GENERATION OF BLOCK STRUCTURED SMOOTH GRIDS*

Yu. V. Likhanova, V. D. Liseikin, D. V. Patrakhin, I. A. Vaseva
Institute of Computational Technologies SB RAS, Novosibirsk, Russia
e-mail: ula@gorodok.net, lvd@ict.nsc.ru

Рассматриваются два подхода для построения гладких многоблочных сеток. Представлены результаты численных экспериментов для двумерных областей и поверхностей.

Introduction

Many physical phenomena involve the rapid formation, propagation, and disintegration of small-scale structures such as shock waves in compressible flows, shear layers in laminar and turbulent flows, phase boundaries in nonequilibrim and boundary and interior layers, etc. One of the promising tools able to overcome the numerical problems related to these structures is adaptive grid generation technology.

With increasing complexity of the physical problem, there is an increased need for more reliable, robust, and fully automated grid generation codes which enable one to generate suitable meshes in a uniform "block box" mode without or with a only a slight human interaction [1]. The development of such grid systems is one of the challenging problems of modern computational physics and applied mathematics. A successful solution of this problem relies on formulation of well-posed and robust mathematical models for generating in a unified manner local single-block grids in domain and surface patches and the development of techniques for providing smooth matching of the local grids in the zones of block intersections.

Originally, the efforts aimed at building automated grid systems were undertaken in the 1990s [2] as a response to the national High Performance Computing and Communications Program [3]. The codes developed relied on the numerical solution of differential equations for generating local structured grids and Delone and frontal methods for producing local unstructured meshes [4]. As a basic tool for building local structured grids, those codes incorporated Poisson equations

$$\sum_{j=1}^{n} \frac{\partial^2 \xi^i}{\partial s^j \partial s^j} = P^i(\mathbf{s}), \quad i = 1, \ldots, n, \tag{1}$$

in which $\xi^i,\ i = 1, \ldots, n$, are the transformed computational variables, $s^i,\ i = 1, \ldots, n$, are

*The research described in this publication was made possible in part by Award No 06-01-08009 of the Russian Foundation for Basic Research (RFBR) and by Award № 1.8 for the Integration Projects of the Siberian Branch of the Russian Academy of Sciences — 2006.

parametric (physical) variables, while $P^i(\mathbf{s}),\ i = 1, \ldots, n$, are control functions. However, the equations (1) have at least three serious drawbacks [1]:

(1) uncertainties in the specification of the control functions to provide efficient adaptation in several intersecting directions simultaneously;

(2) since the Poisson equations do not have a divergent form the cells of the local grids produced by the inverted Poisson equations may be folded;

(3) poor opportunities to provide effective automation of the grid generation process.

A more promising way lies, apparently, in creating new methods that are free from or suffer in less degree from the disadvantages pertinent to the technique based on the Poisson system. In this respect, considerable progress has been achieved by the development of the methods which apply transformations that are inverse to harmonic functions. The harmonic functions are solutions of the system of Beltrami equations in monitor metric:

$$\sum_{j,k=1}^{n} \frac{\partial}{\partial s^j}\left(\sqrt{g^{\mathbf{s}}} g_{\mathbf{s}}^{jk} \frac{\partial \xi^i}{\partial s^k}\right) = 0, \quad i = 1, \ldots, n, \tag{2}$$

where $g_{\mathbf{s}}^{ij}$ are the local contravariant components of the monitor metric in the parametric coordinates $s^1, \ldots, s^n$, over the physical geometry S^{xn}, $g^{\mathbf{s}} = 1/\det(g_{\mathbf{s}}^{ij})$. These equations allow one to generate both structured and unstructured local grids in domains or on surfaces in a unified manner, regardless of their dimension and geometry features. In particular, these systems can be applied to produce grids in spatial blocks by means of the successive generation of grids on curvilinear edges, faces, and parallelepipeds, using the solution at a step $i < n$ as the Dirichlet boundary condition for the following step $i + 1 \le n$. Thus both the interior and the boundary grid points of a domain or surface can be calculated by the similar elliptic solver. In the same way there are readily realized by the monitor metrics the grid properties required in practical applications such as nonsingularity, adaptivity, and vector-field alignment [5, 6].

