Body-Fitting Coordinate Generation for Two-Dimensional Process-Simulation

K. Wimmer, R. Bauer, S. Selberherr

Institute for Microelectronics
Technical University Vienna
Vienna, Austria

Abstract

For the development of VLSI devices the simulation of nonplanar fabrication steps is an absolute necessity. A transformation from physical space to computational space has been implemented in our 2D Process-Simulator PROMIS to deal with nonplanar structures. This transformation is accomplished by specifying a generalized boundary fitted coordinate system, which maps the nonplanar physical domain (x, y) to a rectangular computational domain (u, v). Grid generation techniques have been investigated systematically on their applicability to process simulation problems. The implemented methods can be classified in: (1) algebraic, (2) elliptic, and (3) variational methods, and will be discussed in subsequent sections.

1 Introduction

Many applications dealing with partial differential equations (PDE) require an accurate numerical representation of the boundary conditions. Any kind of boundary condition can be easily implemented in the case the boundary coincides with a coordinate line. Finite difference expressions at, and adjacent to the boundary may be applied using solely grid points at the intersections of coordinate lines. Interpolation is no longer required. Generating a curvilinear coordinate system with coordinate lines coincident with all boundaries is thus an essential part of a numerical solution.

Analytical formulas for body fitting coordinates can be given for relatively simple geometries only, e.g. circular areas, spheres, cylinders, etc. For instance polar coordinates (r, φ) (Eq. (1)) matching circular areas (Fig. 1). In this case the solution of the Laplace equation in cartesian coordinates (Eq. (2)) corresponds to the solution of Eq. (3) in the body fitted polar coordinates.

$$\begin{aligned} x(r,\varphi) &= r \cdot \cos\varphi \\ y(r,\varphi) &= r \cdot \sin\varphi \end{aligned} \tag{1}$$

$$\frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} = 0 \tag{2}$$

$$\frac{1}{r} \cdot \frac{\partial}{\partial r}\left(r \cdot \frac{\partial f}{\partial r}\right) + \frac{1}{r^2} \cdot \frac{\partial^2 f}{\partial \varphi^2} = 0 \tag{3}$$

The general coordinate transformation from physical domain (x, y) to the computational domain (u, v) is given by $u = u(x, y)$, $v = v(x, y)$. Similarly, the inverse transformation is given by $x = x(u, v)$, $y = y(u, v)$ (Fig. 2). We try to find the solution f of our problem in the computational domain $f(u, v)$ rather than in the physical domain $f(x, y)$. Derivatives are transformed as in Eq. (4), where J is the Jacobian of the transformation, $J = x_u y_v - x_v y_u$.

$$\begin{aligned} \frac{\partial f}{\partial x} = f_x &= (y_v f_u - y_u f_v)/J \\ \frac{\partial f}{\partial y} = f_y &= (-x_v f_u + x_u f_v)/J \end{aligned} \tag{4}$$

The basic concept of the transformation is to generate transformation functions so that all boundaries are coincident with coordinate lines. Once the (u, v) system is obtained somehow, any set of partial differential equations can be solved on this body fitted coordinate system by solving the transformed equations in the rectangular computational domain. So the main task is to generate the *body fitted coordinates.* In sections 2 – 4 different generation techniques are presented.

2 Algebraic Methods

The simplest grid generation technique is the algebraic method. An algebraic relation is used to map the grid points of the computational domain to those of the physical domain. This is accomplished by applying an interpolation scheme between the specified boundary points and the interior grid points [1].

2.1 Unidirectional Interpolation

Unidirectional interpolation means the interpolation in one curvilinear coordinate-direction only. As an example we consider a physical domain depicted in Fig. 3. In our case $\vec{r}_1 = \vec{r}(v_1)$ and $\vec{r}_2 = \vec{r}(v_2)$ are two points in opposite boundaries.

