# 3D TCAD at TU Vienna

E. Leitner, W. Bohmayr, P. Fleischmann, E. Strasser, and S. Selberherr

Institute for Microelectronics, TU Vienna
Gusshausstrasse 27–29, A-1040 Vienna, Austria

Abstract

This paper gives an overview about our research on three-dimensional process simulation. Today's activities are worldwide still suffering from a lack of appropriate geometric modeling, robust gridding, accurate and verifiable physical models as well as computationally efficient numerical algorithms. Possible solutions to some of these problems are demonstrated on the basis of our three-dimensional process simulation tools.

## 1. Introduction

The development of today's semiconductor devices often requires to investigate three-dimensional problems, where in many cases numerical simulation delivers useful information. For the reliability of such hints the accuracy of the simulation results is crucial. The effects occurring at intrinsic three-dimensional topologies, like corners etc., are gaining importance with shrinking device dimensions. Thus, consideration of the third dimension within the simulation is a must for both process and device simulation.

In contrast to three-dimensional device simulators which are available from universities as well as from commercial sources, the situation in process simulation is quite different: Today it is not possible to perform a complete three-dimensional simulation of a whole process. Impressive work on topography simulation resulted in excellent programs for surface evolution during etching and deposition processes [1] [2]. An engineering workstation is sufficient for this kind of simulations. Also programs for ion implantation based on Monte Carlo methods are available [3] [4] [5]. Where the first versions consumed CPU-times beyond one week (on an HP9000-735), recent developments allow to compute realistic three-dimensional results over night in the amorphous mode, respectively one day for the crystalline mode. Despite these encouraging results, simulation of a whole process fails on missing diffusion and oxidation simulators. Although some three-dimensional simulators have already been presented [6] [7], the complex structures of realistic devices cannot be handled by them because of the lacking grid flexibility.

Therefore development of fully flexible three-dimensional diffusion and oxidation simulators is highly recommended. However, the problems involved are quite complex. Modern devices have more or less arbitrary geometries which are difficult to handle, and the simulation of thermal activated processes requires the solution of coupled, nonlinear partial differential equation systems. This just can be achieved efficiently

by use of adaptive gridding techniques as well as highly efficient algebraic methods. Additional challenges lie in the simulation of thermal oxidation, where the diffusion equations and the mechanical equation have to be taken into account. The resulting changes of the geometry reveal high demands to the gridding unit, and additionally the stiff mechanical equations expense the solution of the linear systems. This extremely high complexity within one application has scared of many researchers from tackling the problem.

Besides the numerical and geometrical problems, the quantization of the parameters required by the differential equations accounts for a great deal of controversy. Even for one-dimensional diffusion processes, where the profiles can be measured with satisfying accuracy, the range of the proposed diffusivities is quite large. The situation is even worse for the properties of the point defects: Since it is impossible to measure point defect distributions directly, these are usually quantified due to their effects on the dopant distribution. E.g., the initial distribution is chosen, in order to reach the desired influence on dopant diffusion using a certain set of coupling coefficients. These coupling coefficients have to be determined by calibration of the coefficients, which in turn is influenced by the initial condition. Thus, the missing orthogonality allows to produce almost any result.

Furthermore, it has been pointed out [8], that the lack of multi-dimensional measurement techniques inhibits the calibration of corresponding multi-dimensional simulation tools. On the other hand process simulation seems to be the only possibility to obtain multi-dimensional profiles, because of the insufficient measurement techniques. One may speculate that once we will have reliable one-dimensional physical models, their usage within multi-dimensional simulators will allow to predict doping profiles much more accurately than any measurement technique.

Finally, achieving a complete simulation of a three-dimensional process-flow demands highest flexibility in data management. Data exchanging between different simulators using different data representation forms for geometries (e.g. cellular based, octree based or polygonal based) and profiles stored at different grid types account for a large additional effort which is necessary to couple different simulators effectively. As an example, the generation of a tetrahedral grid for a device geometry computed by a topography simulator such as [1] needs first to convert the cellular based geometry to a polygonal based one. Then the polygonal surface has to be adapted in order to fulfill some conformity conditions and grid points within the solids have to be computed before the tetrahedrization algorithm can be applied. Furthermore, optimization of the grid in terms of element quality and minimum node count requires some kind of optimization loops. All those steps suffer on the enormous amount of data and the structural complexity to be dealt with.

Our work regarding the three-dimensional process simulation resulted in several process simulation modules: In Section 2 we present our simulator for surface evolution and the coupling with physical models for etching and deposition processes. Section 3 deals with the ion implantation module and recent improvements there. In Section 4 we present our module for diffusion processes, and finally, Section 5 contains an outlook on our attempts to tackle three-dimensional oxidation simulation.

## 2. Topography Simulation

Over time a variety of surface evolution algorithms has been studied to build three-dimensional topography simulators. Among them many algorithms have been reported for resist development in lithography simulation [9] [10] [11] [12], only a few methods have been proposed for the simulation of etching and deposition processes [2] [13] [14] [15]. Basically there are two types of algorithms used for three-dimensional topography simulation. Volume-removal methods divide the material being etched into a large array of rectangular prismatic cells. Each cell is characterized as etched, unetched or partially etched. During etching cells are removed one-by-one according to the local etch rate and the number of cell faces exposed to the etching medium. These algorithms have been successfully used in three-dimensional lithography simulation [9] [10]. Volume-removal methods can easily handle arbitrary geometries, but unfortunately they suffer from inherent inaccuracy, because they favor certain etch directions as was found by many researchers [9] [16]. The second type of surface evolution algorithms represents the surface of the material being etched by using a mesh of points which are connected by line segments to form triangular facets [2] [16]. Depending on the implementation either the mesh points or the facets are moved according to the local etch rates. A mesh management is necessary to maintain the mesh as it moves in time. In general, these algorithms deliver highly accurate results, though with potential topological instabilities such as erroneous surface loops which result from a growing or etching surface intersecting with itself. The surface loops must be located and removed to conserve memory and maintain efficiency of the simulation tool [17].

### 2.1. A General Method for Surface Advancement

Extensive work in the past has resulted in a general method for surface evolution in three-dimensional topography simulation [1] [18] [19]. This method is based on morphological operations which are performed on a cellular material representation considering the simulation geometry as black and white image (material and vacuum). The resulting surface advancement algorithm allows arbitrary changes of the actual geometry according to a precalculated etch or deposition rate distribution and can support very complex structures with tunnels or regions of material which are completely disconnected from other regions. Surface loops resulting from a growing or etching surface intersecting with itself are inherently avoided.

The material is represented using an array of square or cubic cells, where each cell is characterized as etched or unetched. Additionally, a material identifier is defined for each cell, therefore material boundaries need not be explicitly described as shown in Fig. 1.

The surface boundary consists of unetched cells that are in contact with fully etched cells. Cells on the surface are exposed to the etching medium or to the deposition source, and etching or deposition proceeds on this surface. A linked surface cell list stores dynamically array addresses and rate information of exposed material cells. To advance the surface a structuring element whose spatial dimensions are related to the local etch or deposition rate is applied for the exposed cells. Usually, for anisotropic two-dimensional simulation the structuring element is an ellipse with constant ratio of major to minor axis which is applied in the direction of the local etch or deposition rate vector as shown in Fig. 2, for isotropic movement of the surface point the applied structuring element changes into a circle.

Depending on the simulated process either material cells are removed or added which are located within the structuring element. In case of deposition the structuring

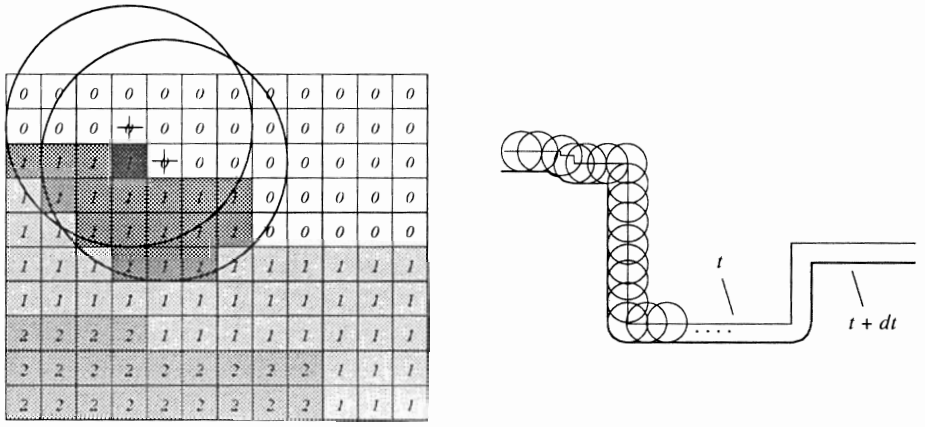


Figure 1: The material representation. The considered surface cell is dark shaded, the number in the cell denotes the material identifier.