The Beltrami equations (2) are invariant with respect to a choice of a coordinate system in the physical geometry S^{xn} which is a guarantee that the grid generated is the same for different parametrizations of S^{xn}. Note the grid obtained by the solution of Poisson equations (1) may vary for different parametrizations of S^{xn} [1].

The Beltrami equations (1) are equivalent to the Euler-Lagrange equations for the functional of energy

$$I[\boldsymbol{\xi}] = \frac{1}{2} \int\limits_{S^n} \sqrt{g^{\mathbf{s}}} \sum_{i,k,l=1}^{n} g_{\mathbf{s}}^{kl} \frac{\partial \xi^i}{\partial s^k} \frac{\partial \xi^i}{\partial s^l} d\mathbf{s}. \tag{3}$$

The numerical minimization of the energy functional rewritten in the coordinates $\xi^1, \ldots, \xi^n$ of the logical domain Ξ^n with respect to the metric of a monitor surface proposed in [7] was used for generating adaptive grids [8, 9] and the solution of two-dimensional fluid dynamics problems on these grids. The energy functional (3) was also applied as a regular component of a more general functional formulated in [10] and [11] for generating adaptive grids. Such energy functionals are also used in the collection and databasing of brain maps for generating conformal mappings between a brain surfaces and a canonical space in order to automatically analyze and compare brain data [12]. There are numerous applications of these mapping algorithms, such as a providing a canonical space for automated feature identification, brain to brain registration, brain structure segmentation, brain surface denoising, and convenient surface visualization.

The expression of functional (3) is also applied to finding the monitor metrics providing the generation of the numerical grids with required properties. For this purpose the monitor

metric is to have such a form so that the integrand in (3) describes a measure of departure of the grid from the necessary grid at points $\mathbf{s}$ in the parametric domain S^n. If such a metric is found then it can be expected that the minimization of functional (3) will produce the grid with the required property. In particular, such metrics have already been found for generating grids with node clustering in the zones of large values of functions [13] or large gradients of functions [7], as well aw field-aligned [14] and balanced grids [15].

The requirement of mutual positioning or "communication" of local grids in the vicinity of adjacent grid blocks and fictive edges or faces have a considerable influence on the efficiency of the numerical calculations. The grid lines of two adjacent blocks, in general, need not have points in common, and can join smoothly or nonsmoothly. If grid lines in adjacent blocks join smoothly, interpolation is not required. If the lines do not join, then during the calculation the solution values at the nodes of one block must be transferred to those of the adjacent block in the neighborhood of their intersection. This is done by interpolation or (in mechanics) using conservation laws.

The types of grid line interaction between adjacent blocks and in the vicinity of fictive edges or faces are selected on the basis of the features of the physical quantities in the region of their intersection. If the gradient of the physical solution is not high in these zones and interpolation can, therefore, be performed with high accuracy, the grid lines do not need to join. This greatly simplifies the algorithm for constructing the grid. If there are high gradients of the solution near the intersection of two blocks or in the neighborhood of fictive edges or faces, a smooth matching is usually performed between the grid lines.

The problem of smooth matching was typically overcome by an algebraic technique using Hermitian interpolation [1], or by elliptic methods, involving a choice of control functions [16]. A combination of Laplace and Poisson equations, yielding equations of fourth or even sixth order, was also used for this purpose [17]. A distinctive feature of these approaches is that the boundaries of joint blocks or fictive intersections as well as grid distribution on these geometries remain fixed, while smooth grid matching is provided by control functions or by high order grid generation equations.