In the linear case, one family of grid lines resembles straight lines connecting corresponding boundary points with linear interpolation (Eq. (5)).

$$\vec{r}(v) = \frac{v_2 - v}{v_2 - v_1} \cdot \vec{r}_1 + \frac{v - v_1}{v_2 - v_1} \cdot \vec{r}_2 \tag{5}$$

In the general form (Eq. (6)) the interpolation function $\vec{r}(v)$ matches the N locations of certain interior grid points $\vec{r}_i = \vec{r}(v_i)$. In this case the interpolation polynomials Φ_i are Lagrange polynomials (Eq. (7)).

$$\vec{r}(v) = \Phi_1(v) \cdot \vec{r}_1 + \ldots + \Phi_N(v) \cdot \vec{r}_N \tag{6}$$

$$\Phi_i(v) = \prod_{\substack{l=1 \\ l \neq i}}^{N} \frac{v - v_l}{v_i - v_l} \tag{7}$$

The Lagrange interpolation is only able to match function values $\vec{r_i}$. To control other properties (e.g. orthogonality or grid spacing) it is necessary to match the first derivative $\vec{t_i} = \vec{r_i}'$, too (Fig. 4). Using Hermite interpolation (Eq. (8)) it is possible to match function $\vec{r}$, as well as first derivative $\vec{t}$.

$$\vec{r}(v) = \sum_{i=1}^{N} \tilde{\Phi}_i(v) \cdot \vec{r_i} + \sum_{i=1}^{N} \tilde{\Psi}_i(v) \cdot \vec{t_i} \tag{8}$$

$$\tilde{\Phi}_i(v) = \left(1 - 2\Phi'(v_i) \cdot \frac{v - v_i}{v_{i+1} - v_i}\right) \cdot \Phi_i^2(v) \tag{9}$$

$$\tilde{\Psi}_i(v) = \left(\frac{v - v_i}{v_{i+1} - v_i}\right) \cdot \Phi_i^2(v) \tag{10}$$

The degree of the polynomial is increasing with each additional condition or point to be matched. Since polynomials of higher degrees exhibit considerable oscillations, such procedures failed in our applications.

2.2 Multidirectional Interpolation

Most of the published two dimensional interpolation schemes are strongly dependent on a special geometry, e.g. airfoils, and therefore inappropriate for process simulation purposes. For our problems we need a general scheme being applicable to a large variety of geometries.

The concepts of unidirectional interpolation can be extended to the multidirectional case. We consider a unidirectional interpolation function individual in each curvilinear direction. Eq. (11) matches the boundaries $\vec{r}(u_1, v)$ and $\vec{r}(u_N, v)$, whereas Eq. (12) matches $\vec{r}(u, v_1)$ and $\vec{r}(u, v_N)$.

$$\vec{r}(u, v) = \sum_{i=1}^{N} \Phi_i(u) \cdot \vec{r}(u_i, v) \tag{11}$$

$$\vec{r}(u, v) = \sum_{j=1}^{N} \Phi_j(v) \cdot \vec{r}(u, v_j) \tag{12}$$

Eq. (13) matches any of the four boundaries, and is called *transfinite interpolation.*

$$\begin{aligned} \vec{r}(u, v) &= \sum_{i=1}^{N} \Phi_i(u) \cdot \vec{r}(u_i, v) + \sum_{j=1}^{M} \Phi_j(v) \cdot \vec{r}(u, v_j) \\ &- \sum_{i=1}^{N} \sum_{j=1}^{M} \Phi_i(u) \cdot \Phi_j(v) \cdot \vec{r}(u_i, v_j) \end{aligned} \tag{13}$$

This interpolation function with $N = M = 2$ and the Lagrange interpolation polynomials (Eq. (7)) was used for the grid in Fig. 5.

The transfinite interpolation is a fast and flexible grid generation technique but by far not as reliable as elliptic and variational methods. As Fig. 6 shows, this method is unable to avoid irregular grids, i.e. intersections of grid lines of the same type. For this reason we use the transfinite interpolation either for the generation of an initial grid for the other methods, or in cases of frequent regeneration of the grid and not too complex geometry (e.g. for the moving boundary problem during local oxidation).

