Three-dimensional Mesh-generator for Finite Element Method Applications

P.G. Akishin, A.A. Sapozhnikov
Laboratory of Information Technologies, JINR

Introduction

The finite element method (FEM) [1] is one of the most effective numeric methods for solving linear and non-linear multi-dimensional scientific and technical problems. It allows modeling of systems with a complex geometry and an irregular physical structure. One of the most time-consuming steps when solving three-dimensional problems using the FEM method is building adaptive mesh. The mesh should satisfy FEM requirements, allow one to represent a complex geometry of unknown parameter areas and describe the medium interface in detail. The creation of an automated generator of the spatial mesh with a user friendly interface which is suitable for defining the calculation geometry, having visualization of incoming data and mesh quality control is a very complex and important problem [2-5]. Such type generator is a critical constituent of any large scale specialized software complex based on FEM. One of the most outstanding representatives of such generators is a mesh generator in OPERA pre-processors by Vector Fields vendor [6]. A similar automated generator of spatial mesh based on MAPPING approach [7] is considered in this article. An initial two-dimensional mesh is generated on the base plane just like this is done in OPERA. Then the mesh is extended into space keeping the topological structure. In this generator in order to improve the quality of the mesh, a library of standard fragmentations is used for triangle macro elements fragmentation on the base plane. A graphical interface for defining incoming geometry is created. Visual control of the quality of the mesh generation is also available.

Approach

There are specific requirements for two- and three-dimensional mesh which is used in FEM. The mathematical problem is formulated as follows: let's assume there is a simply connected domain $\Omega \subset R^3$. There is a polyhedron $\tilde{\Omega}$ which approximates Ω. We need to present $\tilde{\Omega}$ as an aggregation of a finite number of N_0 polyhedrons $S_i(\breve{\Omega} = \bigcup_{i=1}^{N_0} S_i)$, which meet the following conditions:

1. $\mu(S_i \cap S_j) = 0$, if i $\neq$ j, i.e. the measure of different polyhedrons intersection is equal to zero.

2. The vertex of one polyhedron cannot be inside the points of facet or inside the edges of the other polyhedron, i.e. if two polyhedrons intersect they intersect either by a whole edge or by a whole facet or by just one vertex.

In general, if isoperimetric elements are used, their pre-images satisfy these requirements. In order to build a three-dimensional mesh, we use the approach in which primary two-dimensional mesh is built on the base hyperplane. After that the two-dimensional