In this paper we describe two new techniques for producing smooth block structured grids. The essential difference from the previous methods for generating smooth block-structured grids is that the current approaches are based on the computation of both the position of the joint boundary segments of the blocks or fictive segments and grid distribution at these segments. The advantage of these approaches is that the control functions can be applied to providing basic grid properties only and be exempted for providing the smooth matching of local grids.

1. General Block Concept

In the commonly applied block strategy, the physical geometry is divided into a few contiguous subgeometries referred to as blocks, which may be considered as the cells of a coarse, generally unstructured grid (see fig. 1 (*right-hand*) for the tokamak edge domain). And then an individual structured or unstructured mesh is generated in each block. The union of these local grids constitutes a mesh referred to as a multi-block grid. The main reasons for using multi-block grids rather than single-block grids are that:

(1) the physical geometry may be exceedingly complicated, having a multiply connected boundary, cuts, narrow protuberances, cavities, etc.;

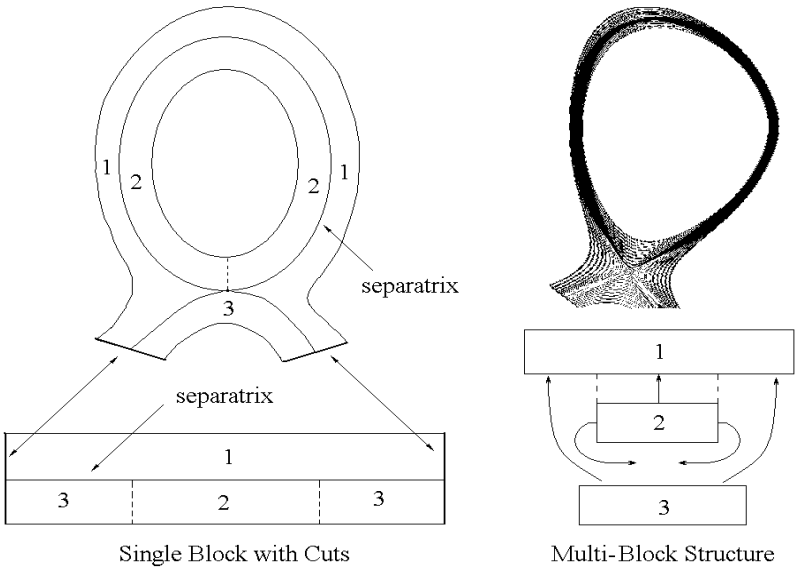

Fig. 1.

(2) the physical problem may be heterogeneous relative to some of the physical quantities, so that different mathematical models are required in different zones of the geometry to adequately describe the physical phenomena;

(3) the solution of the problem may behave non-uniformly: zones of smooth and rapid variation of different scales may exist;

(4) opportunity to apply parallel algorithms.

If meshes in all blocks are structured then the multi-block grid can be considered as locally structured at the level of an individual block, but globally unstructured when viewed as a collection of blocks. Such grids are called block-structured grids. Thus a common idea with a block-structured grid technique is the use of different structured grids, or coordinate systems, in different zones of the physical geometry, allowing the most appropriate grid configuration to be used in each zone.

Block-structured grids are considerably more flexible in handling complex geometries than structured grids. Since these grids retain the simple regular connectivity pattern of a structured mesh on a local level, these block-structured grids maintain, in nearly the same manner as structured grids, compatibility with efficient finite-difference, finite-volume, or spectral element algorithms used to solve partial differential equations. However, the generation of block-structured grids may take a fair amount of user interaction and, therefore, requires the implementation of an automation technique to lay out the block topology.