3 Elliptic Methods

As the basic idea is to generate transformation functions $u(x,y)$ and $v(x,y)$ in a way that all boundaries coincide with coordinate lines, the body fitting coordinates (u,v) are taken as solutions of an elliptic boundary value problem [4]. For each curvilinear coordinate we solve an elliptic PDE with given values at the boundaries, e.g. $v = v_1 = \text{const.}$ in Γ_1 (see Fig. 3).

We consider Laplace equation (Eq. (14)) as a generating system with Dirichlet boundary conditions, $v = v_1$ on Γ_1, $v = v_2$ on Γ_2.

$$\begin{aligned} u_{xx} + u_{yy} &= 0 \\ v_{xx} + v_{yy} &= 0 \end{aligned} \tag{14}$$

Since we perform all numerical computations in the rectangular transformed domain, the dependent and independent variables are interchanged. Thus we get Eq. (15) with the transformed boundary conditions $x = x(u, v_1)$ and $y = y(u, v_1)$ are in Γ_1, and analogous relations in Γ_2 – Γ_4. etc.

$$\begin{aligned} \alpha x_{uu} - 2\beta x_{uv} + \gamma x_{vv} &= 0 \\ \alpha y_{uu} - 2\beta y_{uv} + \gamma y_{vv} &= 0 \end{aligned} \tag{15}$$

$$\alpha = x_v^2 + y_v^2 \qquad \beta = x_u x_v + y_u y_v \qquad \gamma = x_u^2 + y_u^2$$

This nonlinear elliptic system with Dirichlet boundary conditions is discretized using nine-point finite differences. The resulting system of coupled nonlinear algebraic equations is solved using a successive over-relaxation algorithm (SOR) [2].

In Fig. 7 the basic behaviour of a Laplacian grid is shown. Grid lines generated by the Laplacian system (Eq. (14)) gather at convex boundaries and become sparse at concave ones.

To overcome this behaviour we introduced source terms P, Q. The generating system becomes a Poisson-type system (Eq. (16)).

$$u_{xx} + u_{yy} = P$$

$$v_{xx} + v_{yy} = Q \tag{16}$$

Transforming Eq. (16) to the computational domain we get Eq. (17).

$$\begin{aligned} \alpha \cdot x_{uu} - \beta \cdot x_{uv} + \gamma \cdot x_{vv} &= -J \cdot (P \cdot x_u + Q \cdot x_v) \\ \alpha \cdot y_{uu} - \beta \cdot y_{uv} + \gamma \cdot y_{vv} &= -J \cdot (P \cdot y_u + Q \cdot y_v) \end{aligned} \tag{17}$$

The source terms P and Q are used to control grid spacing and orthogonality at the boundaries [3]. To derive expressions for the source terms we specify a grid spacing d_r and an angle at the boundary α_r (usually 90°). At any point of the body the existing spacing is calculated using Eq. (18), and the existing angle at each boundary point can be calculated easily by the scalar product of the tangent vectors (Eq. (19)).

$$d = |\vec{r}(u, v_{i+1}) - \vec{r}(u, v_i)| \tag{18}$$

$$\alpha = \arccos\left(\frac{\vec{r_u} \cdot \vec{r_v}}{|\vec{r_u}||\vec{r_v}|}\right) \tag{19}$$

We start with a Laplacian grid generation system (Eq. (20)) and solve the system of nonlinear PDEs (Eq. (17)) applying a SOR algorithm. Eq. (21) and Eq. (22) are used to update the source terms P and Q on boundary Γ_1 and Γ_2.

$$P^0 = Q^0 = 0 \tag{20}$$

$$P^{k+1} = P^k \mp \epsilon_p \cdot \arctan\left(\frac{\alpha_r - \alpha}{\alpha_r}\right) \tag{21}$$

$$Q^{k+1} = Q^k \pm \epsilon_q \cdot \arctan\left(\frac{d_r - d}{d_r}\right) \tag{22}$$

The upper signs in Eq. (21) and Eq. (22) relate to boundary Γ_1, the lower ones to Γ_2. For the boundaries Γ_3 and Γ_4 we have to exchange the correction terms for P^k and Q^k.