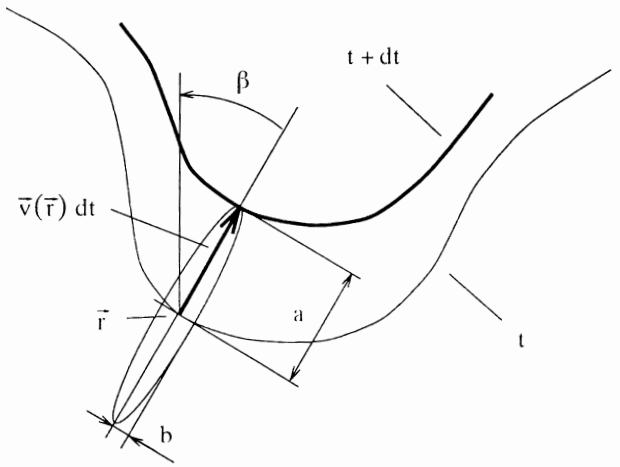


Figure 2: The structuring element for anisotropic surface advancement.

element is centered at the midpoint of the considered surface cell, whereas in case of etching the structuring element is applied at the midpoints of neighboring cells which are located adjacent to the exposed cell sides of the surface cell. For anisotropic three-dimensional surface advancement structuring elements are ellipsoids, for isotropic movement of surface points structuring elements are spheres, although there is no algorithmic restriction on the shape of the applied structuring elements. After each time step the exposed boundary has to be determined. Therefore all cells of the material array are scanned. Material cells are surface cells if at least one cell side is in contact with a vacuum cell. The exposed sides of the detected surface cells finally describe the material surface at a certain time step.

## 2.2. Modeling of Etching and Deposition Processes

Many topography processes are affected by the shape of the surface. Successful two-dimensional simulation programs for etching and deposition processes use macroscopic point advancement models that consider information about particle fluxes and surface reactions to calculate etch or deposition rate distributions along the exposed surface [20] [21] [22]. This approach is extremely desirable, since a variety of process models for etching and deposition in the literature already exists and quantities such as etch or deposition rates are easily measurable in semiconductor technology.

To determine the rate contributions of incoming particles both in etching and deposition the simulator must be capable to calculate the resulting particle flux incident at a surface point. Therefore a spherical coordinate system with polar angle $\vartheta$ and azimuth angle $\varphi$ is assumed and the region above the wafer is divided up into several surface patches ($N_\varphi \times N_\vartheta$) as shown in Fig. 3.

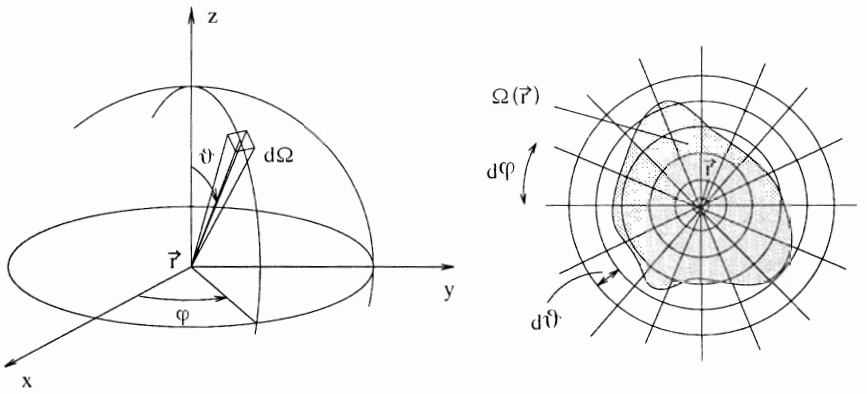


Figure 3: The calculation of the incident particle flux.

The incident flux is then integrated over those patches of the hemisphere which are visible from the surface point $\vec{r}$. To determine if a surface patch is visible from a point on the surface a shadow test has to be performed along a given direction which is within the cellular structure simply the matter of following a discretized line of cells from the surface cell to the boundary of the simulation area. If any cell on this line is a material cell, then the surface cell is shadowed. The calculation of the visible solid angle $\Omega = \Omega(\vartheta, \varphi)$ with $\mathrm{d}\Omega = \sin\varphi \, \mathrm{d}\varphi \, \mathrm{d}\vartheta$ (the radius of the hemisphere may be normalized to one) is then reduced to a series of shadow tests. The number of shadow tests required at a surface point corresponds to the number of patches of the

hemisphere. As this number is a constant (typically $90 \times 45$ ($N_\varphi \times N_\vartheta$) patches are used), the time required to calculate the visible solid angle for the entire surface is proportional to the number of surface points.

Some processes such as ion milling or crystal etching show a strong dependence on the local surface orientation. The cellular material representation does not provide this information inherently, but the calculation is rather simple. At each exposed side of a surface cell a normal vector can be defined as shown in Fig. 4.

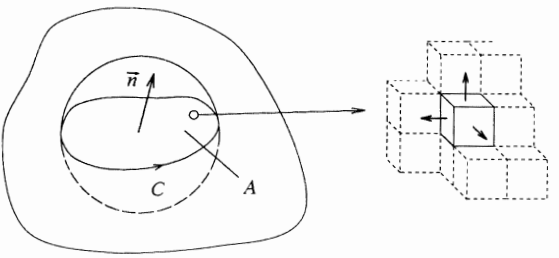


Figure 4: The calculation of the local surface orientation.

The surface normal at a given surface point is then calculated by summing up the normal vectors of surface cells within a certain vicinity to that surface point. For practical simulations surface cells that are located within a sphere are considered, a sphere radius of typically 10 to 15 cells gives highly accurate results.

*Etch Models:* As a basic concept we consider a linear combination of isotropic and anisotropic reactions of directly and indirectly incident particles to calculate the resulting velocity vector of a surface point. The isotropic reaction is mainly a chemical reaction affected by a reactive gas, in which the reactive particles have short mean free paths and move randomly. The anisotropic reaction is a physical or chemical reaction, where the particles have long mean free paths compared to the device dimensions, and angular particle fluxes must be taken into account. A general process model which accounts for various physical mechanisms like directional etching due to incident ions, etching due to reactive neutrals, and etching caused by reflected or re-emmited particles can be expressed by:

$$v_{iso}(\vec{r}) = R_{iso}\,, \tag{1}$$

$$v_{dir}(\vec{r}) = R_i \int_\Omega F_i(\Omega)\, S_i(\alpha) \cos\alpha \,\mathrm{d}\Omega + R_n \left[1 + D_i \int_\Omega F_i(\Omega)\,\mathrm{d}\Omega\right] \int_\Omega F_n(\Omega)\, \cos\alpha \,\mathrm{d}\Omega + \\ + R_r \int_{H-\Omega} \cos\alpha \,\mathrm{d}\Omega\,, \tag{2}$$

with:

$$F_i(\Omega) = \exp(-\vartheta^2 / 2\,\sigma^2) \,/\, N_i\,, \tag{3}$$

$$S_i(\alpha) = a_1 \cos\alpha + a_2 \cos^2\alpha + a_3 \cos^4\alpha\,, \tag{4}$$

$$F_n(\Omega) = \cos^m(\vartheta) \,/\, N_n\,, \tag{5}$$

$$F_r(\Omega) = 1\,, \tag{6}$$

where $v_{iso}$ and $v_{dir}$ describe the surface velocity at a surface point along the surface normal, $R_{iso}$ denotes an isotropic etch rate caused by reactive particles of a plasma whose mean free paths are short compared to characteristic device dimensions. The particles are moving randomly, therefore the etch rate has no orientation or flux dependencies. $R_i$ is the etch rate, $F_i$ is the flux distribution, and $S_i$ is the sputter yield due to directly incident ions. $R_n$ and $F_n$ are the etch rate and angular flux distribution for reactive neutrals, $D_i$ is a damage parameter which accounts for the enhancement of the chemical etch rate of neutral particles by the presence of directly incident ions that damage the surface. $R_r$ and $F_r$ describe etching due to reflected particles. $\alpha$ denotes the angle between the incident direction and the surface normal and $\Omega$ describes the visible solid angle of the considered surface point.

*Deposition Models:* Deposition modeling is based on the original work of Blech who developed a model for describing two-dimensional profiles of evaporated thin films over steps [23]. This model is directly applicable to three-dimensional simulation. In three dimensions the components of the growth vector can be calculated by:

$$v_x(\vec{r}) = R_d \int_{\Omega} F_d(\Omega)\, \cos\varphi\, \sin\vartheta \, \mathrm{d}\Omega \,. \tag{7}$$

$$v_y(\vec{r}) = R_d \int_{\Omega} F_d(\Omega)\, \sin\varphi\, \sin\vartheta \, \mathrm{d}\Omega \,, \tag{8}$$

$$v_z(\vec{r}) = R_d \int_{\Omega} F_d(\Omega)\, \cos\vartheta \, \mathrm{d}\Omega \,, \tag{9}$$

where $R_d$ denotes the deposition rate on a flat wafer without shadowing and $F_d(\Omega)$ is the angular flux distribution function of incoming particles. A general cosine-based flux distribution function may be expressed [2] as:

$$F_d(\Omega) = \cos^n(A\,\vartheta)\,/\,N \,, \quad for\ \ \vartheta \leq \pi/2A\ \ otherwise\ \ 0. \tag{10}$$

The parameter $A$ restricts the angle of incoming particles, the parameter $N$ allows over-cosine and under-cosine distributions.

## 2.3. Conversion of the Cellular Geometry Representation

The surface advancement algorithm based on the cellular material representation allows a very stable simulation of arbitrary three-dimensional device structures. Unfortunately, many other simulators can not handle this geometry representation form directly. They require a polygonal geometry representation as input, where the simulation geometry is described by a number of polygons (in most of the cases triangles).

To convert the cellular geometry representation we use the so called Marching Cube Algorithm which was proposed by Lorensen and Cline [24]. This method determines the surface in a logical cube which is created from eight adjacent cells of the material array. According the eight vertices of such a logical cube, there are exact 256 ways a surface can intersect the cube which can be further reduced due to different symmetries to 15 different patterns. Some of the possible patterns are shown in Fig. 5.