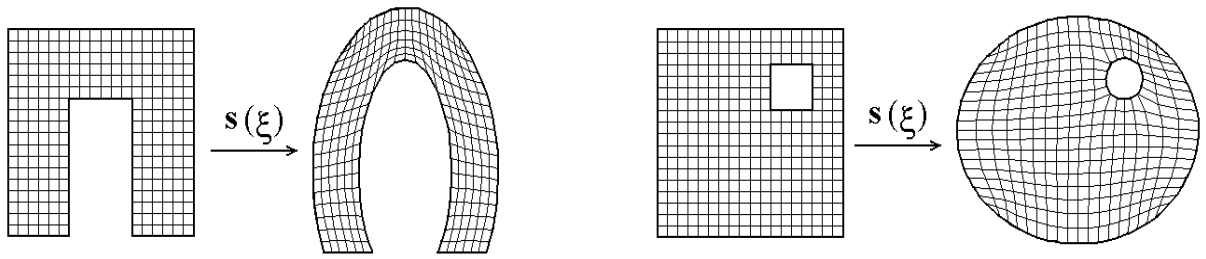

Fig. 2. Computational domains adjusted to the physical domains.

1.1. Topologies of Computational Domains

The correct choice of the topology in a block, depending on the geometry of the logical domain and the qualitative type of the transformation of the region onto the block, has a considerable influence on the quality of the grid. There are two ways of specifying the computational domain for a block:

(1) as a complicated polyhedron which maintains the schematic form of the block subdomain (fig. 1 (*right-hand*));

(2) simply as a solid cube or a cube with cuts (fig. 1 (*left-hand*)).

With the first approach, the problem of constructing the grid transformation is simplified, and this method is often used to generate a single-blocked grid in a complicated domain. The second approach relies on a simplified geometry of the computational domain but requires sophisticated methods to derive suitable grid transformations.

1.2. Basic Topologies of the Local Grids

In a block which is homeomorphic to a cylinder with thick walls, the grid topology is determined by the topology of the two-dimensional grids in the transverse sections. In applications, for sections of this kind, which are annular planes or surfaces with a hole, wide use is made of three basic grid topologies: H (fig. 3), O(fig. 4) and C (fig. 5).

In H-type grids, the computational domain has an interior cut which is opened by the construction of the coordinate transformation and mapped onto an interior boundary of the block. The outer boundary of the logical domain is mapped onto the exterior of the block. The interior boundary has two points with singularities where one grid line splits. H-type grids are used, for instance, when calculating the flow past thin bodies (aircraft wings, turbine blades, etc.).

In O-type grids, the computational domain is a solid domain. In this case the grid lines in the block with a hole are obtained by bending the logical domain, sticking two opposite sides

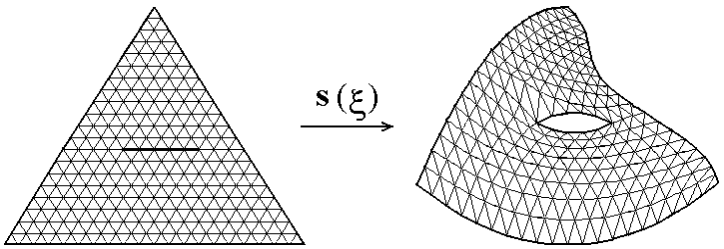

Fig. 3. H-type grid.

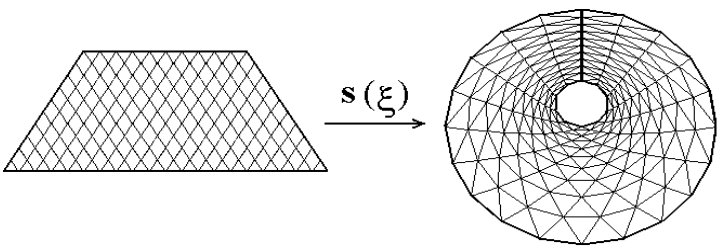

Fig. 4. O-type grid.