We interpolate these new source terms between the boundaries in the interior of the domain, and solve Eq. (17) for the modified source functions. We repeat this procedure until the desired grid quality is achieved (Fig. 8).

Numerous test cases proofed, that this method is very reliable in process simulation applications. This generation method turned out to be highly stable and the quality of the grids with respect to overall smoothness and boundary orthogonality is excellent Fig. 9. Note that a nonuniform computational grid was used in this example.

4 Variational Methods

Variational methods offer the opportunity to get control over several grid properties (e.g. orthogonality, smoothness, etc.) [5]. We minimize a weighted linear combination ($I = \sum w_i I_i$) of the functionals of the different properties. The functionals I_i are integral functionals defined in this section. Three functionals are presented, a criterion for (i) the spacing between the grid lines (*smoothness*), (ii) the *area* of the grid cells, and (iii) the *orthogonality* of the grid lines. The minimization problem is solved by calculating the Euler-Lagrange (EL) equations for the variational problem.

Smoothness Control

The integral functional measuring the spacing between grid lines is given in Eq. (23), where ∇ denotes the gradient in the (x, y) variables.

$$I_s = \int \left((\nabla u)^2 + (\nabla v)^2 \right) \, dx \, dy \tag{23}$$

The EL equations for the minimization problem (I_s) resemble the nonlinear coupled elliptic system Eq. (24), with the shorthands Eq. (25). If $A^2 - BC \neq 0$, the EL equations can be rewritten as Eq. (26).

$$\begin{aligned} B(ex_{uu} - 2fx_{uv} + gx_{vv}) &- A(ey_{uu} - 2fy_{uv} + gy_{vv}) = 0 \\ A(ex_{uu} - 2fx_{uv} + gx_{vv}) &- C(ey_{uu} - 2fy_{uv} + gy_{vv}) = 0 \end{aligned} \tag{24}$$

$$\begin{aligned} A &= x_u y_u + x_v y_v & B &= y_u^2 + y_v^2 & C &= x_u^2 + x_v^2 \\ e &= (x_v^2 + y_v^2)/J^3 & f &= (x_u x_v + y_u y_v)/J^3 & g &= (x_u^2 + y_u^2)/J^3 \end{aligned} \tag{25}$$

$$\begin{aligned} ex_{uu} - 2fx_{uv} + gx_{vv} &= 0 \\ ey_{uu} - 2fy_{uv} + gy_{vv} &= 0 \end{aligned} \tag{26}$$

Eq. (26) is identical to the Laplacian generation system (Eq. (15)).

Area Control

The integral functional that measures the area of the grid cells is given in Eq. (27), recalling that $dx\,dy = J\,du\,dv$ leads to Eq. (28).

$$I_a = -\frac{1}{2} \int J^2 \, du \, dv \tag{27}$$

$$I_a = -\frac{1}{2} \int J \, dx \, dy \tag{28}$$

From this formulation the EL equations Eq. (29) can be derived.

$$\begin{aligned}(y_v - y_u)J + J_u y_v - J_v y_u &= 0 \\ (x_u - x_v)J + J_v x_u - J_u x_v &= 0\end{aligned} \tag{29}$$

Orthogonality Control

The integral functional to measure the orthogonality of the grid lines is shown in Eq. (30). The integrand in Eq. (30) can be calculated using the scalar product of the tangent vectors of the grid lines.

