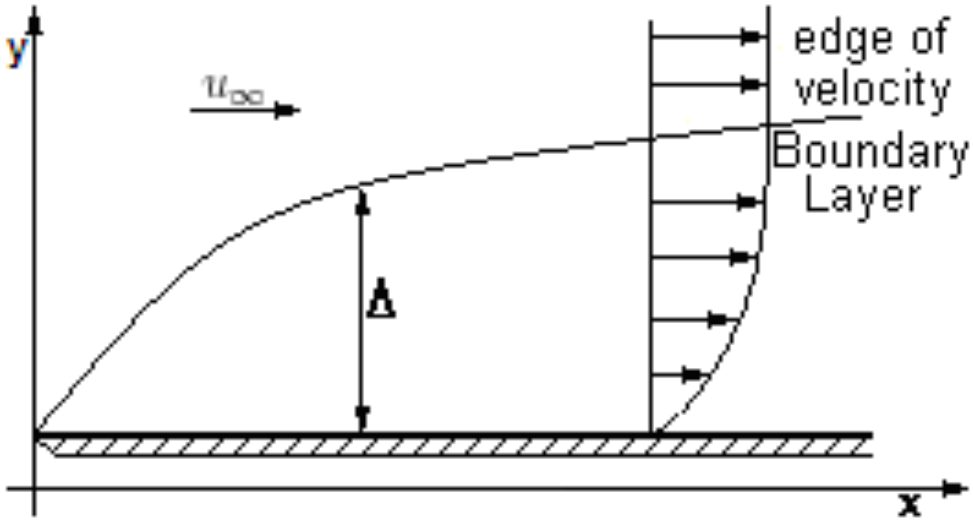

Figure 6.8: Flat-plate boundary-layer process.

becomes parabolic, rather than the elliptical form of the full Navier-Stokes Equations. This greatly simplifies the solution of the equations. Thus, the Navier-Stokes Equations for a 2D steady incompressible flow in Cartesian coordinates are given by the momentum and the continuity equations, i.e. the nonlinear governing PDEs in terms of dimensional variables are given by [Anderson, Tannehill and Pletcher, 1984]

$$Continuity : \nabla \cdot \phi = 0, \tag{6.11}$$

$$Momentum : \vec{u}\nabla \cdot \phi = -\frac{1}{\rho}\frac{dp}{dx} + \nu\frac{\partial^2 \phi}{\partial y^2}, \tag{6.12}$$

where p is pressure, $\vec{u}$ is the vector field, $\phi(u, v)$ is the PDE dependent variable, ρ is the fluid density and ν is the kinematic viscosity.

6.6.1 A finite volume discretization of the Boundary Layer Problem

[Gonzaga, Guedes and Kischinhevsky, 2007] presented details of a quadrangular finite volume discretization of the flat-plate Boundary Layer Problem. In addition,
[Gonzaga and Kischinhevsky, 2009] presented more details of the quadrangular finite volume discretization of the Boundary Layer Problem also simulating a NACA0012 airfoil in the domain. Tailoring their approach, the triangular finite volume discretization of Eq. (6.11) is

$$Continuity : u_P^{t+1} - u_{i_S}^{t+1}\frac{v_P^t}{u_P^t} = u_{i_W}^t - v_P^t, \tag{6.13}$$

where t represents the time step, P is the current volume, u_{i_S} and u_{i_W} represent volumes in south and west of the volume P, respectively.

In relation to the momentum equation, when it comes to the flat-plate Boundary Layer Problem, it has no pressure gradient flow [Neel, 1997]. Thus, since pressure is invariant, i.e. $\frac{dp}{dx} = 0$ because the inviscid flow over a flat plate yields a constant pressure over the surface, Eq. 6.12 can be rewritten as

$$Momentum : \vec{u}\nabla \cdot \phi - \nu\frac{\partial^2\phi}{\partial y^2} = 0. \tag{6.14}$$

Integrating Eq. 6.14 in a control volume yields

$$Momentum : \int_\Omega \vec{u}\nabla \cdot \phi - \nu\frac{\partial^2\phi}{\partial y^2}d\Omega = 0. \tag{6.15}$$

Applying the Divergence Theorem tailoring to triangular control volumes the scheme described in [Gonzaga, Guedes and Kischinhevsky, 2007] and through algebraic manipulations, a semi-implicit approach in order to solve Eq. 6.15 is

$$Momentum : u_P^{k+1} - u_{i_S}^{k+1}\frac{v_P^k}{u_P^k} - u_i^{k+1}\frac{\nu\iota}{u_P^k} = u_{i_W}^k - v_P^k - \nu\iota. \tag{6.16}$$

where ι is obtained from Eq. 3.19 and i represents the neighbor volumes of P.

6.6.2 The analytical solution of the flat-plate Boundary Layer Problem

According to [Schlichting, 1979], when it comes to the flat-plate Boundary Layer Problem, there is no pressure gradient and a constant boundary-layer edge velocity occurs. For the solution of this constant property, the flat-plate flow is known as the Blasius Solution. It shows that for a flow with Reynolds number (Re) much larger than unity, i.e. $Re >> 1$, the velocity profiles have the same dimensionless shape in the boundary-layer region. More precisely, a dimensionless similarity variable in the normal direction is given by [Schlichting, 1979]

$$\eta = y\sqrt{\frac{u_\infty}{2\nu x}}, \tag{6.17}$$

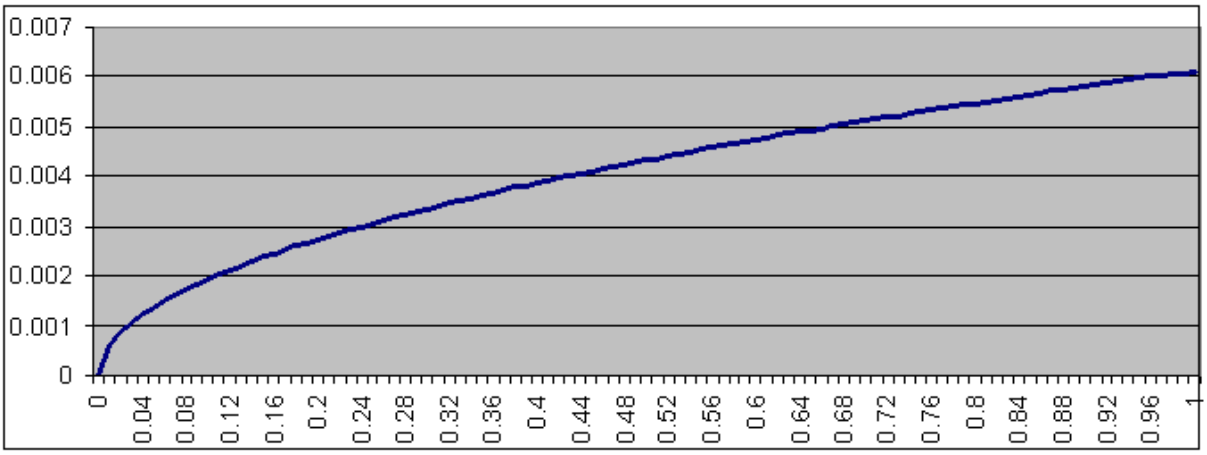

Figure 6.9: Flat-plate boundary-layer process for $\phi = 10m/s$.

where x, y are the distances from the flat-plate leading edge, u_∞ is the air velocity in the free-stream region, and ν is the air kinematic viscosity.

Subsequently, the velocity parallel to the plate is dimensionless by the edge velocity u_∞, which is also the free-stream velocity for this particular case. Briefly, the boundary-layer thickness over a flat plate is given by [Schlichting, 1979]

$$\triangle = \frac{5x}{\sqrt{\frac{u_\infty x}{\nu}}}. \tag{6.18}$$

Exemplifying, Fig. 6.9 depicts the flat-plate boundary-layer process for $\phi = 10m/s$.

6.6.3 A numerical simulation of the flat-plate Boundary Layer Problem

The numerical error applied is the difference between the numerical simulation and the Blasius Solution as the maximum norm

$$||error||_\infty = max(|A_1 - B_1|, |A_2 - B_2|, \ldots, |A_N - B_N|), \tag{6.19}$$

where A_i means the approximations, B_i means the Blasius Solution points and N is the number of volumes.

One example of the tests is depicted in Fig. 6.10 with $u_\infty = 10$, $\nu = 1.5 \cdot 10^{-5}$, $u_\infty = 10$, and $(x, y) = [0; 1]^2$.

In a test with nine levels of refinement, the generated average error was $2.77 \cdot 10^{-4}$. This error magnitude is, very probably, related to the consistency of the scheme applied.