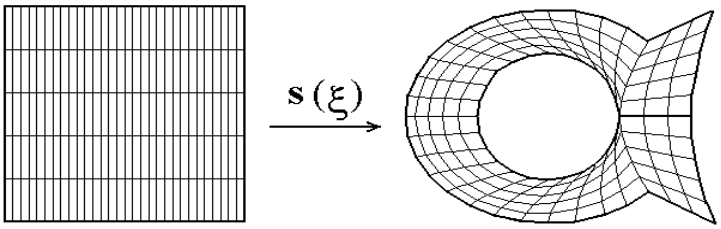

Fig. 5. C-type grid.

together and then deforming. The stuck sides determine the cut, called the fictive edge, in the block. An example of O-type grid is the nodes and cells of a polar system of coordinates. The O-type grid can be constructed without singularities when the boundary of the block is smooth. Grids of this kind are used when calculating the flow past bulky aircraft components (fuselages, gondolas, etc.) and, in combination with H-type grids, for multilayered block structures.

The computational domain is also a solid domain in a C-type grid, but the mapping onto the block with a hole involves the identification of some segments of one of its sides and then deforming it. In the C-type grid, the grid lines of one family leave the outer boundary of the block, circle the inner boundary and return again to the outer boundary. There is one point on the inner boundary which has the same type of singularity as in the H-type grid. The C-type grids are commonly used in regions with holes and long protuberances.

The O- and C-type techniques in fact introduce artificial interior cuts in multiply connected physical geometries to generate single block-structured grids. The cuts are used to join the disconnected components of the boundary of the geometry in order to reduce their number. Theoretically, this operation can allow one to generate a single coordinate transformation in a multiply connected physical geometry.

The choice of the grid topology in a block depends on the structure of the solution, the shape of the physical geometry, and, in the case of continuous or smooth grid-line communication, on the topology of the grid in the adjacent block as well. For complicated physical geometries, such as those near aircraft surfaces or turbines with a large number of blades, it is difficult to choose the grid topology of the blocks, because each component of the system (wing, fuselage, etc.) has its own natural type of grid topology, but these topologies are usually incompatible with each other.

2. Approaches to Smoothing Grids

In this section we describe two novel iterative approaches for providing smooth matching of grid lines across adjacent blocks and for generating smooth grids in the vicinity of fictive edges of faces as in the case of O- or C-types of meshes. The essential difference from the previous methods for generating smooth block-structured grids is that the current approaches are based on the computation of both the position of the joint boundary segments of the blocks and grid distribution at these segments.

The local grid in each block is found by the numerical solution of the Dirichlet problem for inverted Beltrami equations in monitor metrics [18].

The position of the segments of the adjacent blocks and grid distribution in this segments are found: 1) through the numerical solution of grid equations and 2) by the interpolation from the nodes of the grid hypersurfaces neighboring the joint boundary segment.

2.1. Computation Through Grid Equations

The idea of this version of the approach is demonstrated in fig. 6 representing a two-block structured scheme for generating smooth quadrilateral grids in a domain X^2. The left-hand block of the domain is bounded by the curves A_1B_1, B_1C_1, C_1D_1, and D_1A_1. Similarly the right-hand block is bounded by the segments A_2B_2, B_2C_2, C_2D_2, and D_2A_2. The curves C_1D_1 and B_2A_2 are identical presenting the joint boundary of the blocks. By the iterations the grid in the domain X^2 is computed independently at each block via solving numerically the grid equations (2) at the nodes of the corresponding logical domains Ξ_1^2 and Ξ_2^2 with the identical boundary node distribution at joint segments in X^2. These segments and their grid points are found in the process of iterations by solving the boundary value problem at the points of a new logical domain $ABCD$.

