7–1 (Invited)

Grid Generation for Three-Dimensional Process and Device Simulation

P. Fleischmann, R. Sabelka, A. Stach, R. Strasser and S. Selberherr

Institute for Microelectronics, TU Vienna, Gusshausstrasse 27–29, A-1040 Vienna, Austria
Phone +43/1/58801-3859, FAX +43/1/5059224, WWW http://www.iue.tuwien.ac.at

1. Introduction

The growing importance of three-dimensional simulation has made mesh generation the key to accurate and fast solutions. Where in two dimensions many different and only moderately sophisticated methods are established and feasible, there is the need in three dimensions for far more efficient strategies. Not only the amount of data and the complexity of the simulated structures pose an increasing challenge, but also the visualization of the three-dimensional grid as an important feedback for the developer becomes more difficult. Over the past meshing has been geared more and more towards automation [3],[14],[22],[27]. Especially, in Technology Computer-Aided Design (*TCAD*) where the input to the gridder can be the output from a topography simulator the degree of automation requires further discussion. We will investigate the state of the art and give an overview of activities in that field. In Srinivasan et al. [14] a classification of the various methods can be found. Shephard and Schroeder [15] give an overview of mesh generation from a visualization point of view. Bern et al. [13] provide an excellent, detailed and comprehensive text with many references. More information about current research all over the world is available in [20].

2. Surface Modeling

Depending on how the surface of a simulation domain is given various preprocessing steps are required. In fact, the surface itself has to become a two-and-a-half-dimensional surface grid which involves techniques like refining and smoothing. In some fields the surface grid is actually sufficient for a three-dimensional simulation and no volume decomposition is needed. Surface modeling includes the conversion of the various inputs as well as gridding related topics like applying boundary conditions. Whereas in computational fluid dynamics and related fields the surface triangulation of Non-Uniform Rational B-Spline (*NURBS*) entities gains more importance [2],[18], it is the smoothing of the input in semiconductor process and device simulation which is of interest. The output of a topography simulator may contain too much data. A cellular solid modeler approximates slopes with a staircase-like surface. In such cases the subsequent mesh generator would be unnecessarily pushed to its limits without a preceding smoothing or data reduction step. We know that there are various activities concerning smoothing in the semiconductor community, however, we are not aware of any specific references. Aftosmis, Melton, and Berger [2] deal with surface triangulations and their intersection with cells which is important for octree or cartesian based meshing concepts. A datastructure called the Alternating Digital Tree (*ADT*, [16]) which is in general very efficient for geometrical searches is used to create the cell conform surface triangulation. Concerning improved quality triangulations using Steiner points [13] work can be found for two dimensions by Chew [11]. The key activities can be summarized as follows:

Generation of a surface triangulation from a cell-based structure. From any given domain represented by evenly sized cells (cubes) the corresponding surface triangulation is generated using the Marching Cubes algorithm. We can imagine many other gridding purposes for this module in the future. At the moment it is used for cell-based topography simulations. A detailed description of our implementation can be found in [5]. If the cell structure itself is derived from a polygonal boundary representation, it will be desireable to preserve as much of the polygonal information as possible in areas where the topography remains unchanged [4]. Fig. 1 shows the surface triangulation of the cell structure as constructed with the Marching Cubes algorithm.

Smoothing and data reduction. The vertices and their neighborhood relations of a given surface triangulation are sequentially examined. If a vertex does not hold vital geometric information, it will be discarded together with incident triangles. The remaining hole is retriangulated respecting various parameters like, e.g., the aspect ratio of a triangle. A staircase-like surface with very small feature sizes compared to the overall geometric information would, for instance, be converted to an evenly smoothed surface containing much fewer vertices. For a full documentation of our implementation refer to [5]. In Fig. 2 the smoothed and greatly reduced geometric data of the trench are depicted.

Special surface triangulations. Some research also focuses on two-and-a-half-dimensional Delaunay surface and Steiner surface triangulations for practical applications in gridding. The first is important for the generation of Delaunay meshes. If a geometry conform Delaunay mesh is desired, generally, such a step will become necessary in one or the other way. In contrast to Conforming Delaunay tetrahedrizations, Constrained Delaunay tetrahedrizations allow for non-Delaunay edges and faces on the boundary [13]. The second type of surface triangulations is actually a refinement technique using Steiner points to improve the quality of the triangles. We generalized our algorithm for two-and-a-half dimensions by using projected Steiner points.