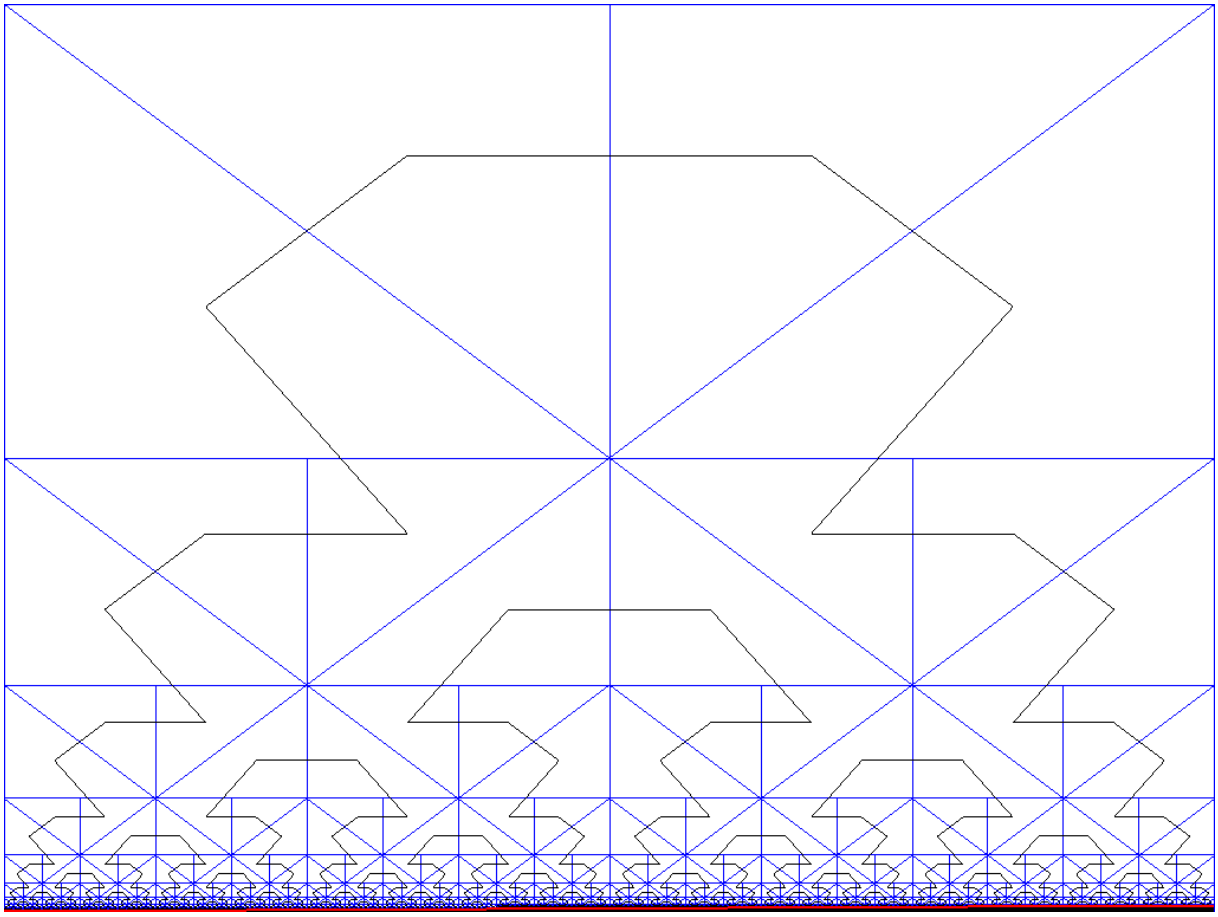

Figure 6.10: Flat-plate Boundary Layer Problem with $u_\infty = 10$, $\nu = 1.5 \cdots 10^{-5}$, $u_\infty = 10$, and $(x, y) = [0; 1]^2$.

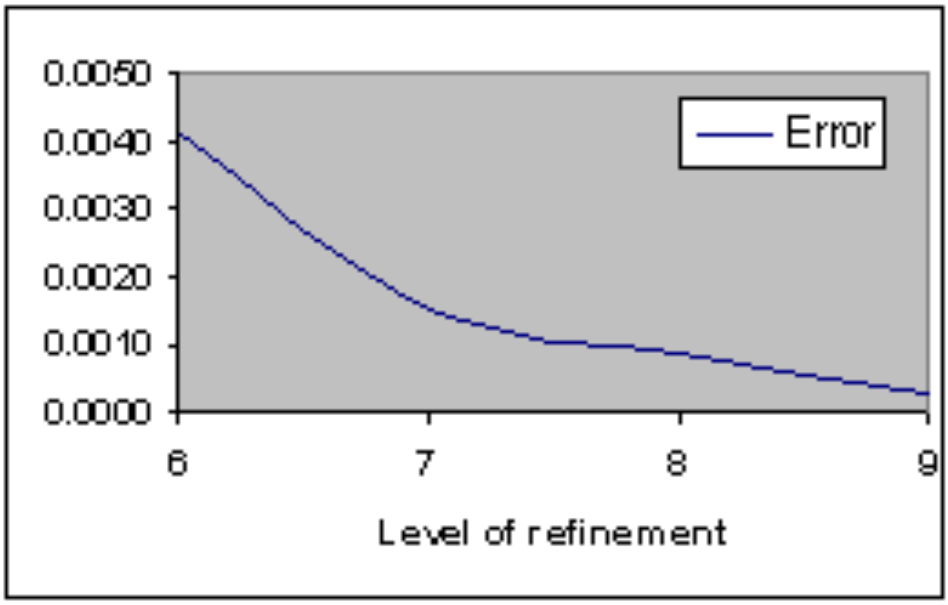

Figure 6.11: Error of the numerical approximations to the flat-plate Boundary Layer Problem with $u_\infty = 10$, $\nu = 1.5 \cdots 10^{-5}$, $u_\infty = 10$, and $(x, y) = [0; 1]^2$.

Therefore, a higher-order scheme should be applied in a future work. Figure 6.11 shows the errors of the approximations with different levels of refinement.

Finally, the total time (in seconds) spent in computations by a test presented in [Gonzaga, Guedes and Kischinhevsky, 2007] with 511 volumes was $8.75 \cdot 10^{-1}$ and the time spent in the refinement process was $2.1 \cdot 10^{-2}$ (2.4%); whereas the total time spent in computations by this present test with 521 volumes spent $4.44 \cdot 10^{-1}$ and the time spent in the refinement process was $8.46 \cdot 10^{-3}$ (1.82%).

Chapter 7

Conclusion

This work implements the Autonomous Leaves Graph (ALG) in a triangular discretization based on the Finite Volume Method for solving partial differential equations (PDEs) with non-uniform meshes through a cell-centered scheme. Triangular volumes may be more efficient to discretize complex domains than quadrangular volumes. The required data for solving a PDE are stored in a graph data structure that performs the adaptive mesh refinement (AMR). Original nodes in the local refinement are not stored. This refinement process allows that the created new pack from a single volume may be unrefined, i.e. it may return to a previous stage. This process intends to be as simple and straightforward as the ALG original technique.

In addition, this scheme seeks to have low computational cost and flexibility in ordering the mesh and also linking neighbor volumes refined in different levels of refinement in comparison with tree-based methods (quadtree and octree) and also the ALG technique. The mesh total-ordering scheme is performed by a Sierpiński-like Curve, which is implemented by means of a double-linked list. It is a non-regular version of the well-known curve due to the AMR. The refinement and unrefinement scheme of the triangular volume enables straightforward update both in the linked list for mesh volume total ordering and graph nodes as well. Moreover, the Modified Hilbert Curve is more computationally expensive than the Sierpinski-like Curve.

Since this scheme yields a generalized sequential triangulation for total ordering, it is a triangular strip-like scheme. Besides, such sequential structure presents reduced connectivity information. Therefore, this scheme may be appropriate to be used in algorithms that generate triangle strip for accelerated rendering, algorithms that compute sequential triangulations for geometry compression and also in theoretical investigation of paths on triangular meshes.

Additionally, in all tests accomplished, the linear systems presented sparse, symmetric and positive-definite coefficient matrices. It is very likely that this scheme always produces

matrices with those properties and linear-system solvers based on the minimization of functionals can be easily employed. Specifically, the Conjugate Gradient Method was employed. Notice that no pre-conditioner was used.

These efficiency results in much smaller memory requirements and larger savings in computation times during the solution phase than tree-based schemes. The relative gain in efficiency may be even more compelling for 3D problems. Since accessing the main memory of a computer system is currently a bottleneck in terms of performance, those features may be quite interesting.

Each problem is evaluated with its own features, and the most appropriate linear interpolation must be employed for the problem. A specific linear integration is used in order to construct the solution in respect to required gradients. This scheme may be applied to either a simplified least-square gradient reconstruction or a vertex-centered scheme. Moreover, the Median Dual scheme and Voronoi Diagrams (and its dual Delaunay Tesselation) may be employed.