Thus, during the first iteration we specify the joint boundary segment $A_2B_2 = D_1C_1$ in X^2, the grid nodes at this segment, and the transformation at this nodes from the grid points of the segments C_1D_1 and A_2B_2 of the corresponding logical domains Ξ_1^2 and Ξ_2^2. Then we compute independently the grid nodes in the both blocks of X^2 through the grid equations. Having done this we choose in the both blocks the corresponding grid lines AB and CD, for example, neighboring the joint boundary line $C_1D_1 = B_2A_2$. After this we solve numerically the boundary value problem for the grid equations at the points of a new logical domain $ABCD$. The number of the points at the segments AD in the domain X^2 and in this new logical domain coincides. The boundary points at the segments AB and CD in the logical domain are mapped at the computed points of the corresponding segments AB and CD in X^2. The transformation of the boundary points at the segments AD and BC coincides with the initial boundary transformation from the corresponding points of the segments A_1D_1, A_2B_2 and B_1C_1, B_2C_2. In particular, the points E and F are mapped at the points C_1 and D_1, respectively. By the computation the points of the segment EF in the new logical domain are transformed at the points of a new joint boundary segment (dotted line) of two new blocks. Then the iterations continue up to satisfy a tolerance condition. Similarly, there are generated smooth block-structured triangular grids.

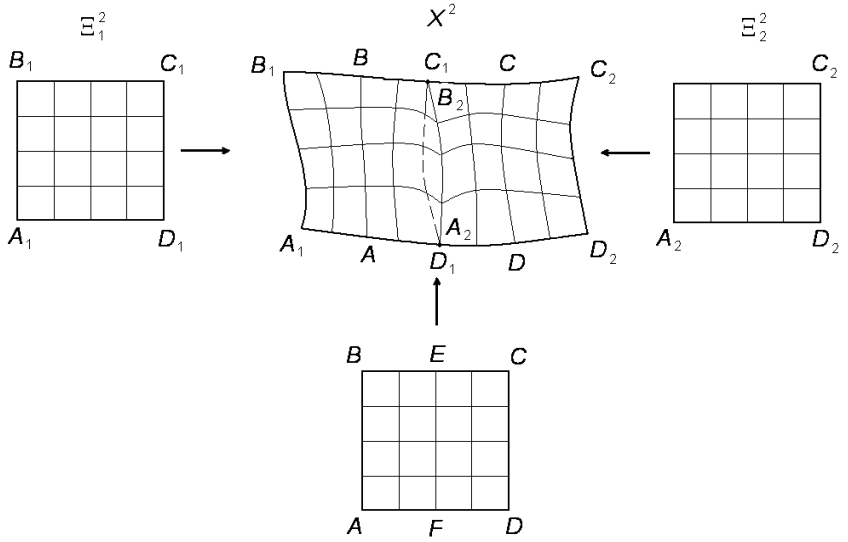

Fig. 6. Scheme for generating smooth grids by computing the common points of the adjacent blocks through grid equations.

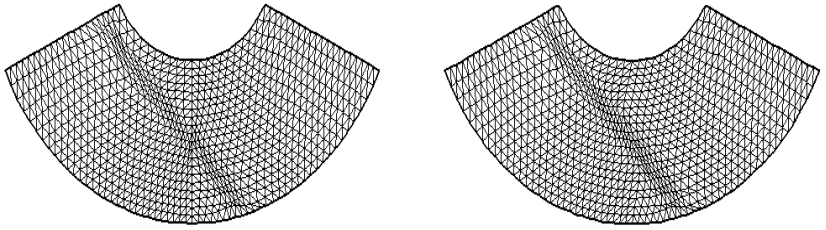

Fig. 7.

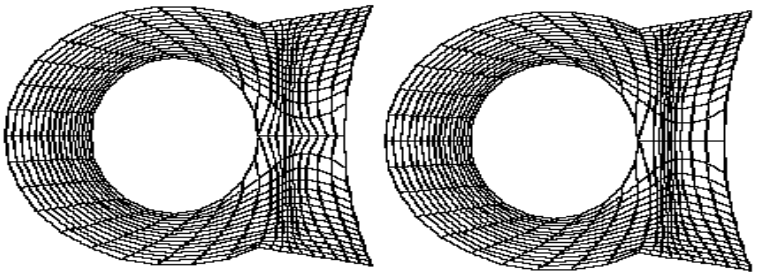

Fig. 8.