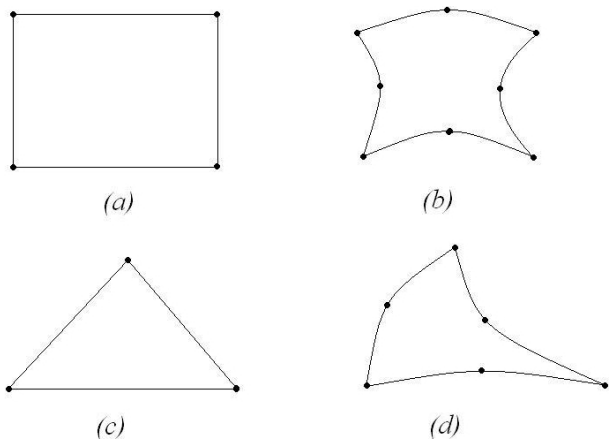

Figure 1: Fundamental two-dimensional elements

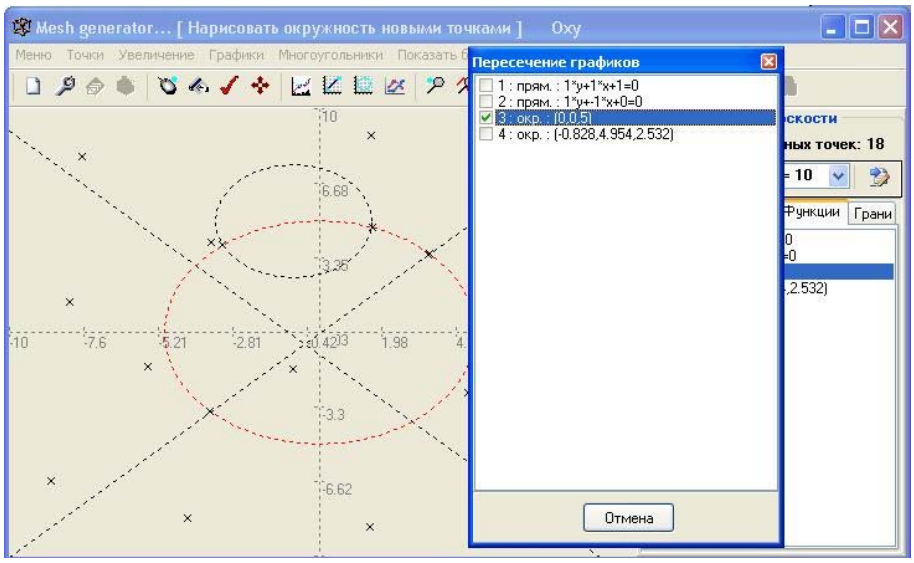

Figure 2: Mesh generator

mesh is projected from the base hyperplane through space onto the next hyperplane keeping the topology, then it is projected to the next hyperplane and so on.

Triangle and quadrangle elements or their topological analogues on the base plane are used as fundamental two-dimensional elements in this generator (Figure 1). This generates hexahedrons and triangular prisms or their isoperimetric analogues in the three-dimensional space.

Generation of three-dimensional mesh

There are three major steps during the mesh generation. The first one is the generation of a base hyperplane. The second one is the extension of the built two- dimensional mesh into the following hyperplanes. The third one is a fragmentation of the initial three-dimensional mesh to meet the user defined requirements.

Working with the program starts with defining the base hyperplane. All the following actions on the first step will be done on this plane. There are 3 ways to define the input geometry. These are defining the points in text mode, defining the points in interactive

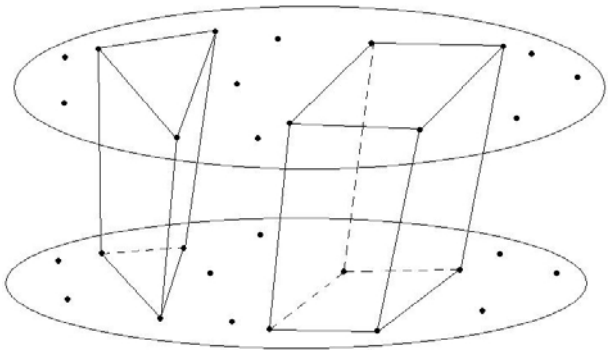

Figure 3: Creation of three-dimensional macro elements

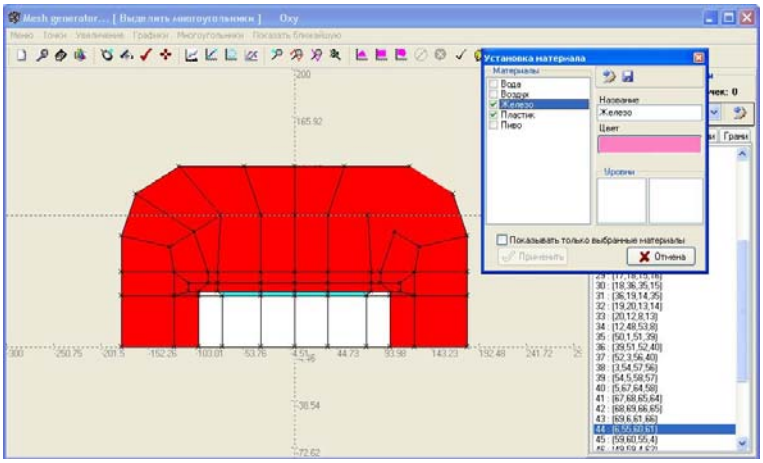

Figure 4: Properties representation

mode using mouse pointer and defining the points as crossings of curves which have been built. The program has a feature for drawing functional graphs (straight lines and circles) both using equations and point by point definition.

The Figure 2 represents a generator panel which is the incoming interface for work with the program on this stage. There are several ways to change points coordinates - changing coordinates in a text mode, moving the point using a mouse pointer, projection of a group of points onto the graph, parametrical change of coordinates for a group of points, rotation of a group of points around a fixed point at a defined angle. Once the points are defined, they are grouped into two-dimensional elements (Figure 1) using the pointer.

The next step in the mesh generation is the creation of three-dimensional macro elements. Points from the base hyperplane are extended into space keeping the topology of the already built two-dimensional elements until they reach the next hyperplane (Figure 3). Then they are extended to reach the next hyperplane and so on. Points coordinates can be changed not only within each hyper plane but also along the plane normal.