3. Volume Decomposition

The relation between the ratio of the biggest to the smallest element size and the overall number of elements is one of the most crucial aspects in three-dimensional mesh generation. Often, the grid is not optimally fitted to the desired grid density, and too coarse or too fine elements exist in various areas of the simulation domain. The result is either a grid suitable for subsequent simulations, but with less accurate solutions in some areas of interest, or a grid providing more than the minimum grid density in all areas, but pushing the limits of the subsequent simulators beyond the scope of an average workstation. An increased flexibility in refinement allowing for rapid changes of the grid density while keeping the overall element count low would be ideal. This flexibility

cannot be increased infinitely: The rapid change of element size is limited under the premise of maintaining a certain element quality.
Aftosmis, Melton, and Berger [2] have accomplished excellent work on cartesian grids in computational fluid dynamics. They manage to incorporate highly complex boundaries with quite a show of force. However, a limited flexibility in refinement is experienced, due to the cartesian nature of the grid and the isotropic refinement technique. Each further refinement level yields about 2–4 times more elements. With intial 1.6 million elements a further increase in accuracy by adding one more refinement level would result in 3.3–6.6 million elements according to one of the examples in [2]. Still, in some areas the achieved accuracy is not optimal and refinement would be motivated. Such a high point propagation after refinement and the brute force used to incorporate the boundaries into the cartesian grid can hardly be afforded on today's workstations.
Another cartesian meshing concept well known in the semiconductor community is described in [7]. This intersection-based method originates from previous works using bisection-based and octree decomposition techniques. Improvements include the ability to tesselate the cartesian cells into Delaunay tetrahedra and a more anisotropic refinement approach. However, it seems that without rigorously generating a cell conform surface triangulation as in [2] difficulties are experienced dealing with the boundaries and interfaces of a semiconductor device. We think that this difficulty is inherent to cartesian based meshing concepts. To attain optimality one needs anisotropy in all (not just three) directions of the domain which leads to unstructured grids. Johnson et al. [1] use a fully unstructured method for three-dimensional flow problems. The mesh generator is very sophisticated, however, it is closely linked to an input modeler raising the question as to how far such a concept can be utilized for *TCAD* applications. Such a link between an input editor and the gridder is related to the degree of automation which is discussed in the last section. Advancing front gridding techniques are mostly used to generate quadrilateral meshes [26],[28]. The boundary is optimally fitted as opposed to a mere boundary conformity.
A greater challenge compared to geometric models in fluid dynamics lies in the fact that semiconductor devices often exhibit a drastic variation of feature size. An extremely thin layer on top of a comparatively big silicon block calls for very sophisticated algorithms to handle the premises to generate an unstructured Delaunay grid conform with the boundaries and interfaces without any human interaction. This is one of the reasons why most of the software for Delaunay triangulations available on the Internet and other commercial and public domain mesh generators are less suitable for semiconductor simulation purposes. It is worthwhile to mention a method using local transformations [9] and Delaunay triangulations constructed by computing the convex hull in higher dimensions [21]. The use of Delaunay grids has two reasons. First, an optimality criterion [13] is fulfilled. Second, a Delaunay tetrahedrization intrinsically provides much of the information needed for a Finite Volume Integration method because of its duality to the Voronoi diagram. Okabe et al. [25] analyze Voronoi diagrams and all related topics in detail.

Our activity in three-dimensional mesh generation involves on one hand previous works which exploit various information of the input domain, and on the other hand our new fully unstructured method resulting in a Delaunay mesh.

Layer-based method. The layered structure of many devices is utilized to greatly simplify the meshing process. The projection of all layers into a single two-dimensional surface incorporates all necessary lateral grid densities and is accordingly triangulated. This surface is then duplicated and "pulled" into the third dimension for every layer of the device. The following tesselation of the prisms results in a tetrahedral mesh. It is allowed to apply transformations to these surfaces which enables the fitting of non-planar layers of the device. Local refinement in two dimensions is possible, but it will be propagated throughout the third dimension. This technique has been used successfully by *SCAP* (Smart Capacitance Analysis Program) for *Interconnect* applications. See Fig. 4 and Fig. 5 and refer to [6] for more details.