The algorithm first determines the surface for one logical cube then moves (or marches) to the next cube. Marching through the whole material array will construct the polygonal geometry representation.

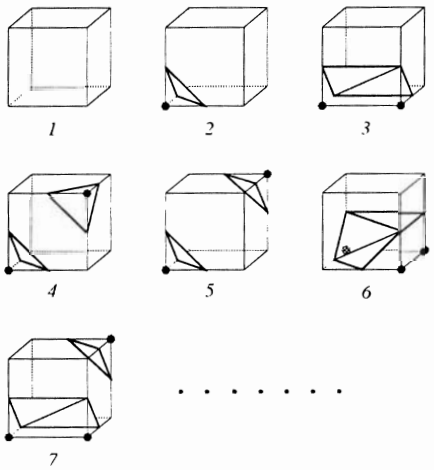


Figure 5: Some of the possible patterns of the Marching Cube Algorithm.

One disadvantage of the Marching Cube Algorithm is that it produces a very large number of triangles which must be reduced afterwards to conserve memory and computational efficiency. We apply a decimation algorithm proposed by Schroeder et al.[25]. In this algorithm multiple passes over all vertices in the mesh are made. During a pass, each vertex is a candidate for removal and, if it meets the specified decimation criteria, the vertex and all the triangles that use the vertex are removed. One such decimation criterion for a vertex is the distance to an average plane which can be calculated using the triangle normals of adjacent triangles to the vertex of interest. The resulting hole after removing the vertex and the triangles in the mesh is patched by a local triangulation. The vertex removal process repeats until some termination condition is met. Usually the termination criterion is specified as a percent reduction of the original mesh.

## 2.4. An Example

Fig. 6 shows the typical barreling phenomenon which results in ion enhanced plasma etching due to high energetic ions that increase the etch rate where they hit the surface (mainly at the bottom of the trench) and due to reactive neutrals which also attack the sidewalls. The picture also shows the well known aperture effect (etch rate decreasing due to limited delivery of ions and radicals) resulting in a deeper trench where the mask size opening is larger.

The chemical etch rate for this example was $R_n = 0.65\,nm/s$ with $D_i = 6.0$ and the etch time was $400\,s$. The parameter of the particle distribution functions were $\sigma = 2.0$ and $m = 1.0$.

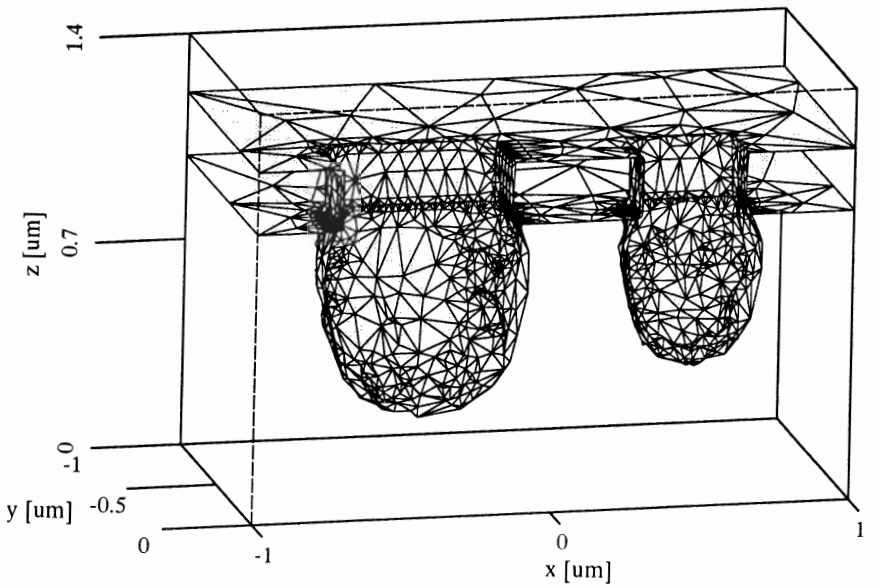


Figure 6: Ion enhanced plasma etching of trenches showing the barreling phenomenon and the aperture effect.

## 3. Monte Carlo Simulation of Ion Implantation

Introducing controllable amounts of dopant impurities into substitutional sites of a semiconductor crystal predictably modifies its electrical properties. During the past fifteen years, ion implantation has progressed steadily from its initial use as a alternative to diffusion to its present dominating role in the manufacture of VLSI and ULSI circuits. The reason is that we can achieve better control and reproducibility of concentration and depth. Further important features are its flexibility and ability to form almost arbitrary doping profiles, e.g. buried layers, and the favorable factor that ion implantation is a low-temperature process. Since modern annealing methods such as rapid thermal annealing (*RTA*) do not alter the implanted profile very much, the initial profile mainly controls the final result and thus, its determination has become an important task.

The Monte Carlo simulation of ion implantation [4] [5] [26] [27] [28] [29] [30] is rapidly gaining acceptance due to its capability of simulating channeling and damage accumulation phenomena in arbitrary multi-dimensional structures. A well-known disadvantage of the Monte Carlo approach is its considerable demand for computer resources to obtain results with satisfying statistical accuracy.

### 3.1. Point-Location and Material Detection: The Octree

The Monte Carlo method is based on tracing a large number of trajectories of individual ions on their way through the target until they find their final position. Therefore

one crucial aspect of this approach is to determine the spatial location of the ion within the three-dimensional simulation area (*point-location* and *material detection*). To keep the computational effort within reasonable limits we use an octree for discretization of the geometry. This scheme provides a fast solution of the point-location problem by mapping the structure into a hierarchical tree representation [31] [32].

The octree method originates from graphical image processing [33] [34] [35], although in this connection the contrary task, namely to combine areas with the same properties, is desired. Nevertheless this method can be suitably adopted for ion implantation where one big area must be subdivided into smaller zones. In any case the zones shall be as large as possible to achieve the fastest solution.

To meet this requirement the whole geometry is included in one cube (*root cube*). This cube is then subdivided into eight subcubes, if it is not composed entirely of the same material. This procedure is recursively continued for every subcube until either the desired accuracy of the discretization is reached — which is measured by the length of the edges of the cube — or no more intersections of this cube with the polygons defining the target geometry exist (*leaf cube*). At the end each leaf cube is related to exactly one material (Fig. 7).

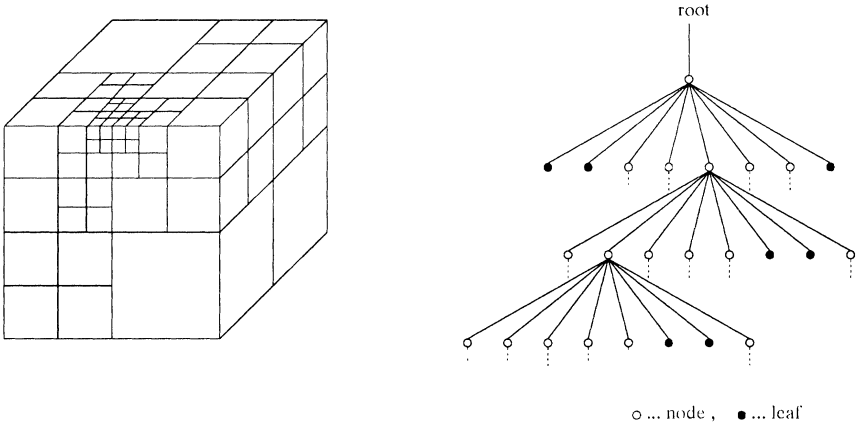


Figure 7: Discretization of a geometry using an octree.

To determine the location of an ion, just a simple test of the coordinate against the related coordinates of the sidewalls of the cube is required, because the cubes of the octree are all aligned with the coordinate axes.

### 3.2. Amorphous Computation Mode: The Superposition Method

For the Monte-Carlo simulation we let the ions start at equidistant lateral positions (no correlations between trajectories). The number of particles to be simulated is a major concern as the simulation time will be proportional to this quantity. A major part of CPU-time is used for the evaluation of the ion-target interaction. The fundamental idea to reduce the simulation time is to use each ion trajectory several times to determine the history of ions entering the target (Fig. 8). This superposition law holds if the history of all ions is independent as it is the case for amorphous targets [30].

The following algorithm is justified by the superposition law:

1. subdivide the width of the simulation area into $N_W$ subwindows (we use the lateral standard deviation to determine the width of such a subwindow)
2. calculate $N$ *physical* model trajectories in an infinite target for each ion-material combination
3. make copies of this trajectory and move them to corresponding points of each subwindow
4. follow each trajectory copy and check if any boundary is crossed. In this case the used model trajectory is changed according to the new material.

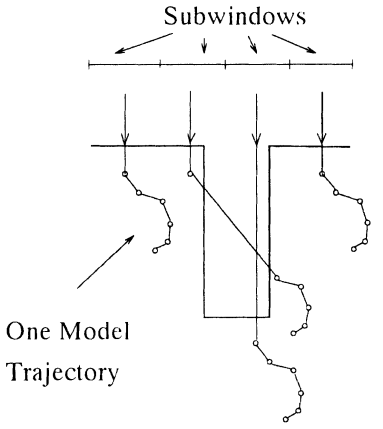


Figure 8: Construction of trajectories from one "physical" ion trajectory.

The simulation will be approximately equivalent to a conventional simulation with $N \cdot N_W$ particles.

However, the computation time required for the necessary geometry checks for the point-location remains uneffected (see Section 3.1).