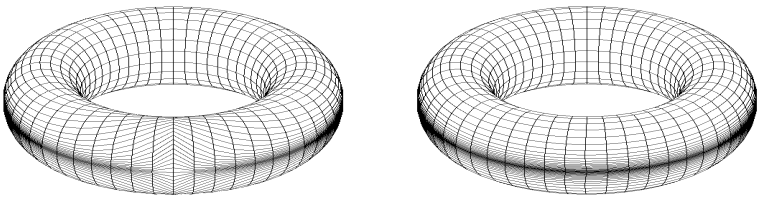

Fig. 9.

Figures 7 and 8 demonstrate adapted to the values of a function nonsmooth grids (left-hand) computed in two-dimensional domains without the use of the smoothing algorithm and smooth grids (right-hand) found by the application of the algorithm.

Analogous procedure is formulated for generating smooth block-structured surface and three-dimensional domain grids. Figure 9 exhibits nonsmooth (left-hand) and smooth (right-hand) grids on a surface.

An essential feature of this approach is that the equations for generating grids in the blocks and for computing new joint boundary segments of the blocks are the same. So such a smoothing process does not breach the properties of the grids realized by particular monitor metrics.

2.2. Computation by Interpolation

In this approach the grid point at the i-th level of a new joint boundary segment (dotted line in fig. 10) is computed by the following procedure: after computing the grid points at both blocks having a joint boundary line with a grid node D we choose at the i-th level of grid lines two points neighboring the previous joint segment point D of one block (the points B and C in Fig. 10) and one neighboring a point of another block (point F in fig. 10). Through these points we draw a smooth line, for example, a circle segment.

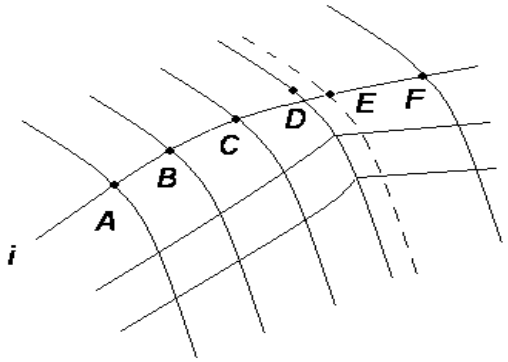

Fig. 10.

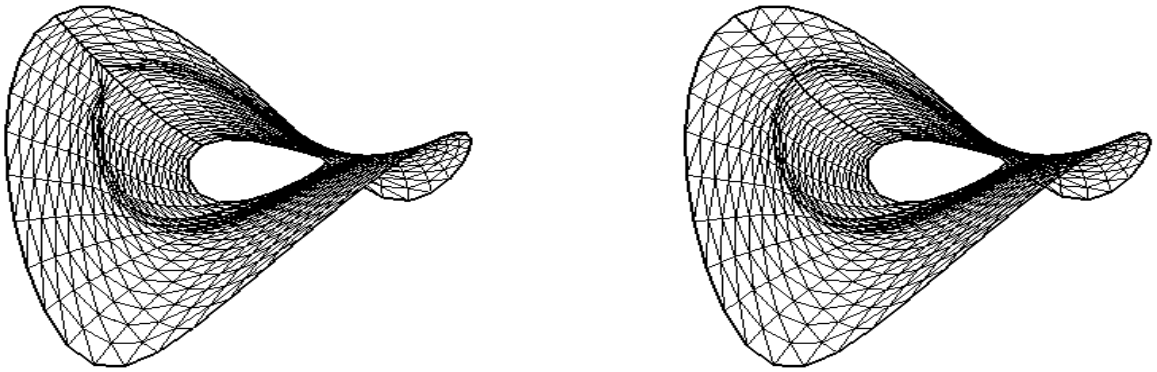

Fig. 11.