$$I_o = -\frac{1}{2}\int (x_u x_v + y_u y_v)^2 \, du \, dv \tag{30}$$

The EL equations for the orthogonality functional I_o are given in Eq. (31) using the shorthands of Eq. (32).

$$\begin{aligned}b_1 x_{uu} + b_2 x_{uv} + b_3 x_{vv} + a_1 y_{uu} + a_2 y_{uv} + a_3 y_{vv} &= 0 \\ a_1 x_{uu} + a_2 x_{uv} + a_3 x_{vv} + b_1 y_{uu} + b_2 y_{uv} + b_3 y_{vv} &= 0\end{aligned} \tag{31}$$

$$\begin{array}{ccc} a_1 = x_v y_v & b_1 = x_v^2 & c_1 = y_v^2 \\ a_2 = x_u y_v + x_v y_u & b_2 = 2(2x_u x_v + y_u y_v) & c_2 = 2(x_u x_v + 2y_u y_v) \\ a_3 = x_u y_u & b_3 = x_u^2 & c_3 = y_u^2 \end{array} \tag{32}$$

The functionals described above are combined to a general measure I for the grid (Eq. (33)). This resulting functional is now to be minimized.

$$\begin{gathered} I = w_s \cdot I_s + w_o \cdot I_o + w_a \cdot I_a \\ w_s \geq 0, \quad w_o \geq 0, \quad w_a \geq 0 \\ w_s + w_o + w + a = 1 \\ \text{minimize} \quad I \end{gathered} \tag{33}$$

The minimization problem is solved by evaluating the EL equations. The system of (nonlinear) elliptic partial differential equations (Eq. (24), Eq. (29) and Eq. (31)) is discretized using nine-point finite differences. The resulting system of coupled nonlinear algebraic equations is solved applying a SOR algorithm.

Fig. 10 – Fig. 11 show some grids achieved with variational methods.

5 Conclusion

Different grid generation techniques has successfully been applied to process simulation problems.

Algebraic grid generation systems are based on an interpolation. These methods provided fast results, but could not avoid irregular grids (i.e. intersection of grid lines of the same type).

Elliptic methods generate curvilinear coordinate systems by numerical solution of partial differential equations. An iterative control algorithm for the source functions P and Q allows grid adaption and optimization.

With variational formulations many grid properties (e.g. orthogonality of the mapping, smoothness of the grid, etc.) can be controlled at the same time. The variational methods yield high quality grids for most geometries occurring in VLSI fabrication. They are also favoured due to their flexibility and numerical stability.

Acknowledgment

This work is considerably supported by the research laboratories of SIEMENS AG at Munich, Germany, and DIGITAL EQUIPMENT CORPORATION at Hudson, U.S.A.

References

[1] J.F. Thompson, A.U.A. Warsi, and C.W. Mastin, *Numerical Grid Generation, Foundations and Applications*, North-Holland, New York, 1985.

[2] J.M. Ortega, W.C. Rheinboldt, *Iterative Solution of Nonlinear Equations in Several Variables*, Academic Press, San Diego, 1970.

[3] A. Hilgenstock, et al., *Elliptic Generation of Three Dimensional Grids for Internal Flow Calculations*, Numerical Grid Generation in Computational Fluid Dynamics (Ed: J. Häuser, C. Taylor), Pineridge Press, Swansea U.K., 1986

[4] J.F. Thompson, F.C. Thames, and C.W. Mastin, *Automatic Numerical Generation of Body-Fitted Curvilinear Coordinate System for Field Containing Any Number of Arbitrary Two-Dimensional Bodies*, J. Comp. Phys. **15**, 299–319, 1974

[5] J.E. Castillo, S. Steinberg, and P.J. Roache, *On Folding of Numerically Generated Grids: Use of a Reference Grid*, Comm. Appl. Num. Meth. **4**, 471–481, 1988