### 3.3. Crystalline Computation Mode: The Trajectory Split Method

In order to maintain the performance of ULSI circuits, it is important to form very shallow Source-Drain junctions and to reduce the thermal budgets. Thus, improved models for ion implantation are needed [36] [37] and the traditional assumption of random targets (see Section 3.2) is not longer applicable [38].

The traditional Monte Carlo approach for crystalline targets is based on the calculation of a large number of "distinct" ion trajectories, i.e. each trajectory is usually followed from the ion starting point at the surface of the target up to the stopping point of the ion. Since the majority of ion trajectories ends at the most probable penetration depth inside the structure, the statistical representation of this target region is good. Regions with a dopant concentration several orders of magnitudes smaller than the maximum (in the following we call these areas "peripheral") are normally represented by a much smaller number of ions (typically $10^4$ times lower than at the maximum). This results in an insufficient number of events at low concentration areas and leads to statistical noise that cannot be tolerated.

For that reason and inspired by S.-H. Yang [39] we developed the *trajectory split method* [3] [40] for the Monte Carlo simulation of ion implantation on the basis of [41]. Our algorithm drastically reduces the computational effort and is applicable for two and three-dimensional simulations.

The fundamental ideas of our new simulation approach are to locally increase the number of calculated ion trajectories in areas with large statistical uncertainty and to utilize the information we can derive from the flight-path of the ion up to a certain depth inside the target. For each ion, the local dopant concentration $C_{loc}$ is checked at certain points of the flight-path (*checkpoints*). At each checkpoint we relate $C_{loc}$ to the current maximum global concentration $C_{max,current}$ by calculating the ratio $C_{loc}/C_{max,current}$. The result is compared with given relative concentration levels (we define ten levels at 0.3, 0.09, 0.027, ..., $0.3^{10}$). Only if the current local concentration falls in an interval below the previous one, a *trajectory split point* is defined at this checkpoint. Therefore our approach is a self-adaptive algorithm because more split points are defined at areas with unsatisfying statistical accuracy. Additional trajectory branches are suppressed, if an ion moves from lower to higher local concentration levels. We store the position of the ion, its energy as well as the vector of velocity and use this data for virtual branches of ion trajectories starting at this split point. In this way, the peripheral areas of the dopant concentration are represented by a much higher number of ion trajectories and the statistical noise is reduced.

Several implementations of this method are conceivable and efficient. We developed three different strategies [40] one shown in Fig. 9. Such a virtual trajectory branch is calculated with the same models and parameters as a regular trajectory, but it starts at the split point with initial conditions obtained from the regular ion. To obtain the correct concentration, a weight is assigned to each branch. The different realizations of the virtual trajectories result from the thermal vibrations of the target atoms [42].

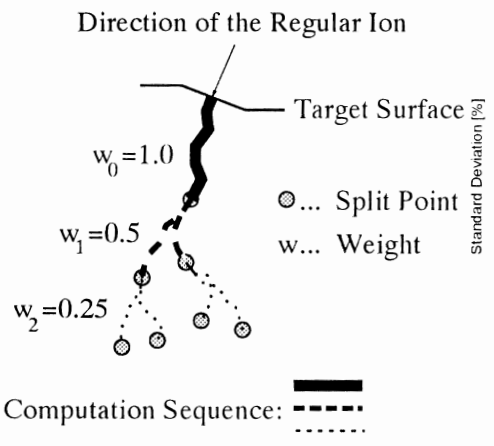


Figure 9: Topological structure of the split-level related split method, the weight of each branch, and the sequence of its calculation

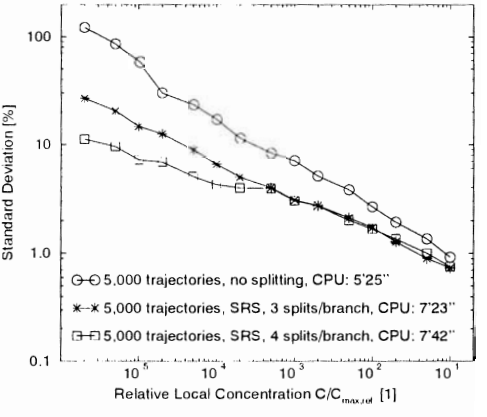


Figure 10: Two-dimensional point response of phosphorus implant, statistical accuracy and CPU time of split-level related split method

To assess the statistical accuracy of the results obtained from the conventional and from the trajectory split methods, we define a mean-square deviation from a reference

distribution. For that reason we carry out a conventional simulation with such a high number of ion trajectories (1,000,000) that statistical fluctuations are negligible in the concentration area considered. As an example, we perform a Monte Carlo simulation of a phosphorus implant at 50keV into (100) oriented single-crystal silicon covered by 2.5nm of oxide to obtain point response distributions.

We present the deviation data for the recursive split-level related split method [40] in Fig. 10 calculated with 5,000 distinct ion trajectories. The relative concentration in this figure is defined as the ratio of $C/C_{max,ref}$, where $C_{max,ref}$ means the maximum concentration of the reference distribution. The computational effort is approximately proportional to the number of distinct ion trajectories and the additional overhead due to trajectory splits is only 25% to 35%.

Further important advantages of the trajectory split method are its lower sensitivity to the local concentration and the opportunity to individualize its error behavior. Increasing the number of splits per each branch, cf. Fig. 10, and/or initializing more than one virtual branch at each split point leads to a significantly smaller error in peripheral areas without effecting the statistic in other regions. In other words there is a chance of optimizing the relation between CPU time and required statistical accuracy for a particular problem.

It should be mentioned that our new strategy is also best suited to compute the collision cascade of a displaced target atom ("recoil"). Depending on the ion energy and the atomic mass ratio of the ion and the recoil some collisions cause a considerable number of recoils which lead to a statistical "over-representation" of such events. The new method offers the possibility to optimize the recoil statistic by a random deletion of recoil trajectories at such places and by splitting them at peripheral areas of the collision cascade.

### 3.4. An Example

The Source-Drain doping in minimum-size transistor designs is an intrinsically three-dimensional problem. Furthermore, in modern shallow-junction processing channeling may affect the device performance [43]. Thus, an ion implantation into a field oxide corner of a conventional LOCOS structure is best suited to demonstrate the merits and the applicability of the *superposition method* and the *trajectory split method*, respectively.

For the simulations we used a phosphorus implant of $5 \cdot 10^{13}\text{cm}^{-2}$ at 40keV. Fig. 11 shows the geometry of the conventional LOCOS. The screening oxide thickness is 10nm and the ion beam was tilted for $-7°$ in the xy-plane. To investigate the channeling effects we cut the LOCOS geometry by a horizontal xz-plane 10nm below the silicon/silicon-dioxide interface. Fig. 12 and Fig. 13 show the amorphous and crystalline mode simulation results for the conventional LOCOS.

From these results follows that in the active region the dopant concentration near the silicon surface is significantly decreased whereas the doping at the periphery (along the bird's beak) remains uneffected due to the dechanneling property of the thicker oxide in that region ($\approx$ 60nm).

The required computational effort for such a rigorous three-dimensional simulation is approximately proportional to the exposed area (*implantation window*) and depends on the energy of the ions. On a HP 9000-735/100 workstation our example takes about 15 hours using the *superposition method* and about one day using the *trajectory split method*. Compared to conventional strategies the speed-up is about a few orders of magnitudes for the amorphous mode and about five for the crystalline mode.

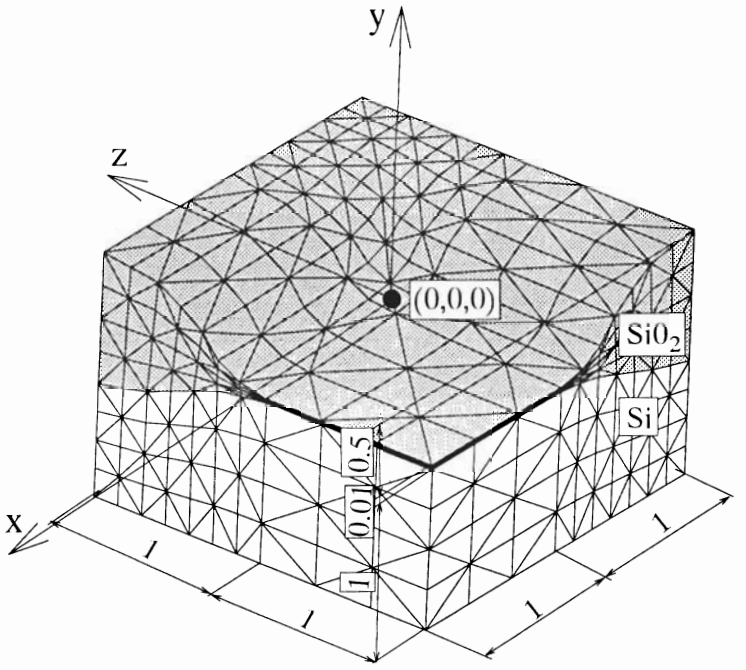


Figure 11: Corner of the conventional LOCOS structure

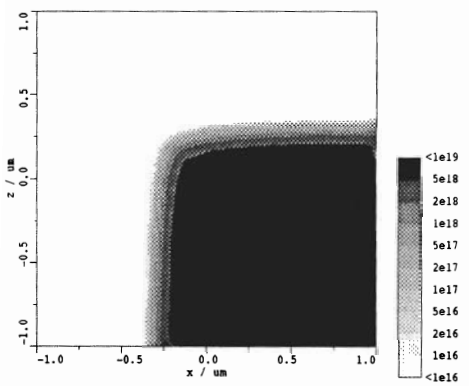


Figure 12: Concentration of phosphorus in $cm^{-3}$ 10nm below the silicon surface (conventional LOCOS geometry is cut by a horizontal xz-plane. amorphous mode)