Interpolation-based method. Transfinite interpolation is used to generate a mesh consisting of structured blocks. These "hyperelements" are essentially hexaedral elements that have been regularly meshed and transformed to fit the boundary. The regular structure obtained improves the accuracy of the following solver. (For numerical grid generation refer to [17].) A complete description using the transfinite interpolation is available in [6]. Fig. 6-a shows the structured mesh of a *LOCOS*.

Fully unstructured method. A modified advancing front algorithm seems highly promising. It runs in $O(n \log n)$ time where n is the number of grid nodes. We implemented a version of such an algorithm which is linked to a visualization tool to allow online viewing of the advancing front during the gridding process. This enables exploring certain behaviors and recognizing problems immediately which is important during the development cycle. The above examples meshed with the fully unstructured method are presented in Fig. 6-b and Fig. 7. The variation of the element size in Fig. 7 compared to the mesh in Fig. 5 shows the increased flexibility in refinement. We are convinced that this approach bears the best means to cope with the previously mentioned challenges.

Currently, research should be focused on the ability to rigorously manage complex geometries, and the further improvement of the automation and refinement aspects. A parallelization of the core tetrahedrization algorithm might be a concern. This has been investigated for a similar algorithm in [19].

4. Grid Adaptation

Local refinement which respects various criteria like second-order statistics is necessary to achieve accurate solutions in a reasonable amount of time. Such a refinement step can be performed statically as preprocessing to a simulation, as well as dynamically during a simulation run. Local grid adaptation might be desired after each timestep of transient simulations. The areas of refinement have to "follow" the migrating areas of

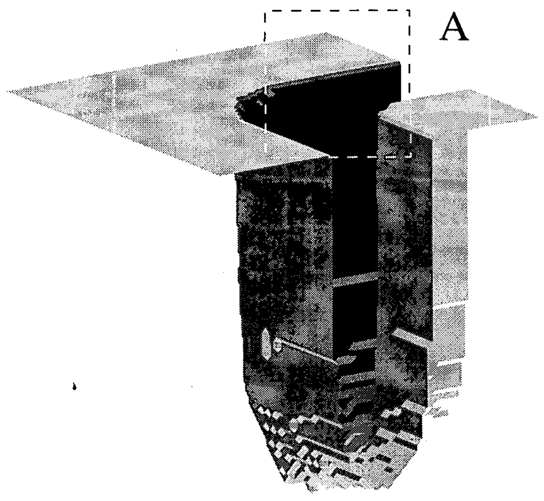

Figure 1: Surface triangulation of a trench automatically generated from a cell-based structure. The surface consists of 22896 triangles. (The triangles are not shown, because of their extremely small size. See detailed view *A* in Fig. 3)

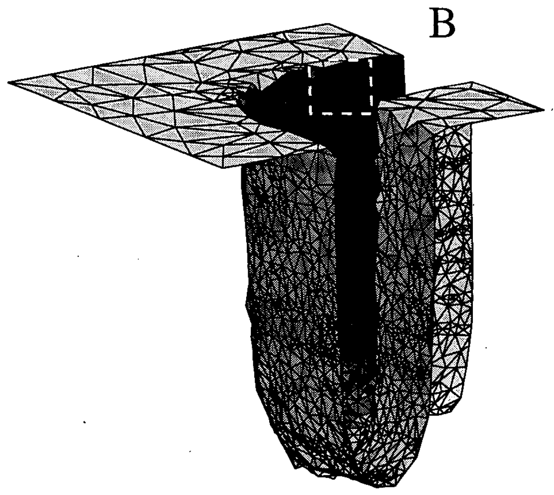

Figure 2: Smoothed and reduced surface triangulation of the trench containing 3759 triangles. Detailed view *B* is shown in Fig. 3