In the refinement process, neither very large nor very small generated angles are desirable. Thus, triangles are divided through successive bisections so that the angles generated by the algorithm are necessarily bounded away from 0 and π. Moreover, firstly each edge is divided into two equal parts, and, secondly, both triangles are bisected again. As a result, four new volumes with equal area are generated. Thus, this scheme intends to present high-quality meshes during the refinement process. This scheme intends to produce graded and smoothed meshes. Namely, the area of neighboring triangles does not dramatically differ, though the scheme makes it possible through the graph data structure. New studies may demonstrate that this approach may generate high-quality meshes during refinement propagation even if the initial mesh does not present a quality mesh. In addition, this scheme may be easily employed to the Finite Element Method.

Finally, although tests show only rectangular triangular volumes, any kind of triangles can be used. Besides, this scheme can be extended to any polygonal volume shape. Furthermore, this technique enables any arbitrary polygonal shape for the initial domain, such as squares, rectangles, any kind of triangles and so on.

Tests showed numerical results in the Laplace (elliptic problem), Heat Conduction (parabolic problem), Wave (hyperbolic problem) Equations and flat-plate Boundary Layer Problem.

Certainly, there is a lot of work yet to be done in this issue, including using the graph data structure in moving meshes, implementing the 3D scheme, and in higher-order schemes, among others.

APPENDIX A – Some historical descriptions

These historical descriptions can be mainly found in Wikipédia.

The word **algebra** is named after an Arabic word from the title of a book meaning *The book of Summary Concerning Calculating by Transposition and Reduction*, a book written by Abu Ja'far Muhammad ibn M$\bar{u}$s$\bar{a}$ al-Khw$\bar{a}$rizm$\bar{i}$ (about 790 - about 840), in 820ac. The word *Al-Jabr* means *reunion*. On the other hand, Diophantus (between 200 and 214 - between 284 and 290) has traditionally been known as *the father of algebra*.

Abu Ja'far Muhammad ibn M$\bar{u}$s$\bar{a}$ al-Khw$\bar{a}$rizm$\bar{i}$ wrote a treatise in 825ac, *On Calculation with Hindu Numerals*, translated into Latin in the 12th century as *Algoritmi de numero indorum*, which title was likely intended to mean *Algoritmi on the numbers of the Indians*, where **Algoritmi** was the rendition of the author's name; but people misunderstanding the title led it to mean *calculation method*.

The **Finite Element Method** originated from the need for solving complex elasticity and structural analysis problems in civil and aeronautical engineering. Its development can be traced back to the work by Alexander Hrennikoff (1896-1984) in 1941 and Richard Courant (1888-1972) in 1942. While the approaches used by these pioneers are dramatically different, they share one essential characteristic: mesh discretization of a continuous domain into a set of discrete sub-domains, usually called elements. Hrennikoff's work discretizes the domain by using a lattice analogy whereas Courant's approach divides the domain into finite triangular subregions for solution of second order elliptic PDEs that arise from the problem of torsion of a cylinder. Courant's contribution was evolutionary, drawing on a large body of earlier results for variational formulation of PDEs. Courant is also a namesake of the Courant minimax principle.

Hamiltonian triangulation is named after Willian Rowan Hamilton (1805-1865). His studies of mechanical and optical systems led him to discover far-reaching mathematical concepts and techniques. His greatest contribution is perhaps the reformulation of Newtonian mechanics, now called Hamiltonian mechanics, through the development of action principles and the eikonal function. This work has proven to be the central to the modern study of classical field theories such as electromagnetism, and to the development

of quantum mechanics. In mathematics, he is perhaps best known for his discovery of quaternions.

Isaac Newton (1642-1727) was a precursor in **finite-differences** publishing a tract, *Methodis differentialis*, in 1711, whereas *Methodus incrementorum directa et inversa* from 1715 also qualifies Brook Taylor (1685-1731) as one of the difference-finite calculus founders. Isaac Newton's book, *Philosophiæ Naturalis Principia Mathematica*, published in 1687, is considered to be the most influential one in the history of science. In this work, Newton described universal gravitation and the three laws of motion, laying the groundwork for classical mechanics, which dominated the scientific view of the physical Universe for the next three centuries and is the basis for modern engineering. Newton showed that the motions of objects on Earth and of celestial bodies are governed by the same set of natural laws by demonstrating the consistency between Kepler's laws of planetary motion and his theory of gravitation, thus removing the last doubts about heliocentrism and advancing the scientific revolution. In mechanics, Newton enunciated the principles of conservation of momentum and angular momentum. In optics, he built the first "practical" reflecting telescope and developed a theory of color based on the observation that a prism decomposes white light into a visible spectrum. He also formulated an empirical law of cooling and studied the speed of sound. In mathematics, Newton shares the credit with Gottfried Wilhelm Leibniz (1646-1716) for the development of the differential and integral calculus. He also demonstrated the generalized binomial theorem, developed the so-called "Newton's method" for approximating the zeroes of a function, and contributed to the study of power series.

The **Divergence Theorem** is an important result for the Mathematics of Physics, in particular in Electrostatics and Fluid Dynamics. The theorem was first stated by Joseph Louis Lagrange (1736-1813) in 1762, then later described independently by Carl Friedrich Gauss (1777-1855) in 1813, by George Green (1793-1841) in 1825 and in 1831 by Mikhail Vasilievich Ostrogradsky (1801-1862), who also provided the first proof of the theorem. Born Giuseppe Lodovico Lagrangia, Lagrange made significant contributions to analysis, to number theory, and to classical and celestial mechanics. Lagrange's treatise on analytical mechanics (*Mécanique Analytique*, 4. ed., 2 vols. Paris: Gauthier-Villars et fils, 1888-89), first published in 1788, offered the most comprehensive treatment of classical mechanics since Newton and formed a basis for the development of mathematical physics. Lagrange was one of the creators of the calculus of variations, deriving the Euler-Lagrange equations for extrema of functionals. He also extended the method to take into account possible constraints, arriving at the method of Lagrange multipliers. Lagrange invented the method of solving differential equations known as variation of parameters, applied differential calculus to the theory of probabilities and attained notable work on the solution of equations. He proved that every natural number is a sum of four squares. His

treatise *Theorie des fonctions analytiques* laid some of the foundations of group theory. In calculus, Lagrange developed a novel approach to interpolation and Taylor series. He studied the three-body problem for the Earth, Sun, and Moon (1764) and the movement of Jupiter's satellites (1766), and in 1772 found the special-case solutions to this problem that are now known as Lagrangian points. But above all he impressed on mechanics, having transformed Newtonian mechanics into a branch of analysis, Lagrangian mechanics as it is now called, and exhibited the so-called mechanical "principles" as simple results of the variational calculus. Ostrogradsky worked mainly in the mathematical fields of calculus of variations, integration of algebraic functions, number theory, algebra, geometry, probability theory and in the fields of mathematical physics and classical mechanics. In the latter his most important work includes researches of the motion of an elastic body and the development of methods for integration of the equations of dynamics.

The **Green-Gauss technique** is named after Green and Gauss. Gauss contributed significantly to many fields, including number theory, statistics, analysis, differential geometry, geodesy, electrostatics, astronomy, and optics. Green wrote *An Essay on the Application of Mathematical Analysis to the Theories of Electricity and Magnetism* in 1828. The essay introduced several important concepts, among them a theorem similar to modern Green's theorem, the idea of potential functions as currently used in physics, and the concept of what are now called Green's functions. He was the first to try to explain a mathematical theory of the theories of electricity and magnetism which formed the basis for other scientists.

The **Cartesian coordinate system** allowing geometric shapes to be expressed in algebraic equations is named after René Descartes (1596-1650). He is accreditied as the father of analytical geometry.

Voronoi Diagrams was proposed by Georgy Feodosevich Voronoi (1868-1908) in 1907. He worked on continued fractions.

Dirichlet Tesselation is named after Johan Peter Gustav Lejeune Dirichlet (1805-1859), credited with the modern "formal" definition of a function.

Delaunay Tesselation, proposed in 1934, is named after Boris Nikolaevich Delone (1890-1980). He worked in the fields of modern algebra, the geometry of numbers. He also worked in the modern mathematical crystallography and general mathematical model of crystals.

Euler Equation is named after Leonhard Paul Euler (1707-1783). Euler made important discoveries in fields as diverse as calculus and graph theory. He also introduced much of the modern mathematical terminology and notation, particularly for mathematical analysis, such as the notion of a mathematical function. He is also renowned for his

work in mechanics, optics, and astronomy.

Navier-Stokes Equations, central to fluid mechanics, are named after Claude-Louis Navier (1785-1836) and George Gabriel Stokes (1819-1903). Navier formulated the general theory of elasticity in a mathematically usable form in 1821, making it available to the field of construction with sufficient accuracy for the first time. In 1819 he succeeded in determining the zero line of mechanical stress, and in 1826 he established the elastic modulus as a property of materials independent of the second moment of area. Navier is therefore often considered to be the founder of modern structural analysis. Stokes made important contributions to fluid dynamics, optics, and mathematical physics (including Stokes' theorem).

The **Courant-Friendrichs-Lewy (CFL) condition** was firstly proposed in 1928 by: Richard Courant (1888-1972), who is attached to the Finite Element Method; Kurt Friedrichs (1901-1982), who greatest contribution was his work on PDEs and; Hans Lewy (1904-1988), who was also known for his work on PDEs.