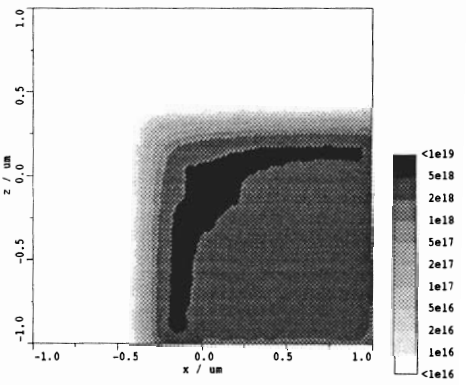


Figure 13: Concentration of phosphorus in $cm^{-3}$ 10nm below the silicon surface (conventional LOCOS geometry is cut by a horizontal xz-plane. crystalline mode)

# 4. Diffusion Processes

The solution of the coupled nonlinear differential equations coming up from diffusion processes usually is done by means of the Finite Element Method respectively the Finite Box Method. Both require a grid, which has to resolve the geometry and the doping profiles in order to achieve a proper discretization. This grid plays the keyrole for solution efficiency, because it determines both the condition and the size of the sparse equation systems. For three-dimensional simulations the size of the systems is the major hurdle, because the numerical solution effort follows in the very best case approximately $n^{3/2}$ with n as the number of unknowns. Therefore the grid has to be optimized in order to fulfill the required accuracy requirements using a minimum number of nodes.

We divide our gridding activities into two parts: generation of an initial grid and adaptation of the grid according to the changed accuracy requirements as the diffusion advances. For initial grid generation we have developed a tetrahedrization module, based on the Delaunay criterion. In order to adapt the grid throughout the diffusion simulation the mixed-element decomposition method has been utilized [44].

## 4.1. Grid Generation

Grid generation plays a keyrole in three-dimensional process and device simulation. The amount of data forces the grid generator to keep the optimal balance between accuracy and efficiency. The geometry has to be partitioned into the smallest number of elements, which still allows accurate solving of the governing equations. These elements have to fulfill certain quality requirements and the gridding process should work entirely automatically. Another challenge lies within a typical TCAD situation, where many simulators have to interact. Tools might generate their own grids which are not valid as an input for subsequent tools. In order to avoid unnecessary re-interpolation of attributes defined on a grid and to minimize gridding efforts after each simulation step the ideal grid generator should also have the capability to read and modify existing grids. For instance, a grid output from one tool has to be made geometry conform before it can be input to the next tool [45].

We use the concept of a "place nodes and link" algorithm [46], which in our opinion offers the best means to deal with the above mentioned tasks. Grid nodes first have to be placed according to local and global grid densities, after which they are linked to yield the grid elements. Note that this is the only approach which allows the input of already existing grid nodes. Thus, grid nodes of different grids can be merged, or previously generated grids can be refined by adding additional grid nodes. The geometry can be arbitrarily complex, the boundary points of the geometry are a direct input to the grid generator. Conformity of the grid with the boundary of the geometry will be discussed in the next section. Many disadvantages of octree-based methods which intrinsically place the grid nodes can be avoided. For instance, the limited flexibility in node placement and the sensitivity to alignment of the geometry with the octree. Even if 2-4-8-Trees [47] are used, anisotropic grid density specifications cannot generally be fulfilled. Finally, the need for special gridding techniques near boundaries to avoid staircase-like representations of the geometry makes the use of octree-based methods less favourable.

After the linking step, the resulting tetrahedral grid elements have to meet the desired quality requirements. There are two degrees of freedom how one can change this element quality. Both *node placement* and the *type of tetrahedrization* have a crucial influence. Unfortunately, they are not entirely independent of each other,

thus, an optimization loop becomes necessary. After the tetrahedrization, the node placement might have to be changed or grid nodes might have to be inserted and the tetrahedrization process has to be repeated. Even if the tetrahedrization process is in some sense optimal (e.g. Delaunay tetrahedrization) elements of poor quality cannot be avoided beforehand. In three dimensions the *aspect ratio* of a simplex can be defined as the ratio of the radii of the circumscribed sphere to the inscribed sphere [48]. Typical elements with poor aspect ratios are [49]:

**Needle:** A tetrahedron with a very long and a very short edge.

**Cap:** A tetrahedron, where the radius of the circumsphere is much larger than the longest edge.

**Sliver:** A tetrahedron consisting of four nearly coplanar points, which are evenly spaced on a great circle of the circumsphere.

Especially in two dimensions interesting dependencies between various optimization criteria (no large angles, no small angles, height) and the Delaunay triangulation have been shown [48]. Essentially, *Steiner points* are added to the initial point set (*Steiner triangulation*) and the Delaunay triangulation of the modified point set is used. Considering a fixed point set in two dimensions the Delaunay triangulation is known to maximize the minimum angle. In three dimensions the Delaunay tetrahedrization (DT) minimizes the maximum radius of a minimum-containment sphere [50]. The minimum-containment sphere is the smallest sphere that contains the tetrahedron. It can be identical to the circumsphere. The need for a general, numerically robust tool to compute the DT for a fixed point set becomes evident. In fact, the DT is the only tetrahedrization which is dual to the well known Voronoi diagram. Thus, if the Box integration method is applied, the DT becomes a *necessary* tool to avoid negative control volumes.

#### 4.1.1. Boundary Conformity

If a general tetrahedral Delaunay grid is required to be conform with the boundary, the boundary has to be represented by a surface triangulation where each triangle fulfills the Delaunay criterion. If and only if at least one empty sphere passing through the three points of a boundary triangle exists, the boundary triangle is a Delaunay triangle. This sphere can have any size. For the Box integration method a stronger criterion has to be satisfied.

**Boundary refinement criterion:** The smallest sphere passing through the three points of a boundary triangle may not contain any other point.

The step in which the input boundary triangles (generally not Delaunay) are modified according to this criterion is called *boundary refinement*. This is a complex two and a half dimensional problem (Literature mostly covers the one and half dimensional case, [45][48][51][52]). Multiple domains pose no difficulties. The boundary refinement step has to be applied to the interfaces as well. Our boundary refinement module uses a combination of flipping boundary triangles (sometimes also called edge-swapping) and inserting additional boundary points to satisfy the above stated criterion: If two adjacent boundary triangles are coplanar and violate the criterion, their common edge can be flipped without having to insert a point. This technique greatly reduces the number of additional inserted points. (A discussion of such local transformations can be found in [53].) Inserting points only becomes necessary along edges of the geometry or in the case of (parallel) planes with small distance compared to the size of the triangles.

### 4.1.2. Delaunay Tetrahedrization

We implemented an incremental tetrahedra construction algorithm (Fig. 14). It can be imagined as "advancing front" that pervades the input geometry. The domain on the backside of the front is entirely tetrahedrized and the domain on the front side not at all. The great advantage of this algorithm is how rigorous it deals with complex input geometries (e.g. multiple connected). It only tetrahedrizes the interior of the geometry, because the advancing front will be stopped by the surface triangulation of the boundary. (As opposed to other algorithms, where a convex hull is tetrahedrized and a subsequent segmentation step is necessary which distinguishes interior and exterior elements.)

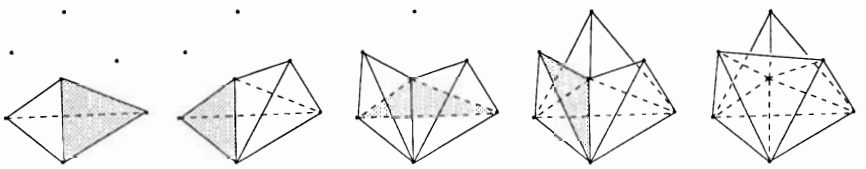

Figure 14: Incremental algorithm

A typical problem for DT algorithms are degeneracies due to cospherical point sets. (If more than four points are located on the perimeter of an empty sphere, the point set is said to be cospherical.) The implemented algorithm allows any number of cospherical points and will not have to add points to deal correctly with these cases.

Another difficulty lies in finite-precision arithmetics. Topological point connectivity evolves from numerical calculations. Thus, numerical errors manifest in topological inconsistencies. The degree of freedom in choosing the point connectivity is spent to satisfy the Delaunay criterion. In order to guarantee a topological correct tessellation the Delaunay criterion need not always be satisfied. A small tolerance $\epsilon$ has to be granted. Sugihara and Iri discussed this topic for the two-dimensional Voronoi diagram in [54]. Our implementation solves this problem. It detects topological inconsistencies that would occur due to a sole adherence to the Delaunay criterion and overrides the criterion by $\epsilon$. The algorithm uses an octree point location method and runs in $O(n \log n)$ time (Fig. 15).