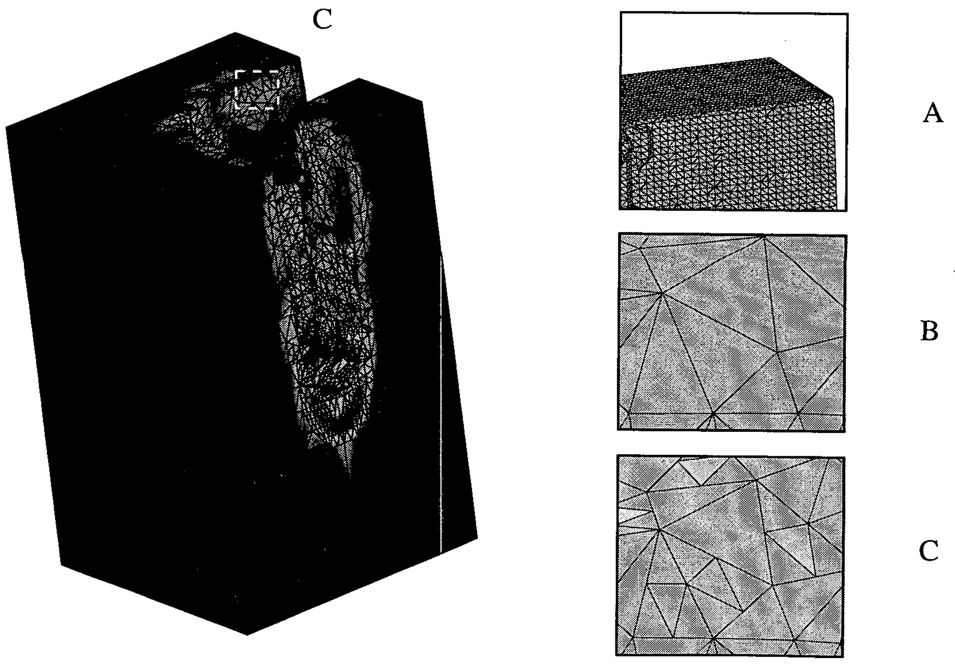

Figure 3: Unstructured Delaunay mesh of the trench. Shading corresponds to the doping. Details *B* and *C* show the excact same area to illustrate grid adaptation using the Mixed-Element decompostion method. The initial grid consisted of 25537 tetrahedra.

interest. Essentially, local refining as well as coarsening becomes necessary to avoid meshing of the entire domain repeatedly. A possible solution is to store the different levels of refinement at the same time. These hierarchical grids are typically employed for multigrid methods [8]. However, such grids also have limitations because of the following two disadvantages. If the initial grid with areas of refinement is already close to the limit of computer memory, it will not be feasible to store several grid hierarchies. If the domain is composed of moving boundaries, all grid hierarchies will have to be updated. Thus, it can be advantageous to store only one grid and use very fast regridding techniques for the purposes of adjusting to a moved boundary as well as coarsening and refining where necessary. This poses a major challenge, and we do not know of anyone in the semiconductor community who achieves this at ease in practical three-dimensional applications. In computational fluid dynamics a very intriguing solution for the case of several spheres falling into a liquid has been accomplished in parallel on a Thinking Machine CM-5 by [1]. Generally, tetrahedral elements are preferred to hexaedral elements, because the latter loose their coplanarity under grid deformations.
Certainly, the issue of accurately discretizing the desired quantities merits further investigation. The mesh refinement algorithm does not know a priori the required element sizes to do this correctly. It can only judge upon the existing grid whether or not the grid density needs to be locally increased. It has no knowledge of how pronounced this increase should be. This leaves room for optimization of such iterative refinement steps to reduce the number of their occurance.
Another way of adapting a grid is the relaxation of the grid nodes. A straight-forward method applicable for surface grids as well as volume grids can be described as follows. A grid node is moved to the geometric center of the surrounding grid nodes that are directly connected via single edges. Note the implication of an isotropic grid node distribution. Concerning bisection-based refinement a new method has been introduced by Liu and Joe [10].
Our work on grid adaptation in three dimensions consists of:

Hierarchical grid refinement. This refinement technique uses mixed elements (octahedra and tetrahedra) and can be applied to any unstructured tetrahedral meshes. The element quality is preserved throughout further refinement levels. All levels are stored. Based on a discretization error criterion the grid used for the simulation can be coarsened or refined where necessary. This method is currently implemented in our three-dimensional diffusion simulator, see enlarged details in Fig. 3. The algorithm is explained in [12].

Local regridding. The previously described fully unstructured volume decomposition algorithm is intrinsically ideal for regridding purposes. In fact, the mentioned parallelization capabilities and the local (re-)gridding capabilities are different appearances of the same property. The reason for this lies in the modified advancing front method. Grid nodes can be erased as well as inserted, and deformed elements can be discarded. The emerging holes in the grid correspond to local surrounding fronts which are directly fed back into the algorithm. The consistency is restored by a following regridding step. Another application of regridding are quality tetrahedrizations using the insertion of Steiner points.