Euclidian space is named after Euclid of Alexandria (about 325bc - about 265bc). He is often referred to as the Father of Geometry. His work *Elements* is the most successful textbook in the history of mathematics. In it, the principles of what is now called Euclidean geometry were deduced from a small set of axioms. Euclid also wrote works on perspective, conic sections, spherical geometry, number theory and rigor.

Discrete Boltzmann Equation is named after Ludwig Eduard Boltzmann (1844-1906). He is famous for his founding contributions in the fields of statistical mechanics and statistical thermodynamics. He was one of the most important advocates for atomic theory when that scientific model was still highly controversial.

Generalized Riemann Problems are named after Georg Friedrich Bernhard Riemann (1826-1866). He gave fundamental contributions to the differential analysis and geometry, which some opened a path to the development of the general relativity. His name is linked to Riemann Zeta function, Riemann Lemma, Rieman Fold and the Riemann surfaces.

Taylor-series expansion are named after Brook Taylor (1685-1731). He obtained a remarkable solution of the problem of the "centre of oscillation", published in 1714. Taylor's *Methodus Incrementorum Directa et Inversa* (1715) added a new branch to the higher mathematics, now designated the "calculus of finite differences". He used it to determine the form of movement of a vibrating string, by him first successfully reduced to mechanical principles. The same work contained the celebrated formula known as Taylor's theorem, the importance of which remained unrecognized until 1772, when J. L. Lagrange realized its powers and termed it *le principal fondement du calcul différentiel*. In his 1715

essay Linear Perspective, Taylor set forth the true principles of the art in an original and more general form than any of his predecessors; but the work suffered from the brevity and obscurity which affected most of his writings.

The **upwind** scheme was first proposed, in 1952, by Richard Courant (1888-1972), Eugene Isaacson (1919-2008), one of the pioneers of modern numerical mathematics, and Mina Spiegel Rees (1902-1997), who worked in Applied Mathematics.

In 1887, Marie Ennemond Camille Jordan (1838-1922) introduced a rigorous definition, since adopted a precise description, of the *continuous* ***curve*** notion. He is known both for his foundational work in group theory and for his influential *Cours d'analyse*.

Space-filling curves or Peano Curves are curves, first described by Giuseppe Peano (1858-1932) in 1880, whose ranges contain the entire 2D unit square (or the 3D unit cube). He was author of over 200 books and papers. He was a founder of mathematical logic and set theory, to which he contributed with much notation. The standard axiomatization of the natural numbers is named in his honor. As part of this axiomatization effort, he made key contributions to the modern rigorous and systematic treatment of the method of mathematical induction.

Wacław Franciszek Sierpiński (1882-1969) was known for outstanding contributions to set theory (research on the axiom of choice and the continuum hypothesis), number theory, theory of functions and topology. He published over 700 papers and 50 books. Three well-known fractals are named after him (the Sierpiński triangle, the Sierpiński carpet and the **Sierpiński curve**), as are Sierpiński numbers and the associated Sierpiński problem.

Jacques Salomon Hadamard (1865-1963) was known for his proof of the prime number theorem in 1896. He introduced the idea of **well-posed problem** in the theory of partial differential equations. He also gave his name to the Hadamard inequality on volumes, and the Hadamard matrix, on which the Hadamard transform is based. The Hadamard gate in quantum computing uses this matrix.

David Hilbert (1862-1943) proposed the **Hilbert Curve**, a space-filling curve, in 1891. He is recognized as one of the most influential and universal mathematicians of the 19th and early 20th centuries. He invented or developed a broad range of fundamental ideas in many areas, including invariant theory and the axiomatization of geometry. He also formulated the theory of Hilbert spaces, one of the foundations of functional analysis. Hilbert adopted and warmly defended the set theory and transfinite numbers. A famous example of his leadership in mathematics is his 1900 presentation of a collection of problems that set the course for much of the mathematical research of the 20th century. Hilbert and his students supplied significant portions of the mathematical infrastructure required for quantum mechanics and general relativity. He is also known as one of the

founders of proof theory, mathematical logic and the distinction between mathematics and metamathematics.

Boundary layers have been of great importance in the study of viscous fluid flow. In 1904, Ludwig Prandtl (1875-1953) demonstrated the existence of a thin boundary layer in fluid flow. Moreover, he found that there exists a thin layer near an object surface, where the viscous aerodynamic forces are as important as the inertial forces. He was a pioneer of aerodynamics, and developed the mathematical basis for the fundamental principles of subsonic aerodynamics in the 1920s. His studies identified the boundary layer, thin-airfoils, and lifting-line theories. The Prandtl number was named after him.

Blasius Solution is named after Paul Richard Heinrich Blasius (1883-1970). One of his contributions involves a description of the steady two-dimensional boundary-layer that forms on a semi-infinite plate which is held parallel to a constant unidirectional flow.

Reynolds number is named after Osborne Reynolds (1842-1912). He was a prominent innovator in the understanding of fluid dynamics. Separately, his studies of heat transfer between solids and fluids brought improvements in boiler and condenser design.

APPENDIX B – Finite volume approximations

Consider an elementar 2D control volume P represented in Fig. B.1 (adapted from [Sperandio, Mendes and Silva, 2003]), where e, w, n and s identify the control volume edges in the discretized domain.

Consider the mass conservation equation in steady state

$$\rho \nabla \cdot \phi = 0. \tag{B.1}$$

According to [Sperandio, Mendes and Silva, 2003], finite volume approximations can be obtained integrating the partial differential equation (PDE) in the conservative form on an elementar control volume. The PDE is integrated in an elementar volume

$$\int_s^n (\rho u|_e - \rho u|_w) dy + \int_w^e (\rho v|_n - \rho v|_s) dx = 0, \tag{B.2}$$

where u and v are the horizontal and vertical components of the vector field, respectively.

Admiting that the mass flux in the mean of the control volume edge represents the average of its edge variation yields

$$\rho u \Delta y|_e - \rho u \Delta y|_w + \rho v \Delta x|_n - \rho v \Delta x|_s = 0, \tag{B.3}$$

which is, in steady state, the mass balance in the elementar volume. Equation B.3 is valid for volume P. Integrating all elementar volumes, an algebraic equation system is obtained. The reader is referred to [Sperandio, Mendes and Silva, 2003] and
[Versteeg and Malalasekera, 1995] for a further reading.

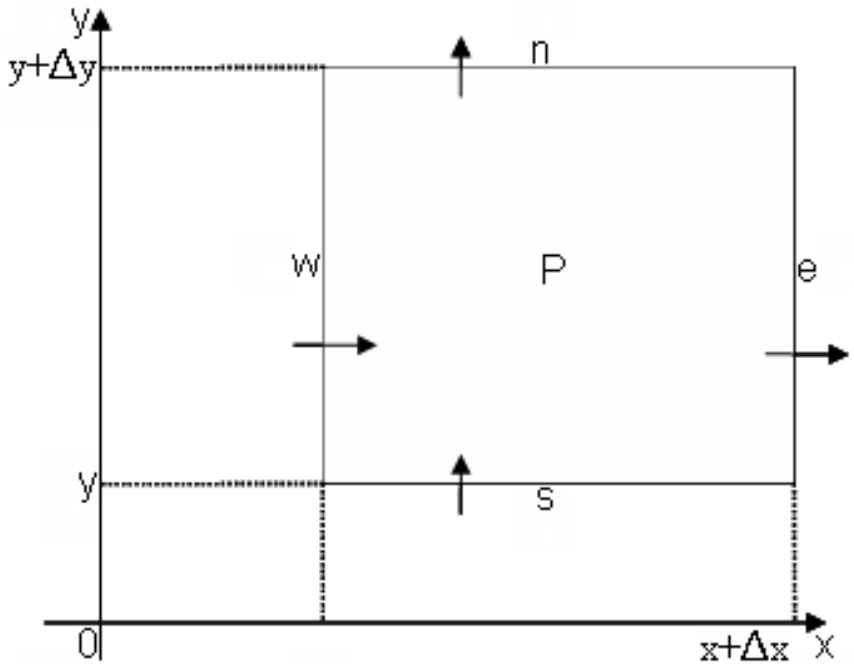

Figure B.1: Elementar 2D control volume.