The i-th level of a new joint boundary point E will lie on this segment. Its position at the segment is defined from the relation $\overline{CE}/\overline{EF} = \overline{AB}/\overline{BC}$, where A is the third grid point of the first block computed at the i-th level (overline means the distance between the corresponding points). When all grid points of the new joint boundary segment are found by such way then the grid points in the blocks are found independently by solving grid equations. The process continues up to satisfy a required tolerance.

Figure 11 illustrates an O-type nonsmooth (left-hand) and smooth (right-hand) surface grid generated by this approach.

Similarly there are generated smooth block-structured three-dimensional domain grids.

References

[1] Liseikin V.D. Grid generation methods. Berlin: Springer-Verl., 1999.

[2] Thompson J.F. A reflection on grid generation in the 90s: trends, needs influences // Numerical Grid Generation in CFD. Mississippi State Univ. 1996. Vol. 1. P. 1029–1110.

[3] Raveche H.J., Lawrie D.H., Despain A.M. A National Computing Initiative. SIAM. Philadelphia, 1987.

[4] Thompson J.F., Warsi Z.U.A., Mastin C.W. Numerical Grid Generation. Foundations and Applications. N. Y.: Elsevier Sci. Publ., 1985.

[5] Liseikin V.D. Layer resolving grids and transformations for singular perturbation problems. Utrecht: VSP, 2001.

[6] Methods of Riemahnnian Geometry for Generating Numerical Grids / Yu.I. Shokin, V.D. Liseikin, A.S. Lebedev et al. Novosibirsk: Nauka, 2005.

[7] Liseikin V.D. On generation of regular grids on n-dimensional surfaces // USSR Comput. Math. Phys. 1991. Vol. 31, N 11. P. 41–57.

[8] Azarenok B.N., Tang T. Second-order Godunov-type scheme for reactive flow calculations on moving meshes // J. Comput. Phys. 2005. Vol. 206, N 1. P. 48–80.

[9] Charakch'an A.A., Ivanenko S.A. A variational form of the Winslow grid generator // J. Comput. Phys. 1997. Vol. 136. N 2. P. 385–398.

[10] Huang W. Variational mesh adaptation: isotropy and equidistribution // J. Comput. Phys. 2001. Vol. 174. P. 903–924.

[11] Huang W., Ren Y., Russel R.D. Moving mesh PDEs based on the equidistribution principle // SIAM J. Numer. Anal. 1994. Vol. 31. P. 709–730.

[12] Gu X., Yau S. Computing conformal structures of surfaces // Communicatios in Information and Systems. 2002. Vol. 2, N 2. P. 121–146.

[13] Danaev N.T., Liseikin V.D., Yanenko N.N. Numerical solution on a moving curvilineer grid of viscouse heat-conducting flow about a body of revolution // Chisl. Metody Mekhan. Sploshnoi Sredy. 1980. Vol. 11, N 1. P. 51–61.

[14] Liseikin V.D. On a universal monitor metrics for generating numerical grids // Doklady Mathematics. 2005. Vol. 400, N 1. P. 1–5.

[15] Glasser A.H., Liseikin V.D., Kitaeva I.A. Specification of monitor metrics for generating vector field-aligned numerical grids // Russ. J. Numer. Anal. Math. Modelling. 2005. Vol. 20, N 5. P. 439–461.

[16] Thomas P.D., Middlecoff J.F. Direct control of the grid points distribution in meshes generated by elliptic equations // AIIA J. 1980. Vol. 18, N 6. P. 652–656.

[17] Bell J.B., Shubin G.R., Stephens A.B. A segmentation approach to grid generation using biharmonics // J. Comput. Phys. 1982. Vol. 47, N 3. P. 463–472.

[18] Liseikin V.D. A computational differential geometry approach to grid generation. Berlin: Springer-Verl., 2004.

Received for publication Yanuary 19, 2006