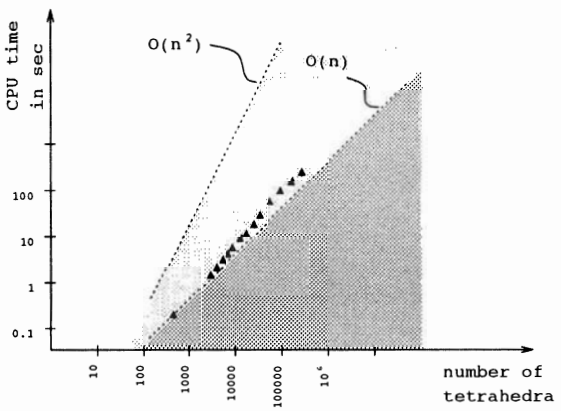


Figure 15: Time complexity (HP 9000-735/100)

So far the point connectivity describing the linking of the grid nodes was of concern. However, in the three-dimensional space the *face connectivity* plays an important role. One face (triangle) connects the two spaces on each side. In the presence of cocircular points in three dimensions (more than three points which are located on an empty circle) the two half spaces on each side of the plane containing the cocircular points can be connected via triangles in more than one way. If on one side of the plane the set of cocircular points is triangulated in a different manner than on the other side, the result is inconsistent face connectivity. Note that the cocircular point set implies the existence of two neighboring sets of cospherical points. The question arises whether it is possible to tetrahedrize a cospherical point set, if its convex hull contains a fixed and given triangulation of cocircular points. Fig. 16 depicts a case where the specified face connectivity cannot be achieved by any tetrahedrization of the interior. This is a similar problem to the un-tetrahedrizable polyhedron mentioned in [48]. It can be solved by local transformations of the tetrahedra, which are connected to the plane containing the cocircular point set.

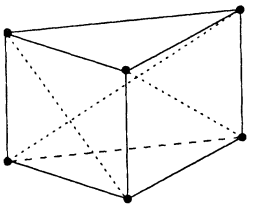

Figure 16: Impossible face-connectivity

## 4.2. Adaptive Gridding Strategy

Once an initial grid is available which resolves the geometry, it has to be adapted to the implanted doping profiles and the adaptation has to be redone as the diffusion advances and the local discretization errors exceed their limits. Several grid updates are necessary throughout a whole simulation. Thus, a fast adaptation algorithm is needed in order to keep the computational overhead for the grid management low. Therefore, we use a recursive element decomposition method.

For recursive refinement algorithms it is indispensable to preserve the grid quality, i.e., the unavoidable degradation of the grid quality has to stay within a limit which is independent of the number of refinements. If we can find a refinement method which keeps the element shape, the above requirement is fulfilled automatically. Unfortunately for tetrahedra no such method exists. However, it is possible to define a two-level splitting method, which preserves the element shape during multiple refinement.

We divide a tetrahedron into four tetrahedra of the same shape and one octahedron. The four tetrahedra are located at the parent's corners and the remaining part has octahedral shape (Fig. 17). An octahedron we divide into six octahedra of the same shape and eight tetrahedra. The six octahedra are located at the parent's corners and the remaining parts have tetrahedral shape (Fig. 18). In order to discretize an octahedron, we split it into eight tetrahedron. each of which has one face of the octahedron as ground plane and the octahedral center as opposite node.

The first refinement step introduces elements with a new aspect ratio. The elements generated by all following refinement steps have either the shape of the tetrahedra or

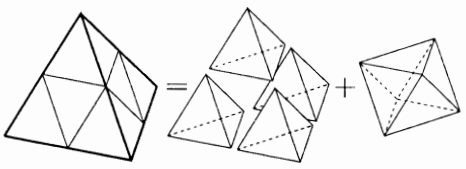

Figure 17: Tessellation for a tetrahedron

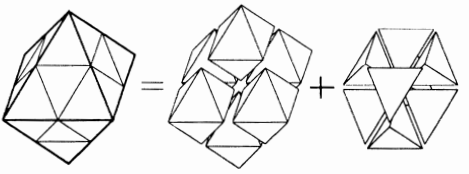

Figure 18: Tessellation for an octahedron

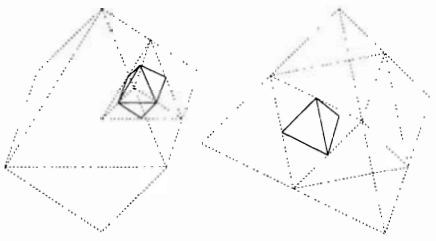

Figure 19: Shape preservation for recursive refinement

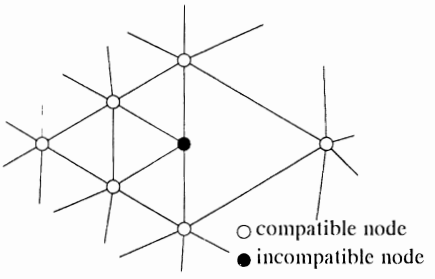


Figure 20: Incompatible elements

the shape of the octahedra which exist after the first refinement step (see Fig. 19). Thus, the element quality is affected only at the first refinement step. Taking into account the discretization of the octahedron permits a reasonable comparison of the octahedral child of the tetrahedron: we compare the element quality of the tetrahedral parent with the element quality of the tetrahedra used for the discretization of the octahedron. It can be shown, that the degradation of the element quality according (11) is limited to a factor of 1/2. A similar comparison of the tetrahedral child with the octahedral parent results in a maximum degradation factor of 1/4.

$$Q = \frac{V}{h_{max}^3} \tag{11}$$

As the refinement is always done locally, unrefined elements may be adjacent to refined ones. These neighboring elements are called incompatible elements, and we define the order of incompatibility as the difference of the refinement levels of two adjacent elements. In our algorithm the order of incompatibility is restricted to one. A two dimensional example of such an incompatible situation is shown in Fig. 20. In order to estimate the grid quality at a compatible node between incompatible elements, we use (12), where $V_i$ are the volumes of all elements incident to the node [55]. It can be shown, that the degradation of this nodal grid quality is limited to a factor of 1/4 for the tetrahedron and 1/8 for the octahedron.

$$Q = \frac{\min(V_i)}{\max(V_i)} \tag{12}$$

### 4.3. Discretization Method

For the practical application of the mixed-element decomposition method, we implemented a finite element method in order to solve (13), where $C_k$ are the quantities to solve and $J_k$ are the according fluxes. The quantities are coupled by the coefficients

$\gamma_{kl}$, which allow modeling of generation and recombination, and by the coefficients $\alpha_{kl}$, which couple the fluxes of the different quantities.

$$\frac{\partial C_k}{\partial t} + \mathrm{div} J_k + \sum_{l=1}^{N_Q} \gamma_{kl} = 0$$

$$J_k = \sum_{l=1}^{N_Q} (a_{kl} \cdot \mathrm{grad} C_l)$$

$$\text{for} \quad k = 1, \ldots, N_Q \tag{13}$$

As discretizing element we use the tetrahedron with linear shape functions. The octahedrons coming from the mixed-element decomposition method are split into eight tetrahedrons for discretization. The weak form of (13) results from the method of Galerkin-weighted residuals [56] and is integrated by means of Gaussian integration using one integration point in the center of the tetrahedron.

In order to estimate the discretization errors, we use a gradient smoothing method [56]. This method uses the shape functions of the elements for a continuos approximation of the piecewise constant gradient of the solution. By twofold integration of the gradient difference along an element the local dosis error of the solution is computed. The decision about local grid refinement respectively coarsening is based on a weighted combination of the local dosis error related to the local dosis and the local dosis error related to the global dosis. The weights allow to control the grid density at high concentration levels nearly independently from the grid density at low concentration levels.

The discretization in time uses the standard finite differences method according the Backward-Euler scheme. Error estimation is used in order to test the accuracy of the previous time step as well as to predict the size of the next time step. The estimation is based on a parabolic approximation of the piecewise linear evolution of the solution at each node. By means of extrapolation the size for the new time step is estimated.

### 4.4. Solution Strategy

For solution of the nonlinear equation systems, we implemented a damped Newton iteration scheme for the coupled equations. To achieve quadratic convergence, we extend the element matrices to the full Fréchet derivative of the nonlinear functional. The resulting linear equation systems are solved iteratively by means of a BICGStab-solver [57] with an incomplete Gauß-elimination for preconditioning. For a particular time step the initial condition for the Newton iteration scheme is obtained by quadratic extrpolation of the solution of the previous time steps. This technique reduces the number of Newton iterations by one for each time step.

Once the grid is adapted to the initial doping profiles, it is successively modified after each time step according to the adaptation criteria. All elements are checked upon their discretization error and are either refined or replaced by their parents. On inserting of new nodes, the solution values are interpolated by a third order interpolation function, which satisfies the continuity condition for the gradients.

### 4.5. An Example

To demonstrate the benefits of the adaptive gridding algorithm, we computed a diffusion step for a Boron channel-implant in a conventional LOCOS-structure at 1000 ° C

and an annealing time of 30min. Figure 11 shows the coarse initial tessellation of the simulation region where the field oxide is on top of the silicon bulk. The channel implant has been computed by Monte-Carlo ion implantation [3] with an energy of 20keV and a dose of 1e14cm$^{-2}$. The initial grid has been adapted to the initial profile for a discretization error limit of 1% relative to the total implanted dosis. The grid for the silicon region with the distribution is shown in Fig. 21 and consists of 9534 nodes and 19259 elements.

As the diffusion advances the steep gradients are smoothed and therefore, the grid was reduced continuously by the automatic adaptation algorithm. Thus, the final grid at the end of the simulation consists of only 8093 nodes and 15636 elements (Fig. 22).

For the needed 34 time steps the program consumed a CPU-time of 41 minutes on an HP 9000-735/100 workstation and used approximately 32MB of memory, which shows, that fully three-dimensional diffusion simulation with a controlled discretization error is feasible.