APPENDIX C – A mathematical proof by induction

$$Theorem : [\forall n \in \mathbb{N} | n \ even \wedge 2^n = \sum_{i=1}^{2^{\frac{n}{2}}} (2i-1)] \tag{C.1}$$

The theorem C.1 states that for all n natural even, 2^n is the sum of odd numbers from 1 to $2^{\frac{n}{2}}$.

$$\begin{cases} 2^0 = 1 \\ 2^2 = 1+3 \\ 2^4 = 1+3+5+7 \\ 2^6 = 1+3+5+7+9+11+13+15 \\ 2^8 = 1+3+5+7+9+11+13+15+\ldots+31 \\ \cdots \\ 2^n = \sum_{i=1}^{2^{\frac{n}{2}}} (2i-1). \end{cases}$$

Proof: for this demonstration, the induction basis is

$$P(0) : 2^0 = 1 \tag{C.2}$$

or

$$P(2) : 2^2 = 1+3. \tag{C.3}$$

The inductive hypothesis is

$$P(k) : 2^k = \sum_{i=1}^{2^{\frac{k}{2}}} (2i-1). \tag{C.4}$$

The inductive step is

$$P(k+2) : 2^{k+2} \stackrel{?}{=} \sum_{i=1}^{2 \cdot 2^{\frac{k}{2}}} (2i-1) \tag{C.5}$$

or

$$P(k+2): 4 \cdot 2^k \stackrel{?}{=} 4 \cdot \underbrace{\sum_{i=1}^{2^{\frac{k}{2}}}(2i-1)}_{2^k} \stackrel{HI}{=} \underbrace{\sum_{i=1}^{2\cdot 2^{\frac{k}{2}}}(2i-1)}_{4\cdot 2^k?}. \tag{C.6}$$

The righ-hand side summation of C.6 can be decomposed in

$$\sum_{i=1}^{2^{\frac{k}{2}+1}}(2i-1) = \underbrace{\sum_{i=1}^{2^{\frac{k}{2}}}(2i-1)}_{2^k} + \underbrace{\sum_{i=2^{\frac{k}{2}}+1}^{2^{\frac{k}{2}+1}}(2i-1)}_{3\cdot 2^k?}. \tag{C.7}$$

By the inductive hypothesis, C.4 is assumed true. Then, the first summation on right-hand side of C.7 is 2^k. Thus, the second summation on right-hand side of C.7 must be verified if it is equal to $3 \cdot 2^k$.

One can define

$$Lemma: [\forall n \in \mathbb{N}^* | n \ even \wedge 2^n = \frac{\sum_{i=2^{\frac{n}{2}}+1}^{2^{\frac{n}{2}+1}}(2i-1)}{3}]. \tag{C.8}$$

Proof: for the sum $\underbrace{\sum_{i=2^{\frac{k}{2}}+1}^{2\cdot 2^{\frac{k}{2}}}(2i-1)}_{3\cdot 2^k?}$, a variable transformation from i to j is performed

$$j = i - 2^{\frac{k}{2}}. \tag{C.9}$$

Thus,

$$i = j + 2^{\frac{k}{2}}. \tag{C.10}$$

For the initial limit of the summation in j

$$j = 2^{\frac{k}{2}} + 1 - 2^{\frac{k}{2}} = 1. \tag{C.11}$$

For the final limit of the summation in j

$$j = 2 \cdot 2^{\frac{k}{2}} - 2^{\frac{k}{2}} = 2^{\frac{k}{2}}. \tag{C.12}$$

Thus,

$$\underbrace{\sum_{j=1}^{2^{\frac{k}{2}}}[2(j+2^{\frac{k}{2}})-1]}_{3\cdot 2^k} = \underbrace{\sum_{j=1}^{2^{\frac{k}{2}}}(2j-1)}_{2^k} + 2\cdot 2^k = 3\cdot 2^k, \tag{C.13}$$

which verifies and proves C.8. Therefore, substituting C.13 in C.7 and afterwards substituting in C.6, the validity of *P(k+2)* is verified and proves that the equation in C.1 is true for any natural even number, concluding the demonstration. □

Corollary: The sum of the first half of the odd number sum in $\{\forall n \in \mathbb{N}^* | n \ even\}$ of 2^n is three times smaller than the second half of the sequence.

Consider

$$\begin{cases} 2^2 = 1+3 \\ 2^4 = \underbrace{1+3}_{4} + \underbrace{5+7}_{12} \\ 2^6 = \underbrace{1+3+5+7}_{16} + \underbrace{9+11+13+15}_{48} \\ 2^8 = \underbrace{1+3+5+7+9+11+13+15}_{64} + \underbrace{17+19+\ldots+31}_{192} \\ \ldots \\ 2^n = \underbrace{\sum_{i=1}^{2^{\frac{n}{2}-1}}(2i-1)}_{\frac{1}{4}\cdot 2^n} + \underbrace{\sum_{i=2^{\frac{n}{2}-1}+1}^{2^{\frac{n}{2}}}(2i-1)}_{\frac{3}{4}\cdot 2^n}. \end{cases}$$

$$\begin{cases} Result: [\forall n \in \mathbb{N}^* | n \ even \wedge \\ \frac{1}{4}\cdot 2^n = \sum_{i=1}^{2^{\frac{n}{2}-1}}(2i-1) \wedge \\ \frac{3}{4}\cdot 2^n = \sum_{i=2^{\frac{n}{2}-1}+1}^{2^{\frac{n}{2}}}(2i-1) \therefore \\ 3\cdot \underbrace{\sum_{i=1}^{2^{\frac{n}{2}-1}}(2i-1)}_{\frac{1}{4}\cdot 2^n} = \underbrace{\sum_{i=2^{\frac{n}{2}-1}+1}^{2^{\frac{n}{2}}}(2i-1)]}_{\frac{3}{4}\cdot 2^n} \end{cases} \tag{C.14}$$

Proof: in order to demonstrate

$$\frac{1}{4}\cdot 2^n \stackrel{?}{=} \sum_{i=1}^{2^{\frac{n}{2}-1}}(2i-1), \tag{C.15}$$

the basis induction is

$$P(2): \frac{1}{4} \cdot 2^2 = \sum_{i=1}^{2^{\frac{2}{2}-1}} (2i-1) = 1. \tag{C.16}$$

The inductive hypothesis is

$$P(k): \frac{1}{4} \cdot 2^k = \sum_{i=1}^{2^{\frac{k}{2}-1}} (2i-1). \tag{C.17}$$

One must verify if

$$\frac{1}{4} \cdot 2^k \stackrel{?}{=} \frac{1}{4} \cdot \underbrace{\sum_{i=1}^{2^{\frac{k}{2}}} (2i-1)}_{2^n} = 2^{-2} \cdot \underbrace{\sum_{i=1}^{2^{\frac{k}{2}}} (2i-1)}_{2^n} \tag{C.18}$$

or

$$2^{k-2} \stackrel{HI}{=} \underbrace{\sum_{i=1}^{2^{\frac{k}{2}-1}} (2i-1)}_{\frac{1}{4} \cdot 2^n ?}. \tag{C.19}$$

The inductive step

$$P(k+2): 2^{k-2+2} \stackrel{?}{=} \sum_{i=1}^{2 \cdot 2^{\frac{k}{2}-1}} (2i-1) = \sum_{i=1}^{2^{\frac{k}{2}-1+1}} (2i-1) \tag{C.20}$$

is the theorem C.1 already proved: $2^n = \sum_{i=1}^{2^{\frac{n}{2}}} (2i-1)$. Consequently, it proves that C.15 is true.

In order to prove $\frac{3}{4} \cdot 2^n = \sum_{i=2^{\frac{n}{2}-1}+1}^{2^{\frac{n}{2}}} (2i-1)$, a variable transformation from i to j is performed

$$j = i - 2^{\frac{k}{2}-1}. \tag{C.21}$$

Thus,

$$i = j + 2^{\frac{k}{2}-1}. \tag{C.22}$$

For the initial limite of the summation in j

$$j = 2^{\frac{k}{2}-1} + 1 - 2^{\frac{k}{2}-1} = 1. \tag{C.23}$$

For the final limit of the summation in j

$$j = 2^{\frac{k}{2}} - \frac{1}{2} \cdot 2^{\frac{k}{2}} = 2^{\frac{k}{2}-1}. \tag{C.24}$$

Thus,

$$\sum_{j=1}^{2^{\frac{k}{2}-1}} [2(j + 2^{\frac{k}{2}-1}) - 1] = \underbrace{\sum_{j=1}^{2^{\frac{k}{2}-1}} (2j-1)}_{\frac{1}{4}\cdot 2^k} + \underbrace{2 \cdot (2^{-1} \cdot 2^{\frac{k}{2}}) \cdot (2^{\frac{k}{2}} \cdot 2^{-1})}_{(2\cdot 2^{-1})\cdot \underbrace{2^k \cdot 2^{-1}}_{\frac{2}{4}\cdot 2^k}} = \frac{3}{4} \cdot 2^k, \tag{C.25}$$

which verifies and proves that C.14 is true, concluding the demonstration. □

APPENDIX D – Remarks for 3D

This scheme allows tetrahedral volumes to be straightforwardly refined and ordered. Figure D.1 sketches a tetrahedral volume and its faces highlighted. A tetrahedral volume presents 12 external angles (three angles for each four faces).

Figure D.2 sketches a tetrahedral refinement. Similarly to the 2D case, the largest edge (of the largest face) is chosen for refinement. Assume that the face with the largest area is ACD and the edge $\overline{AD}$ is the longest-edge of the tetrahedral shown in Fig. D.2a. Locating the mean point E of the edge $\overline{AD}$, the medians $\overline{BE}$ and $\overline{CE}$ are traced.