Figures

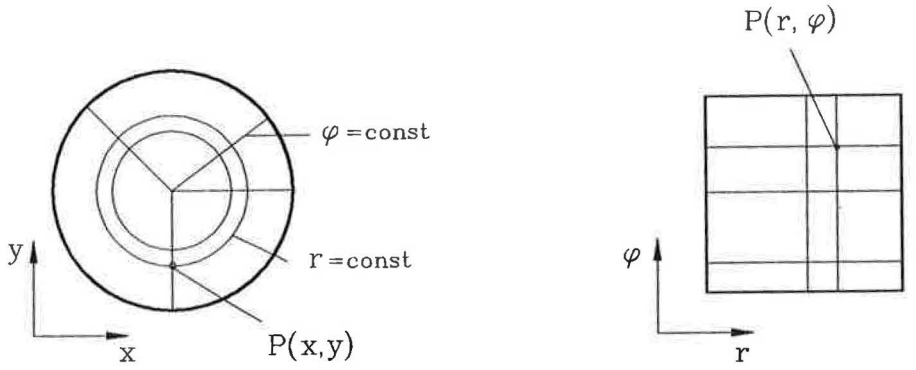

Figure 1: Polar Coordinates

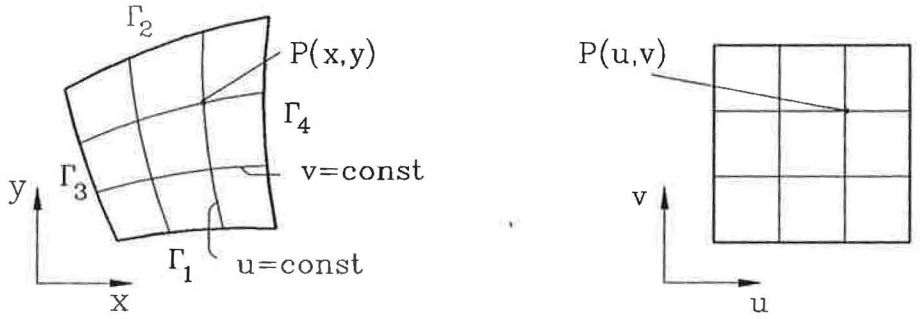

Figure 2: Transformation from physical domain to computational domain

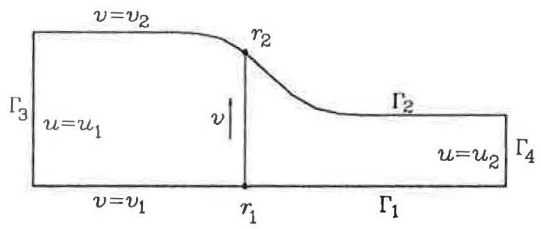

Figure 3: Unidirectional algebraic interpolation

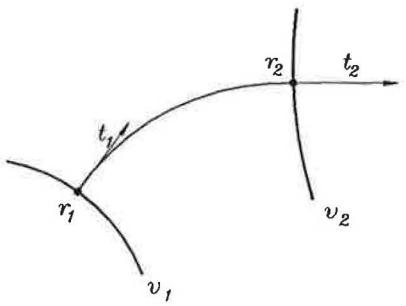

Figure 4: Hermitean interpolation

Figures

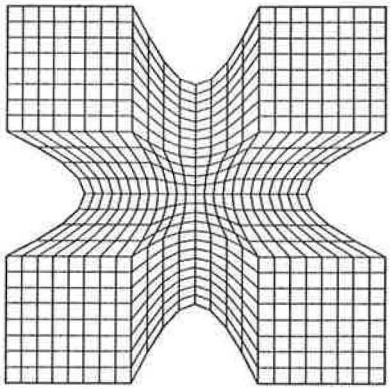

Figure 5: Transfinite interpolation for a test geometry

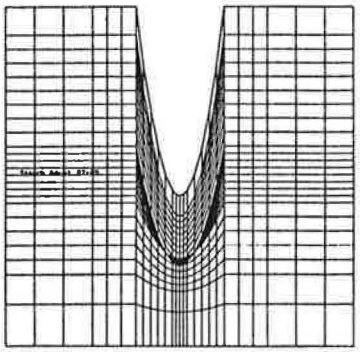

Figure 6: Irregular grid