5. General Aspects and Conclusion

In the early beginnings of mesh generation developers had to create their meshes manually exploiting their knowledge and understanding of the geometry. Since then, more general algorithms have been introduced and today almost every mesh generator claims to work automatically and to "understand" the geometry without human interaction. We feel that human interaction essentially means additional geometric information, and therefore we find the degree of automation highly related to the amount of geometric information the mesh generator is provided with through an input interface. Consider the following: With impressive looking grids of, e.g., an engine consisting of complicated mechanical parts, cylinders, and valves, one has to wonder how it is possible for the meshing algorithm to understand the geometry that well. The answer is that the meshing algorithm is provided with the information of each cylinder and part itself. It merely has to generate an, e.g., structured grid for a general sized and transformable cylinder and other basic objects. If a single, combined surface of all parts would be the only input information, the meshing would become unproportionally more complicated. For practical engineering purposes we can therefore distinguish two concepts which both have their own applications: (i) A mesh generator utilizing as much information of the domain as possible to simplify and to speed up the meshing process. (ii) A mesh generator capable of meshing the same domain with the minimum information necessary. As for the first concept we refer to the layer-based gridding method. We can also imagine a tight link between a geometry editor and a less general mesh generator to compose an intial grid more quickly. The second concept has gained importance in semiconductor process and device simulation. In a more extensive and complete design cycle like in *TCAD* where the form of the geometry is itself subject to simulation and generated through, e.g., a topography simulator the second concept becomes a critical factor. In order for the mesh generator to deal with automatically generated topographies the most general and sophisticated algorithms have to be applied. It might even be necessary to repair the geometric input data as it oftens contains inconsistencies and other artifacts. The orientation of the polygonal faces might not be well defined or gaps in the boundary might exist. Computer generated topographies not only evolve from etching and depositon modules, but also through three-dimensional extraction of geometric data from two-dimensional simulations and through solid modelers deriving their information from layout data. Fig. 1, Fig. 2, and Fig. 3 show several geometric steps of a successful link between the etching, implantation, and diffusion modules including grid adaptation. Still, a lot of work has to be accomplished before a degree of automation for the mesh generator is reached which permits its linkage to a simulation flow control unit to carry out all gridding tasks between the many coupled simulators within an integrated system for Technology CAD applications. The reader might also be interested in [23] concerning the automation, and in [24] for future aspects of *TCAD*.

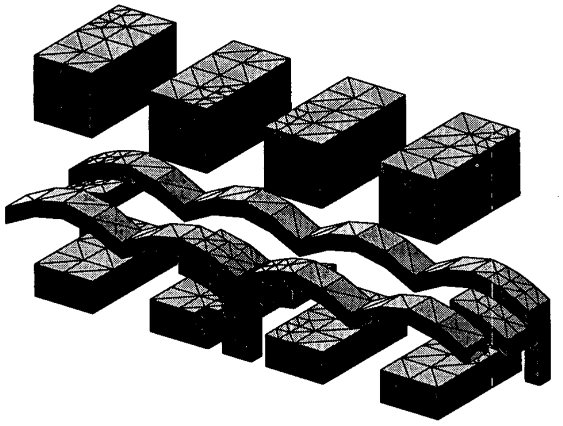
Figure 4: Structure of the discretized conductors in a DRAM ($0.8\mu m$ x $3.2\mu m$).

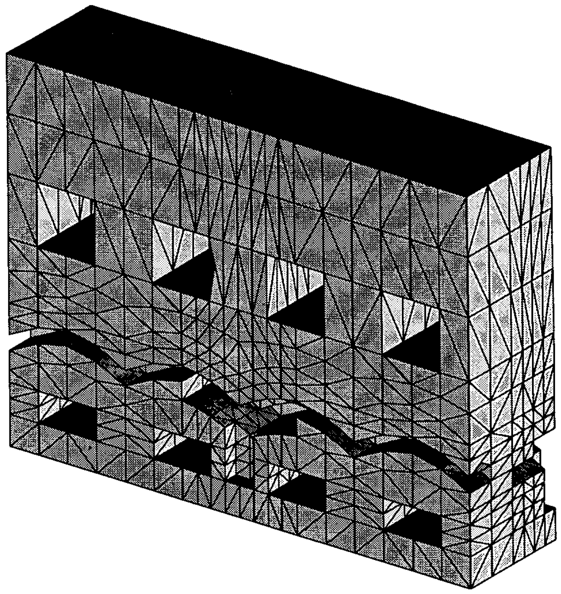
Figure 5: Mesh generated with the layer-based method for Interconnect simulation, 6480 tetrahedra. Note the vertical propagation of refinement through all layers.