The triangular face BCE divides the tetrahedral in two new ones which each one has the half-volume of the original tetrahedral. Figures D.2b-d display the two new tetrahedra. Figures D.2e-g depict both tetrahedra divided a second time. The four new tetrahedra presents each one the fourth-volume of the original tetrahedral.

This scheme is similar to the 4-triangles largest side partition of the largest face ACD, where point E is the mean point of the largest edge. This is followed by tracing the medians of the three remaining faces: $\overline{BE}$, $\overline{BF}$ and $\overline{BG}$, since the vertex B is the opposite vertex of mean points E, F and G in the faces ABD, ABC and BCD, respectively.

Figure D.3a sketches three tetrahedral volumes $ABCD$, $ABCF$ and $BCFG$, which

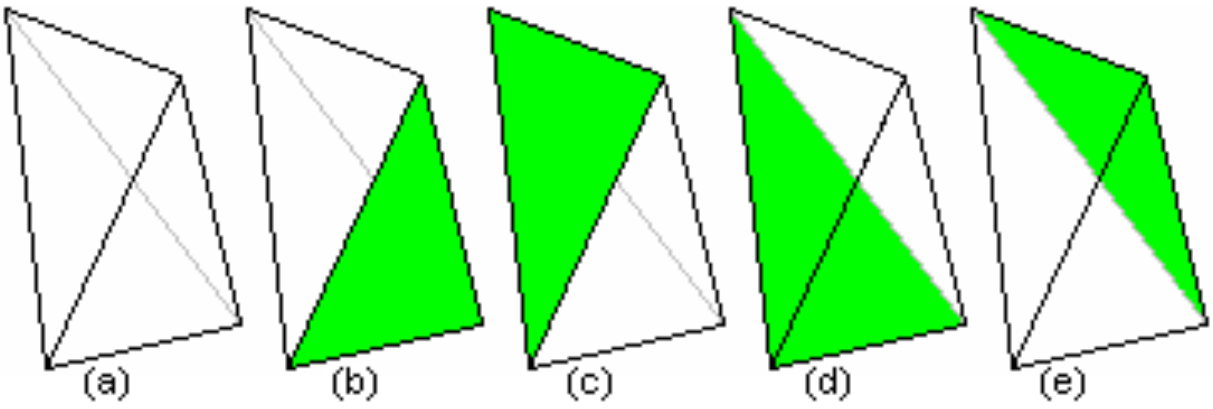

Figure D.1: (a) Tetrahedral volume with view from top; (b) detail of the front face view; (c) detail of the left front face view; (d) detail of the background face and; (e) detail of the right face.

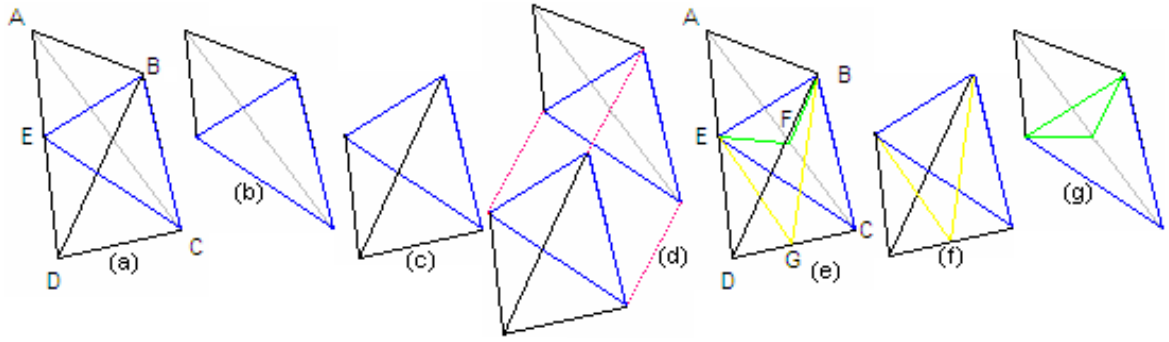

Figure D.2: (a) original tetrahedral; (b) new tetrahedral, which is *in the back*; (c) new tetrahedral, which is *in front*; (d) both new tetrahedral uncoupled; (e) four new tetrahedral; (f) first two tetrahedra and (g) last two tetrahedra.

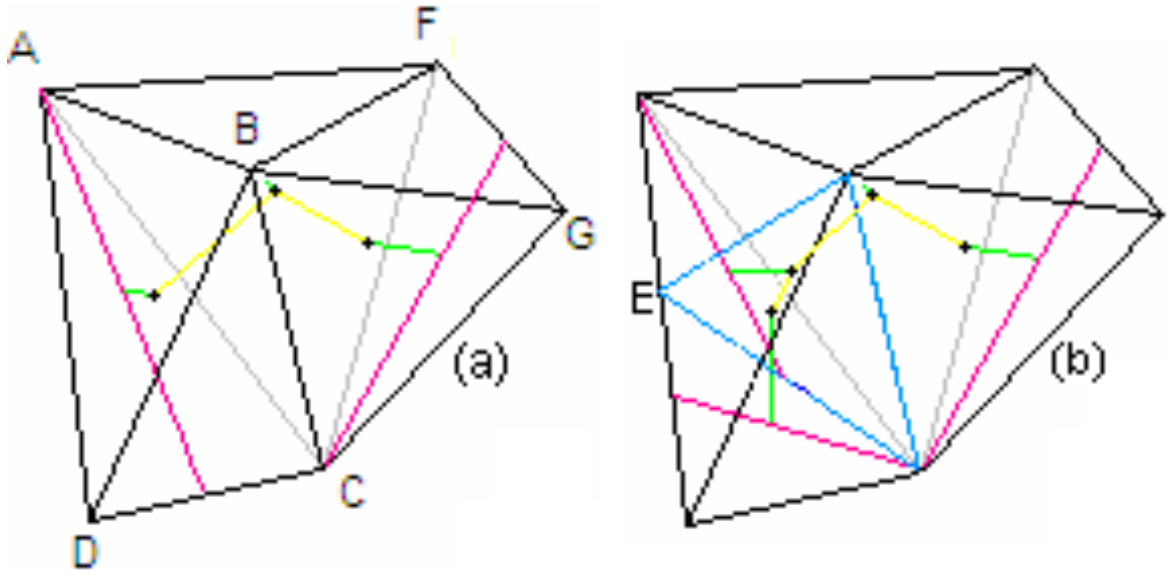

Figure D.3: (a) Three tetrahedral volumes (GFBC, FACB and CBAD); (b) refinement of the front left tetrahedral volume into two new ones where black points indicate the tetrahedral volume centroids and there is a curve representing the 3D Sierpiński-like Curve as well as lines indicating median lines of the background of tetrahedral volumes.

are linked by a 3D Sierpiński-like Curve. The tetrahedral volume on left is refined and the 3D Sierpiński-like Curve is updated in Fig. D.3b. Each new tetrahedral has the half-volume of the original tetrahedral. As a simple example, it is parted in just two tetrahedra and the new partition of both could follow the scheme shown. Solution of this scheme should follow the Frink's reconstruction [Frink, Parikh and Pirzadeh, 1991].

Bibliography

[Anderson, Tannehill and Pletcher, 1984]ANDERSON, D. A.; TANNEHILL, J. C.; PLETCHER, R. H. *Computational Fluid Mechanics and Heat Transfer*: Hemisphere, 1984.

[Arkin et al., 1994]ARKIN, E.; HELD, M.; MITCHEL, J.; SKIENA, S. Hamiltonian triangulations for fast rendering. *The 2nd Annual European Symposium on Algorithms, (LNCS)*, v. 855, p. 36–47, . Springer, Berlin Heidelberg New York 1994.

[Arnold, 2001]ARNOLD, D. N. *A Concise Introduction to Numerical Analysis*: http://www.ima.umn.edu/ arnold/, 2001.

[Azevedo and Korzenowsk, 1999]AZEVEDO, L. F. F. da Silvaand João L. F.; KORZENOWSK, H. On the development of an unstructured grid solver for inert and reactive high speed flow simulations. *J. Braz. Soc. Mech. Sci.*, v. 21, n. 4, 1999.

[Baliga and Patankar, 1980]BALIGA, B.; PATANKAR, S. V. A new finite element formulation for convection diffusion problems. *Numerical Heat Transfer*, v. 3, p. 393–409, 1980.

[Barth, 1993]BARTH, T. J. Recent developments in high order k-exact reconstruction on unstructured meshes. *31st Aerospace Sciences Meeting and Exhibit*, January 1993.

[Barth, 1994]BARTH, T. J. *Aspects of unstructured grids and finite-volume solvers for the Euler and Navier-Stokes Equations*: VKI Lecture Series, 1994.

[Barth and Ohlberger, 2004]BARTH, T. J.; OHLBERGER, M. *Finite Volume Methods: foundation and analysis*: John Wiley and Sons, 2004.