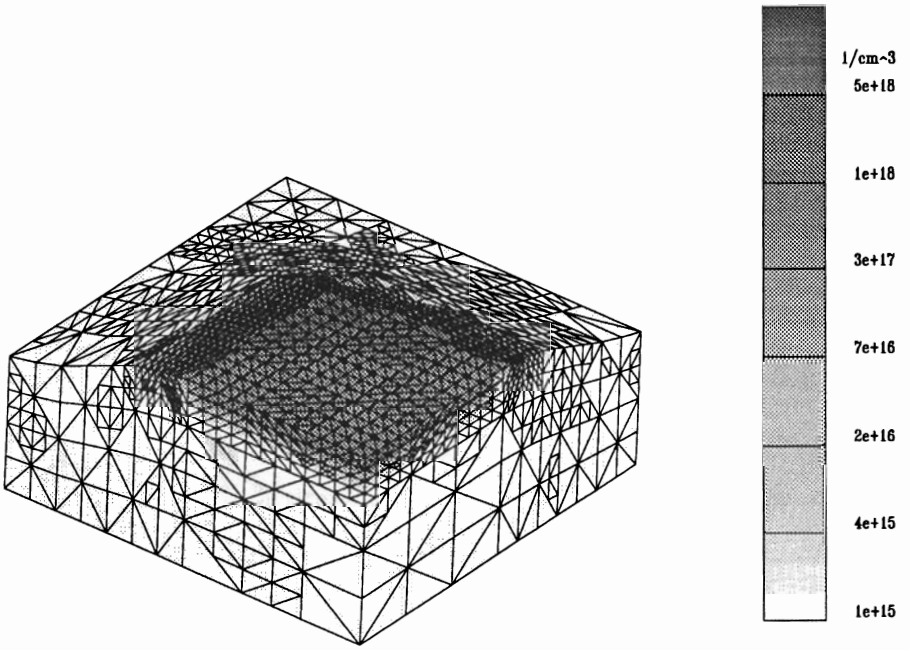


Figure 21: The implanted Boron profile in the Silicon region

## 5. Oxidation

The physics of thermal oxidation results in a high number of coupled differential equations (typically 6-10 for a three-dimensional simulation). Even for two-dimensional simulations the equation systems resulting from a coupled solution are very large. The standard methods using Newton methods or decoupled iteration schemes combined with iterative solvers for the linear systems require significant amounts on computational resources. Therefore, we investigated other (preferably matrix-free) algorithms and found the multigrid method feasible for oxidation problems.

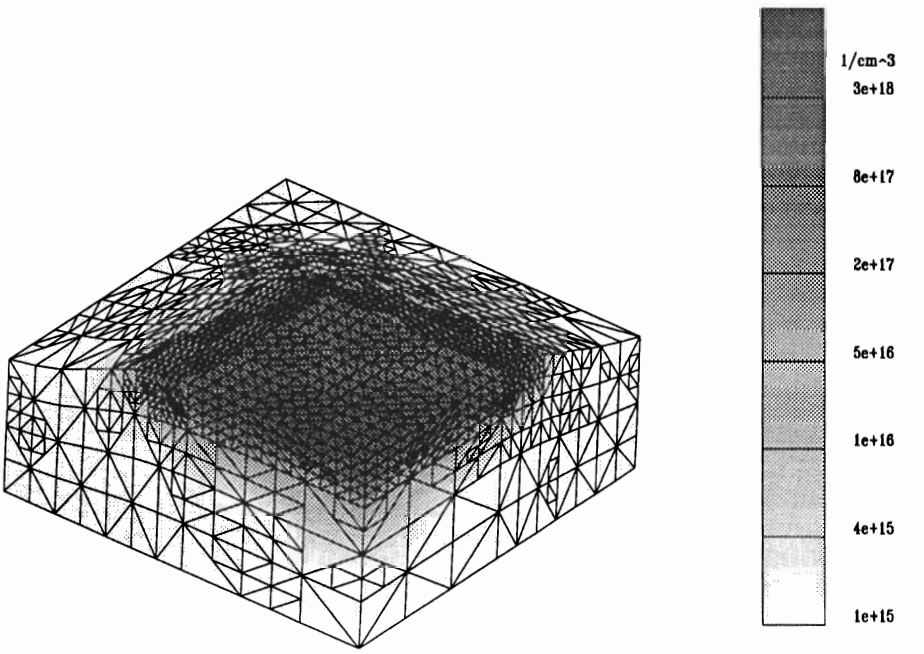


Figure 22: Boron profile after 30min. annealing at 1000 ° C

We are currently evaluating the efficiency of multigrid methods for mechanical equations, and the results seem very promising. We compared different methods in order to solve the mechanical stress/strain equations, including the nonlinearity coming up from large displacement. For a problem with about 16000 unknowns, where the iterative solver diverged on solving the linear system, the Gaußsolver consumed about 2 min CPU time and 60 MB memory on an DEC-Alpha workstation for one matrix inversion within the Newton scheme which needed 5 iterations, resulting in 10 min overall CPU time. The multigrid algorithm was able to solve the whole nonlinear problem within about 30 sec and needed just 8 MB memory.

Another advantage of the multigrid method is, that the solution effort is growing just linearly with the number of nodes. This feature makes the multigrid method appearing best suited for use in three-dimensional process simulation with coupled stress/strain equations, where high node counts are unavoidable.

## 6. Conclusion

Despite the large variety of problems coming up from three-dimensional process simulation, several steps can already be simulated reasonably. Topography simulation of etching and deposition processes and ion implantation simulation deliver accurate results and consider the physical effects sufficiently. The development of diffusion and oxidation process simulation tools is still at the beginning. However, we have shown reasonable solutions to the grid adaptation and grid generation problem, which allows us to perform simulation of dopant diffusion with a controlled discretization error and reasonable demands on computational resources.

Furthermore, the coupling of different simulators is a must. The problems arising from different data representation formats have been pointed out and solutions for converting polygonal to cellular based geometries and vice versa have been shown.

## Acknowledgement

TU Vienna wants to acknowledge important support by Austria Mikro Systeme AG, Unterpremstätten, Austria; Digital Equipment Corp., Hudson, USA; Hitachi Ltd., Tokyo, Japan; LSI Logic Corp., Milpitas, USA; Motorola Inc., Austin, USA; National Semiconductor Corp., Santa Clara, USA; Philips B.V., Eindhoven, The Netherlands; Siemens AG, München, Germany; Sony Corp., Atsugi, Japan.

Part of this work was carried out in cooperation between PROMPT (JESSI project BT8B) and ADEQUAT (JESSI project BT11) and has been funded by the EU as ESPRIT projects No. 8150 and 8002, respectively.

## References

[1] E. Strasser, K. Wimmer, and S. Selberherr. A New Method for Simulation of Etching and Deposition Processes. In *1993 International Workshop on VLSI Process and Device Modeling*, pp 54–55, 1993.

[2] E.W. Scheckler and A.R. Neureuther. Models and Algorithms for Three-Dimensional Topography Simulation with SAMPLE-3D. *IEEE Transactions on Computer-Aided Design of Integrated Circuits*, 13:219–230, 1994.

[3] W. Bohmayr, G. Schrom, and S. Selberherr. Trajectory Split Method for Monte Carlo Simulation of Ion Implantation Demonstrated by Three-Dimensional Poly-Buffered LOCOS Field Oxide Corners. In *Int.Symposium on VLSI Technology, Systems, and Applications*, Taipei, 1995.

[4] A.M. Mazzone and G. Rocca. Three-Dimensional Monte Carlo Simulations – Part I: Implanted Profiles for Dopants in Submicron Devices. *IEEE Trans.Computer-Aided Design*, CAD-3(1):64–71, 1984.

[5] A.M. Mazzone. Three-Dimensional Monte Carlo Simulations – Part II: Recoil Phenomena. *IEEE Trans.Computer-Aided Design*, CAD-4(1):110–117, 1985.

[6] S. Odanaka, H. Umimoto, M. Wakabayashi, and H. Esaki. SMART-P: Rigorous Three-Dimensional Process Simulator on a Supercomputer. *IEEE Trans.Computer-Aided Design*, 7(6):675–683, 1988.

[7] C. S. Sun, O. K. Kwon, C. G. Hwang, and H. J. Hwang. Three-Dimensional Numerical Simulation for Low Dopant Diffusion in Silicon. In S. Selberherr, H. Stippel, and E. Strasser, editors, *Simulation of Semiconductor Devices and Processes*, pp 413–416. Springer-Verlag Wien New York, 1993.

[8] M. E. Law. Challenges to Achieving Accurate Three-Dimensional Process Simulation. In S. Selberherr, H. Stippel, and E. Strasser, editors, *Simulation of Semiconductor Devices and Processes*, pp 1–8. Springer-Verlag Wien New York, 1993.

[9] E.W. Scheckler, N.N. Tam, A.K. Pfau, and A.R. Neureuther. An Efficient Volume-Removal Algorithm for Practical Three-Dimensional Lithography Simulation with Experimental Verification. *IEEE Transactions on Computer-Aided Design of Integrated Circuits*, 12:1345–1356, 1993.

[10] W. Henke, D. Mewes, M. Weiß, G. Czech, and R. Schließl-Hoyler. A Study of Reticle Defects Imaged into Three-Dimensional Developed Profiles of Positive Photoresists Using the SOLID Lithography Simulator. *Microelectronic Engineering*, 14:283–297, 1991.

[11] Y. Hirai, S. Tomida, K. Ikeda, M. Sasago, M. Endo, S. Hayama, and N. Nomura. Three-Dimensional Resist Process Simulator PEACE (Photo and Electron Beam Lithography Analyzing Computer Engineering System). *IEEE Transactions on Computer-Aided Design of Integrated Circuits*, CAD-10:802–807, 1991.