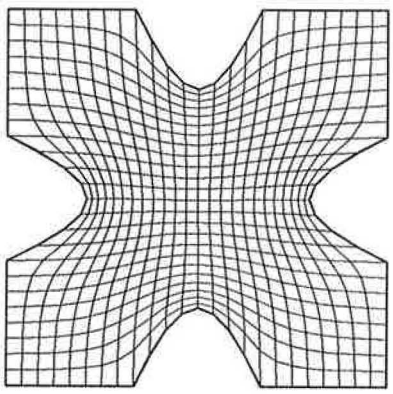

Figure 7: Laplace type grid for a test geometry

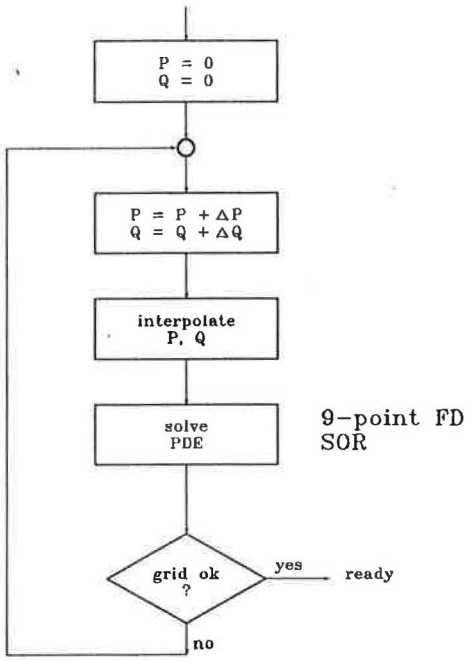

Figure 8: Flowchart for source term (P, Q) control

Figures

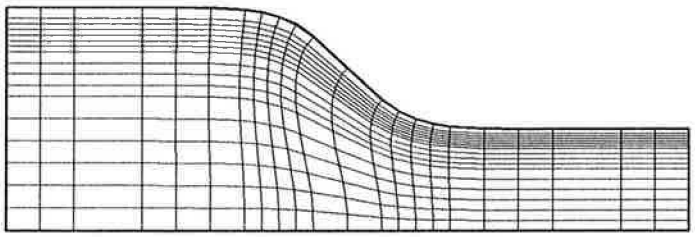

Figure 9: Poisson type grid for diffusion simulation during local oxidation

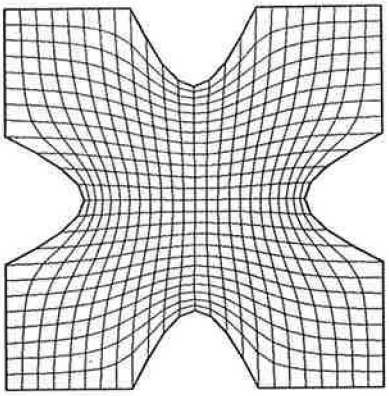

Figure 10: Grid generated with variational method ($w_s = 1, \quad w_a = w_o = 0$)

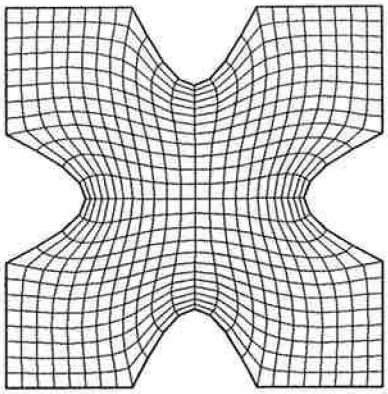

Figure 11: Grid generated with variational method
($w_s = 0.3,\ w_a = 0.7,\ w_o = 0$)