[Berger, Aftosmis and Melton, 1998]BERGER, M. J.; AFTOSMIS, M.; MELTON, J. Accuracy, adaptive methods and complex geometry. *Proc. First AFOSR Conference on Dynamic Motion CFD*, 1998.

[Berger, Helzel and LeVeque, 2003]BERGER, M. J.; HELZEL, C.; LEVEQUE, R. J. *H*-box methods for the approximation of hyperbolic conservation laws on irregular grids. *SIAM Journal on Numerical Analysis*, v. 41, n. 3, p. 893–918, 2003.

[Bertolazzi and Manzini, 2008]BERTOLAZZI, E.; MANZINI, G. *Polynomial reconstructions and limiting strategies in finite volume approximations*: http://historical.ncstrl.org/tr/fulltext/tr/ercimcnr/2002-1289.txt, 2008.

[Bouche, Ghidaglia and Pascal, 2005]BOUCHE, D.; GHIDAGLIA, J.-M.; PASCAL, F. Error estimate and the geometric corrector for the upwind Finite Volume Method applied to the linear advection equation. *SIAM J. Numer. Anal.*, v. 43, n. 2, p. 578–603, 2005.

[Bramkamp, Ballmann and Muller, 2000]BRAMKAMP, F.; BALLMANN, J.; MULLER, S. Development of a flow solver employing local adaptation based on multiscale analysis on b-spline grids. *Proceedings of 8th Annual Conf. of the CFD Society of Canada*, June 2000.

[Bramkamp et al., 2003]BRAMKAMP, F. et al. H-adaptive multiscale schemes for the compressible Navier-Stokes Equations - polyhedral discretization, data compression and mesh generation. *Numerical Notes on Fluid Mechanics*, v. 84, p. 125–204, 2003.

[Bramkamp, Lamby and Muller, 2004]BRAMKAMP, F.; LAMBY, P.; MULLER, S. An adaptive multiscale finite volume solver for unsteady and steady state flow computations. *Journal of Computational Physics*, v. 197, p. 460–490, 2004.

[Brandao, Gonzaga and Kischinhevsky, 2008]BRANDAO, D. N.; GONZAGA, S. L. de O.; KISCHINHEVSKY, M. Finite element h-adaptive refinement procedure based on autonomous leaves graph. *XXIX CILAMCE - Iberian Latin American Congress on Computational Methods in Engineering*, November 2008.

[Brandao, Gonzaga and Kischinhevsky, 2009]BRANDAO, D. N.; GONZAGA, S. L. de O.; KISCHINHEVSKY, M. Finite-element non-conforming h-adaptivity strategy based on autonomous leaves graph (to appear). *International Conference on Computational Science. LNCS,*, May 2009.

[Burgarelli, Kischinhevsky and Biezuner, 2006]BURGARELLI, D. D.; KISCHINHEVSKY, M.; BIEZUNER, R. J. A new adaptive mesh refinement strategy for numerically solving evolutionary PDE's. *Journal of Computational and Applied Mathematics*, v. 196, p. 115–131, 2006.

[Carpenter et al., 1989]CARPENTER, R. L. J.; DROEGEMEIER, K. K.; WOODWARD, P. R.; HANE, C. E. Application of the piecewise parabolic method (PPM) to metereological modeling. *Monthly Weather Review*, v. 118, p. 586–612, 1989.

[Colella and Woodward, 1984]COLELLA, P.; WOODWARD, P. R. The piecewise parabolic method (PPM) for gas-dynamical simulations. *Journal of Computational Physics*, v. 54, p. 174–201, 1984.

[Cormen et al., 2001]CORMEN, T. H.; LEISERSON, C. E.; RIVEST, R. L.; STEIN, C. *Introduction to Algorithms*: McGraw-Hill Book Company, 2001.

[Douglas and Russel, 1982]DOUGLAS, J.; RUSSEL, T. Diffusion problems on combining the method of characteristics with finite element or finite difference procedures. *SIAM J. Numer. Anal.*, v. 19, p. 871–885, 1982.

[Fonseca, 2005]FONSECA, L. C. M. da. *Estudo comparativo de esquemas para o cálculo de trajetórias no método semi-lagrangeano*: Dissertation, Universidade Federal do Rio de Janeiro, 2005.

[Frink, 1994]FRINK, N. T. Recent progress toward a three-dimensional unstructured Navier-Stokes flow solver. *32nd Aerospace Sciences Meeting*, n. 0061, January 1994.

[Frink, 1996]FRINK, N. T. Assessment of an unstructured-grid method for predicting 3-D turbulent viscous flows. *34th Aerospace Sciences Meeting*, n. 0292, January 1996.

[Frink, Parikh and Pirzadeh, 1991]FRINK, N. T.; PARIKH, P.; PIRZADEH, S. A fast upwind solver for the Euler Equations on three-dimensional unstructured meshes. *AIAA*, n. 0102, 1991.

[Furst, 2006]FURST, J. A weighted least square scheme for compressible flows. *Applied Scientific Research*, v. 76, n. 4, p. 331–342, June 2006.

[Galante, 2006]GALANTE, G. *Métodos Multigrid Paralelos Aplicados à Simulação de Problemas de Dinâmica de Fluidos Computacional*: Universidade Federal do Rio Grande do Sul, 2006.

[Gonzaga, Guedes and Kischinhevsky, 2007]GONZAGA, S. L. de O.; GUEDES, M.; KISCHINHEVSKY, M. Método dos Volumes Finitos com refinamento adaptativo de malhas aplicado ao problema da camada limite. *XXIIX CILAMCE - Iberian Latin American Congress on Computational Methods in Engineering*, June 2007.

[Gonzaga and Kischinhevsky, 2008]GONZAGA, S. L. de O.; KISCHINHEVSKY, M. Sierpinski curve for total ordering of a graph-based adaptive simplicial-mesh refinement for finite volume discretizations. *XXXI CNMAC - Congresso Nacional de Matemática Aplicada e Computacional*, September 2008.

[Gonzaga and Kischinhevsky, 2009]GONZAGA, S. L. de O.; KISCHINHEVSKY, M. Autonomous leaves graph applied to the boundary layer problem (to appear). *International Conference on Computational Science. LNCS,*, May 2009.

[Gonzaga et al., 2008]GONZAGA, S. L. de O.; KISCHINHEVSKY, M.; BURGARELLI, D. D.; BIEZUNER, R. Graph-based adaptive simplicial-mesh refinement for finite volume discretizations. *XXIX CILAMCE - Iberian Latin American Congress on Computational Methods in Engineering*, October 2008.

[Guibas and Stolfi, 1985]GUIBAS, L.; STOLFI, J. Primitives for the manipulation of general subdivisions and the computation of Voronoi Diagrams. *ACM Transactions on Graphics*, v. 4, n. 2, p. 74–123, 1985.

[Harten, 1983]HARTEN, A. High resolution schemes for hyperbolic conservation laws. *J. Comput. Phys.*, v. 49, n. 197, p. 357–293, 1983.

[Helzel, Berger and LeVeque, 2005]HELZEL, C.; BERGER, M. J.; LEVEQUE, R. J. A high-resolution rotated grid method for conservation laws with embedded geometries. *SIAM Journal on Scientific Computing*, v. 26, n. 3, p. 785–809, 2005.

[Hestenes and Stiefel, 1952]HESTENES, M. R.; STIEFEL, E. L. Methods of conjugate gradients for solving linear systems. *J.Res.Nat.Bur.Stand.*, v. 49, p. 409–436, 1952.

[Iaccarino and Ham, 2004]IACCARINO, G.; HAM, F. Automatic mesh generation for les in complex geometries. *Annual Research Briefs 2005 - Center for Turbulence Research*, v. 197, p. 460–490, 2004.

[Iske and Kaser, 2004]ISKE, A.; KASER, M. Conservative semi-lagrangian advection on adaptive unstructured meshes. *Numer. Meth. Par. Diff. Eq.*, v. 20, n. 3, p. 388–411, 2004.

[Jeon and Sheen, 2005]JEON, Y.; SHEEN, D. Analysis of a cell Boundary Element Method. *Advances in Computational Mathematics*, v. 22, p. 201–222, 2005.

[Jones and Plassmann, 1997]JONES, M.; PLASSMANN, P. Adaptive refinement of unstructured finite-element meshes. *Journal of Finite Elements in Analysis and Design*, v. 25, n. 1-2, p. 41–60, 1997.

[Ju, 2004]JU, L. Conforming centroidal Voronoi Delaunay triangulation for quality mesh generation. *International Journal of Numerical Analysis and Modeling*, v. 1, n. 1, p. 1–17, 2004.

[Kaser and Iske, 2005]KASER, M.; ISKE, A. Ader schemes on adaptive triangular meshes for scalar conservation laws. *Journal of Computational Physics*, v. 205, p. 486–508, 2005.