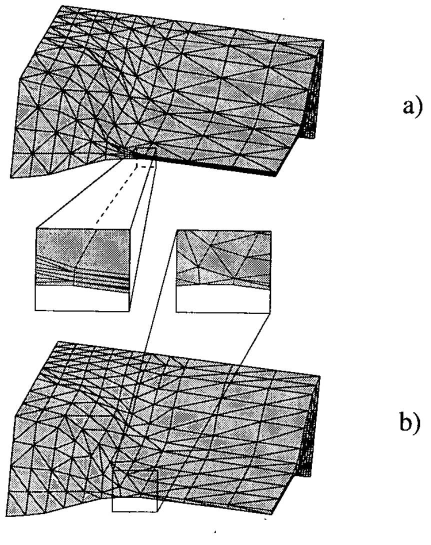

Figure 6: *LOCOS*: a) Mesh generated with transfinite interpolation, 2000 tetrahedra. b) Unstructured Delaunay Mesh consisting of 957 tetrahedra.

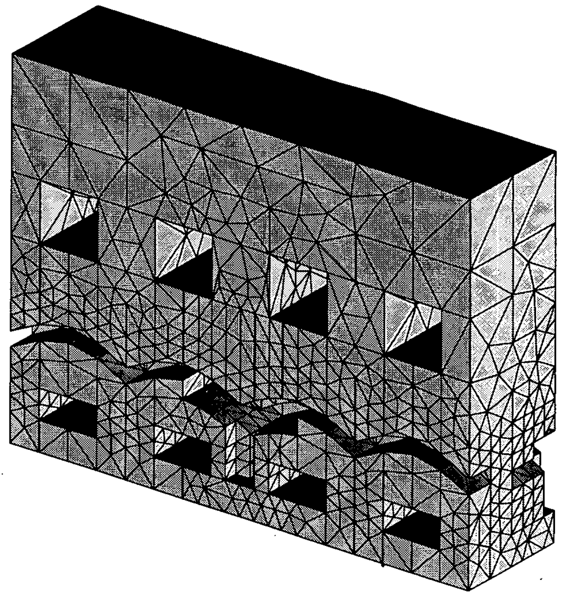
Figure 7: Fully unstructured Delaunay mesh of the DRAM for Interconnect simulation, 5290 tetrahedra.

[1] A.A. Johnson and T.E. Tezduyar, "Mesh Generation and Update Strategies for Parallel Computation of 3D Flow Problems", In S.N. Atluri, G. Yagawa, and T.A. Cruse, editors, *Proceedings of the International Conference on Computational Engineering Science*, 1995.

[2] M.J. Aftosmis, J.E. Melton, and M.J. Berger, "Adaptation and Surface Modeling for Cartesian Mesh Methods", *AIAA* Paper 95-1725-CP, 1995.

[3] J.E. Melton, F.Y. Enomoto, and M.J. Berger, "3D Automatic Cartesian Grid Generation for Euler Flows", *AIAA* Paper 93-3386-CP, 1993.

[4] R. Mlekus and S. Selberherr, "Polygonal Geometry Reconstruction after Cellular Etching or Deposition Simulation", In H. Ryssel and P. Pichler, editors, *Simulation of Semiconductor Devices and Processes*, pp. 50–53, 1995.

[5] C. Ledl, "Konvertierung rasterorientierter Geometrien in polygonal begrenzte", Masters thesis, Institute for Microelectronics, Technical University Vienna, 1994.

[6] R. Bauer, "Numerische Berechnung von Kapazitäten in dreidimensionalen Verdrahtungsstrukturen", Ph.D. thesis, Institute for Microelectronics, Technical University Vienna, 1994.