[12] T. Ishizuka. Bulk Image Effects of Photoresist in Three-Dimensional Profile Simulation. *Journal for Computation and Mathematics in Electrical and Electronic Engineering*, 10:389–399, 1991.

[13] K.K.H. Toh, A.R. Neureuther, and E.W. Scheckler. Algorithms for Simulation of Three-Dimensional Etching. *IEEE Transactions on Computer-Aided Design of Integrated Circuits*, 13:616–624, 1994.

[14] J. Pelka. Three-dimensional Simulation of Ion-Enhanced Dry-Etch Processes. *Microelectronic Engineering*, 14:269–281, 1991.

[15] S. Tazawa, F.A. Leon, G.D. Anderson, T. Abe, K. Saito, A. Yoshii, and D.L. Scharfetter. 3-D Topography Simulation of Via Holes Using Generalized Solid Modeling. In *Int.Electron Devices Meeting*, pp 173–176, 1992.

[16] K.K.H. Toh. *Algorithms for Three-Dimensional Simulation of Photoresist Development.* PhD thesis, University of California, Berkeley, 1990.

[17] J.J. Helmsen, E.W. Scheckler, A.R. Neureuther, and C.H. Séquin. An Efficient Loop Detection and Removal Algorithm for 3D Surface-Based Lithography Simulation. In *Workshop on Numerical Modeling of Processes and Devices for Integrated Circuits*, pp 3–8, 1992.

[18] E. Strasser, G. Schrom, K. Wimmer, and S. Selberherr. Accurate Simulation of Pattern Transfer Processes Using Minkowski Operations. *IEICE Transactions on Electronics*, E77-C:92–97, 1994.

[19] E. Strasser and S. Selberherr. Algorithms and Models for Cellular Based Topography Simulation. *IEEE Trans.Computer-Aided Design*, accepted for publication, 1995.

[20] W.G. Oldham, A.R. Neureuther, C. Sung, J.L. Reynolds, and S.N. Nandgaonkar. A General Simulator for VLSI Lithography and Etching Processes: Part II - Application to Deposition and Etching. *IEEE Trans.Electron Devices*, ED-27:1455–1459, 1980.

[21] J. Lorenz, J. Pelka, H. Ryssel, A. Sachs, A. Seidel, and M. Svoboda. COMPOSITE - A Complete Modeling Program of Silicon Technology. *IEEE Transactions on Computer-Aided Design of Integrated Circuits*, CAD-4:421–430, 1985.

[22] S. Tazawa, S. Matsuo, and K. Saito. Unified Topography Simulator for Complex Reaction including both Deposition and Etching. In *1989 Symposium on VLSI Technology*, pp 45–46, 1989.

[23] I.A. Blech. Evaporated Film Profiles Over Steps in Substrates. *Thin Solid Films*, 6:113–118, 1970.

[24] W.E. Lorenson and H.E. Cline. Marching Cubes: A High Resolution 3D Surface Construction Algorithm. *Computer Graphics*, 21:163–169, 1987.

[25] W.J. Schroeder, J.A. Zarge, and W.E. Lorenson. Decimation of Triangle Meshes. *Computer Graphics*, 26:65–70, 1992.

[26] M.T. Robinson and O.S. Oen. Computer Studies of the Slowing Down of Energetic Atoms in Crystals. *Physical Rev.*, 132(6):2385–2398, 1963.

[27] M.T. Robinson and I.M. Torrens. Computer Simulation of Atomic-Displacement Cascades in Solids in the Binary-Collision Approximation. *Physical Rev. B*, 9:5008–5024, 1974.

[28] J.P. Biersack and L.G. Haggmark. A Monte Carlo Computer Program for the Transport of Energetic Ions in Amorphous Targets. *Nucl.Instr.Meth.*, 174:257–269, 1980.

[29] J.F. Ziegler, J.P. Biersack, and U. Littmark. The Stopping and Range of Ions in Solids. Pergamon Press, 1985.

[30] G. Hobler and S. Selberherr. Monte Carlo Simulation of Ion Implantation into Two- and Three-Dimensional Structures. *IEEE Trans.Computer-Aided Design*, 8(5):450–459, 1989.

[31] H. Stippel and S. Selberherr. Three Dimensional Monte Carlo Simulation of Ion Implantation with Octree Based Point Location. In *Int. Workshop on VLSI Process and Device Modeling (1993 VPAD)*, pp 122–123, Nara, Japan, 1993. Jap.Soc.Appl.Phys.

[32] H. Stippel and S. Selberherr. Monte Carlo Simulation of Ion Implantation for Three-Dimensional Structures Using an Octree. *IEICE Transactions on Electronics*, E77-C(2):118–123, 1994.

[33] W.D. Fellner. *Computergrafik*. Reihe Informatik. B.I. Wissenschaftsverlag, 1992.

[34] M. Mäntyla. *An Introduction to Solid Modeling*. Computer Science Press, 1988.

[35] S. Abramowski and H. Müller. *Geometrisches Modellieren*. B.I. Wissenschaftsverlag, Mannheim, 1991.

[36] K.M. Klein, C. Park, and A.F. Tasch. Monte Carlo Simulation of Boron Implantation into Single-Crystal Silicon. *IEEE Trans.Electron Devices*, 39(7):1614–1621, 1992.

[37] G. Hobler, H. Pötzl, L. Gong, and H. Ryssel. Two-Dimensional Monte Carlo Simulation of Boron Implantation in Crystalline Silicon. In W. Fichtner and D. Aemmer, editors, *Simulation of Semiconductor Devices and Processes*, volume 4, pp 389–398, Konstanz, 1991.

[38] H. Ryssel, G. Prinke, K. Haberger, K. Hoffmann, K. Müller, and R. Henkelmann. Range Parameters of Boron Implanted into Silicon. *Applied Physics A*, 24:39–43, 1981.

[39] S.-H. Yang, D. Lim, S. Morris, and A.F. Tasch. A More Efficient Approach for Monte Carlo Simulation of Deeply-Channeled Implanted Profiles in Single-Crystal Silicon. In *Int. Workshop on Numerical Modeling of Processes and Devices for Integrated Circuits NUPAD V*, pp 97–100. Honolulu, 1994.

[40] W. Bohmayr, A. Burenkov, J. Lorenz, H. Ryssel, and S. Selberherr. Statistical Accuracy and CPU Time Characteristic of Three Trajectory Split Methods for Monte Carlo Simulation of Ion Implantation. In *Simulation of Semiconductor Devices and Processes*, Erlangen, 1995.

[41] A. Phillips and P.J. Price. Monte Carlo Calculations on Hot Electron Tails. *Appl.Phys.Lett.*, 30(10):528–530, 1977.

[42] M. Jaraiz, J. Arias, E. Rubio, L.A. Marques, L. Pelaz, L. Bailon, and J. Barbollan. Dechanneling by Thermal Vibrations in Silicon Ion Implantation. In *X International Conference on Ion Implantation Technology*, pp Abstract P–2.19, 1994.

[43] W. Bohmayr, G. Schrom, and S. Selberherr. Investigation of Channeling in Field Oxide Corners by Three-Dimensional Monte Carlo Simulation of Ion Implantation. In *Int.Conference on Solid-State and Integrated-Circuit Technology*, Beijing, 1995.

[44] E. Leitner and S. Selberherr. Three-Dimensional Grid Adaptation Using a Mixed-Element Decomposition Method. In *Simulation of Semiconductor Devices and Processes*, Erlangen, 1995.

[45] S. Halama. *The Viennese Integrated System for Technology CAD Applications — Architecture and Critical Software Components.* PhD thesis, Technische Universität Wien, 1994.

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

[47] P. Conti. *Grid Generation for Three-Dimensional Semiconductor Device Simulation.* Hartung-Gorre, 1991.

[48] 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. World Scientific, 1992.

[49] T.J. Baker. Element Quality in Tetrahedral Meshes. *7th Int. Conf. on Finite Element Models in Flow Problems*, Huntsville, Alabama, 1989.

[50] V.T. Rajan. Optimality of the Delaunay Triangulation in $R^d$. *Proc. 7th ACM Symp. Comp. Geometry*, pp 357–363, 1991.

[51] L. Nackman and V. Srinivasan. Point Placement for Delaunay Triangulation of Polygonal Domains. *Proc. 3rd Canadian Conf. Comp. Geometry*, pp 37–40, 1991.

[52] A. Saalfeld. Delaunay Edge Refinements. *Proc. 3rd Canadian Conf. Comp. Geometry*, pp 33–36, 1991.

[53] B. Joe. Three-Dimensional Triangulations from Local Transformations. *SIAM J.Sci.Stat.Comput.*, 10(4):718–741, 1989.

[54] K. Sugihara and M. Iri. Construction of the Voronoi Diagram for One Million Generators in Single-Precision Arithmetic. *Proc.IEEE*, 80(9):1471–1484, 1992.

[55] R.E. Bank. *PLTMG: A Software Package for Solving Elliptic Partial Differential Equations*, volume 7 of *Frontiers in Applied Mathematics*. SIAM, 1990.

[56] O.C. Zienkiewicz and R.L. Taylor. *The Finite Element Method.* McGraw-Hill, fourth edition, 1989.

[57] H.A. van der Vorst. BI-CGSTAB: A Fast and Smoothly Converging Variant of BI-CG for the Solution of Nonsymmetric Linear Systems. *SIAM J.Sci.Stat.Comput.*, 13(2):631–644, 1992.