[Kim and Choi, 2000]KIM, D.; CHOI, H. A second-order time-accurate Finite Volume Method for unsteady incompressible flow on hybrid unstructured grids. *Journal of Computational Physics*, v. 162, p. 411–428, 2000.

[Kobayashi, Pereira and Pereira, 1999]KOBAYASHI, M.; PEREIRA, J.; PEREIRA, J. A conservative finite-volume second-order-accurate projection method on hybrid unstructured grids. *Journal of Computational Physics*, v. 150, p. 40–75, 1999.

[Leithold, 1994]LEITHOLD, L. *O cálculo com Geometria Analítica (3a. ed.)*: Editora Harbra, 1994.

[Madsen, 1998]MADSEN, J. *Design optimization of internal flow devices*. Thesis (Ph.D.), 1998.

[Márquez et al., 2008]MÁRQUEZ, A.; MORENO-GONZÁLEZ, A.; PLAZA, A.; SUÁREZ, J. P. The seven-triangle longest-side partition of triangles and mesh quality improvement. *Finite Elements in Analysis and Design*, v. 44, p. 748–758, July 2008.

[Mathur and Murthy, 1997]MATHUR, S.; MURTHY, J. A pressure based method for unstructured meshes. *Numerical Heat Transfer part B*, v. 31, p. 195–215, 1997.

[McManus et al., 2000]MCMANUS, K. et al. A scalable strategy for the paralelization of multiphysics unstructured mesh-iterative codes on distributed-memory systems. *The International Journal of High Performance Computing Applications*, v. 4, n. 14, p. 137–174, 2000.

[Moura, 2002]MOURA, C. A. *Análise Funcional para aplicações - Posologia*: Editora Ciência Moderna, 2002.

[Neel, 1997]NEEL, R. *Advances in computational fluid dynamics: turbulent separated flows and transonic potential flows*: Dissertation, Faculty of Virginia Polytechnic Institute and State University, 1997.

[Northrup, 2004]NORTHRUP, S. A. *A Parallel Adaptive-Mesh Refinement Scheme for Predicting Laminar Diffusion Flames*: Dissertation, University of Toronto, 2004.

[OpenGL, 1993]OPENGL, A. R. B. *OpenGL Reference Manual*: Addison Wesley, 1993.

[Ozawa and Tanahashi, 2005]OZAWA, T.; TANAHASHI, T. CIVA (Cubic Interpolation with Volume/Area Coordinates) and AMR (Adaptive Mesh Refinement) Method for Discrete Boltzmann Equation. *JSME International Journal, Series B*, v. 48, n. 2, p. 229–234, 2005.

[Pandya and Frink, 2004]PANDYA, M. J.; FRINK, N. T. An agglomeration multigrid for an unstructured-grid flow solver. *42nd AIAA Aerospace Sciences Meeting and Exhibit*, n. 0759, January 2004.

[Pantano et al., 2007]PANTANO, C.; DEITERDING, R.; HILL, D.; PULLIN, D. A low numerical dissipation patch-based adaptive mesh refinement method for large-eddy simulation of compressible flows. *Journal of Computational Physics*, v. 221, p. 63–87, 2007.

[Pascal and Ghidaglia, 2001]PASCAL, F.; GHIDAGLIA, J.-M. Footbridges between finite volumes and finite elements with applications to CFD. *Int. J. for Numer. Meth. in Fluids*, v. 37, p. 951–986, 2001.

[Rivara, 1984a]RIVARA, M. C. Design and data structure of fully adaptive, multigrid, finite-element software. *ACM Transactions on Mathematical Software*, v. 10, n. 3, p. 242–264, September 1984a.

[Rivara, 1984b]RIVARA, M. C. Algorithms for refining triangular grids suitable for adaptive and multigrid techniques. *Int. J. for Num. Meth. in Eng.*, v. 20, p. 745–756, 1984b.

[Rivara, 1984c]RIVARA, M. C. Mesh refinement processes based on the generalized bisection of simplices. *SIAM Journal on Numerical Analysis*, v. 21, n. 3, p. 604–613, June 1984c.

[Rivara and Iribarren, 1996]RIVARA, M. C.; IRIBARREN, G. The 4-triangles longest-side partition of triangles and linear refinement algorithms. *Mathematics of Computation*, v. 65, n. 216, p. 1485–1502, November 1996.

[Sachdev, Groth and Gottlieb, 2005]SACHDEV, J.; GROTH, C.; GOTTLIEB, J. Parallel AMR scheme for turbulent multi-phase rocket motor core flows. *17th AIAA Computational Fluid Dynamics Conference*, n. 5334, p. 6–9, June 2005.

[Schlichting, 1979]SCHLICHTING, H. *Boundary-Layer Theory*: McGraw-Hill Inc., 1979.

[Schneider and Maliska, 2002]SCHNEIDER, F. A.; MALISKA, C. R. Solução numérica de problemas convectivos-difusivos bidimensionais pelo Método dos Volumes Finitos usando malhas não estruturadas. *IX Congresso Brasileiro de Engenharias e Ciências Térmicas*, n. 0346, 2002.

[Schneider and Raw, 1987]SCHNEIDER, G.; RAW, M. Control volume Finite Element Method for heat transfer and fluid flow using co-located variables 2: Application and validation. *Numerical Heat Transfer*, v. 11, p. 391–400, 1987.

[Shewchuk, 1999]SHEWCHUK, J. R. *Lecture Notes on Delaunay Mesh Generation*: University of California at Berkeley, 1999.

[Sierpiński, 1912]SIERPIŃSKI, W. Sur une nouvelle courbe continue qui remplit toute une aire plane. *Bull. l'Acad. des Sciences Cracovie A*, p. 462–478, 1912.

[Sierpiński, 1916]SIERPIŃSKI, W. Sur une courbe cantorienne qui contient une image biunivoque et continue de toute courbe donee. *Comptes Rendus Acad. Sci.*, v. 162, p. 629–632, 1916.

[Speckmann and Snoeyink, 1997]SPECKMANN, B.; SNOEYINK, J. Easy triangle strips for TIN terrain models. *Canadian Conference on Computational Geometry*, p. 239–244, 1997.

[Sperandio, Mendes and Silva, 2003]SPERANDIO, D.; MENDES, J. ao T.; SILVA, L. H. M. e. *Cálculo Numérico: Características Matemáticas e Computacionais dos Métodos Numéricos*: Pearson Prentice Hall, 2003.

[Stynes, 1980]STYNES, M. On faster convergence of the bisection method for all triangles. *Mathematics of Computation*, v. 35, n. 152, p. 1195–1201, October 1980.

[Tanaka, 1999]TANAKA, N. Development of a highly accurate interpolation method for mesh-free flow simulations i. integration of gridless, particle and CIP methods. *J. Numer. Methods Fluids*, v. 30, n. 8, p. 957–976, 1999.

[Tanaka, Yamasaki and Tagushi, 2004]TANAKA, N.; YAMASAKI, T.; TAGUSHI, T. Accurate and robust fluid analysis using cubic interpolation with volume/area coordinates (CIVA) method on unstructured grids. *JSME International Journal, Series B*, v. 47, n. 4, p. 672–680, 2004.

[Turner and Ferguson, 1995]TURNER, I.; FERGUSON, W. An unstructured mesh cell-centered control volume method for simulating heat and mass transfer in porous media: application to softwood drying-part ii: the anisotropic model. *Appl. Math. Modeling*, v. 19, p. 668–674, 1995.

[Velho, Figueiredo and Gomes, 1999]VELHO, L.; FIGUEIREDO, L. H. de; GOMES, J. Hierarchical generalized triangle strips. *The Visual Computer*, v. 15, p. 21–35, 1999.

[Versteeg and Malalasekera, 1995]VERSTEEG, H.; MALALASEKERA, W. *An introduction to computational fluid dynamics: the Finite Volume Method*: Longman Scientific and Technical, 1995.

[Yousuf, 2005]YOUSUF, M. *2D Compressible viscous-flow solver on unstructured meshes with linear and quadratic reconstruction of convective fluxes*: Dissertation, Indian Institute of Technology Guwahati, 2005.

[Zachmanoglou and Thoe, 1986]ZACHMANOGLOU, E.; THOE, D. W. *Introduction to partial differential equations with applications*: Courier Dover Publications, 1986.

Livros Grátis

(http://www.livrosgratis.com.br)

Milhares de Livros para Download:

Baixar livros de Literatura
Baixar livros de Literatura de Cordel
Baixar livros de Literatura Infantil
Baixar livros de Matemática
Baixar livros de Medicina
Baixar livros de Medicina Veterinária
Baixar livros de Meio Ambiente
Baixar livros de Meteorologia
Baixar Monografias e TCC
Baixar livros Multidisciplinar
Baixar livros de Música
Baixar livros de Psicologia
Baixar livros de Química
Baixar livros de Saúde Coletiva
Baixar livros de Serviço Social
Baixar livros de Sociologia
Baixar livros de Teologia
Baixar livros de Trabalho
Baixar livros de Turismo