[7] G. Garretón, L. Villablanca, N. Strecker, and W. Fichtner, "Unified Grid Generation and Adaptation for Device Simulation", In H. Ryssel and P. Pichler, editors, *Simulation of Semiconductor Devices and Processes*, pp. 468–471, 1995.

[8] R. Strasser, "Multigrid Methods in 2D-Process Simulation", Masters thesis, Institute for Microelectronics, Technical University Vienna, 1995.

[9] B. Joe, "Construction of Three-dimensional Improved-Quality Triangulations Using Local Transformations", *SIAM: J. Sci. Comput.*, 16(6), pp. 1292–1307, 1995.

[10] A. Liu and B. Joe, "Quality Local Refinement of Tetrahedral Meshes Based on Bisecton", *SIAM: J. Sci. Comput.*, 16(6), pp. 1269–1291, 1995.

[11] L.P. Chew, "Guaranteed-quality triangular meshes", Tech. Rep. TR-89-983, Cornell University, 1989.

[12] E. Leitner and S. Selberherr, "Three-dimensional Grid Adaptation Using a Mixed-Element Decomposition Method", In H. Ryssel and P. Pichler, editors, *Simulation of Semiconductor Devices and Processes*, pp. 464–467, 1995.

[13] M. Bern and D. Eppstein, "Mesh Generation and Optimal Triangulation", In F.K. Hwang and D.-Z. Du, editors, *Computing in Euclidean Geometry*, pp. 201–204, 1992.

[14] V. Srinivasan, L. Nackman, J. Tang, and S. Meshkat, "Automatic Mesh Generation Using the Symmetric Axis Transformation of Polygonal Domains", *Proc.IEEE*, 80(9), pp. 1485–1501, 1992.

[15] M.S. Shephard and W.J. Schroeder, "Analysis Data for Visualization", In R.S. Gallagher, editor, *Computer Visualization*, pp. 61–87, 1995.

[16] J. Bonet and J. Peraire, "An Alternating Digital Tree Algorithm for Geometric Searching and Intersection Problems", *Int. J. Numer. Meth. Engng.*, 1990.

[17] P. Knupp and S. Steinberg, "Fundamentals of Grid Generation", CRC press, 1993.

[18] J.W. Peterson, "Tessellation of NURB Surfaces", In P.S. Heckbert, editor, *Graphics Gems IV*, pp. 286–320, 1994.

[19] P. Cignoni, D. Laforenza, C. Montani, R. Perego, and R. Scopigno, "Evaluation of Parallelization Strategies for an Incremental Delaunay Triangulator in E^3", WWW `http://www.di.unipi.it/~cignoni/papers.html`, 1994.

[20] R. Schneiders, "Information on Finite Element Mesh Generation", WWW `http://www-users.informatik.rwth-aachen.de/~roberts/meshgeneration.html`, 1996.

[21] B. Barber, "QHULL", WWW `http://www.geom.umn.edu/~bradb/qhull-news.html`, 1996.

[22] J.C. Cavendish, D.A. Field, and W.H. Frey, "An Approach to Automatic Three-dimensional Finite Element Mesh Generation", *Int. J. Numer. Meth. Engng.*, 21, pp. 329–347, 1985.

[23] B.A. Cipra, "A Rapid-Deployment Force for CFD: Cartesian Grids", *SIAM NEWS*, 28(10), 1995.

[24] M.E. Law, "The Virtual IC Factory ... can it be achieved ?", *IEEE* Paper 8755-3996, 1995.

[25] A. Okabe, B. Boots, and K. Sugihara, "Spatial Tesselations — Concepts and Applications of Voronoi Diagrams", Wiley, 1992.

[26] T.D. Blacker and M.B. Stephenson, "Paving: A New Approach to Automated Quadrilateral Mesh Generation", *Int. J. Numer. Meth. Engng.*, 32(4), pp. 811–847, 1991.

[27] M.A. Yerry and M.S. Shephard, "Automatic Three-dimensional Mesh Generation by the Modified-Octree Technique", *Int. J. Numer. Meth. Engng.*, 20, pp. 1965–1990, 1984.

[28] R. Löhner and P. Parilch, "Three-dimensional Grid Generation by the Advancing Front Method", *Int. J. Numer. Meths. Fluids.*, 8, pp. 1135–1149, 1988.