Once the second step of mesh generation is finished, all created macro elements can be broken down into groups in accordance with their properties (i.e. various material type filling in the elements, the difference in equations, the difference in coefficients, etc.) You can mark macro elements with various colors according to their properties (Figure 4).

You can visually control the coarse mesh which is built, observation point as well as

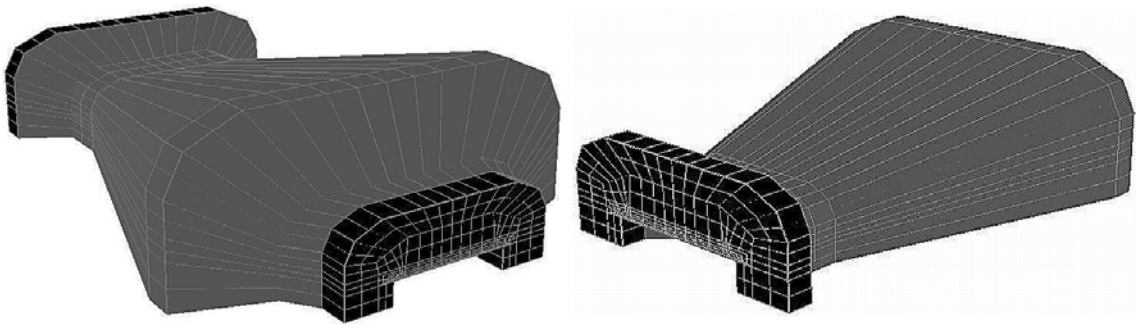

Figure 5: Change observation point and angle

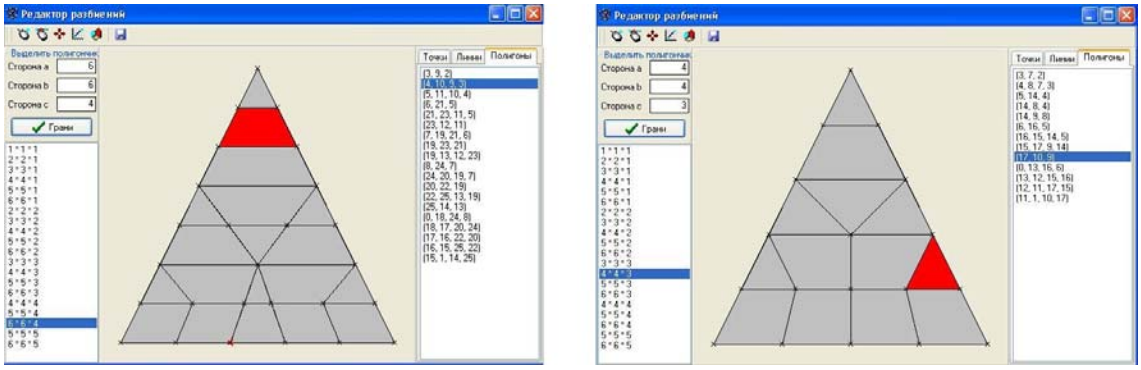

Figure 6: Fragmentations for triangles

observation angle can be changed (Figure 5).

On the last step of generation a fragmentation of the coarse mesh which has been built, is performed in accordance with predefined requirements. The user defines a fragmentation number for each edge of every two-dimensional element of the base hyperplane. The fragmentation number will be the same for all topologically equal edges on the other hyperplanes, which were created via translation in the space of the base hyperplane. It should be noted that there are certain limitations in this version of the generator in terms of edges fragmentation. In particular, for all opposite edges of two-dimensional quadrangles the same fragmentation number should be defined. The user should also define the number of layers into which space is divided by any two hyperplanes. A library of standard fragmentations is used to improve the quality of the fragmentation. Examples of various standard fragmentations for triangles are given in Figure 6.

Both triangles and quadrangles are used for fragmentation. Fragmentation on the base hyperplane is represented on Figure 7.

An example of using the automated generator for building three-dimensional finite element model of the dipole magnet is given. Figure 8 represents a three-dimensional view of a half a dipole magnet. You can rotate and move images in space.

Conclusion

This article gives a description of the automatic generator of spatial mesh, which satisfies the requirements defined for finite elements method fragmentations. Such type generators are normally included as a module of large scale expensive commercial software, which is based on finite elements formulation of the solving problems. The described

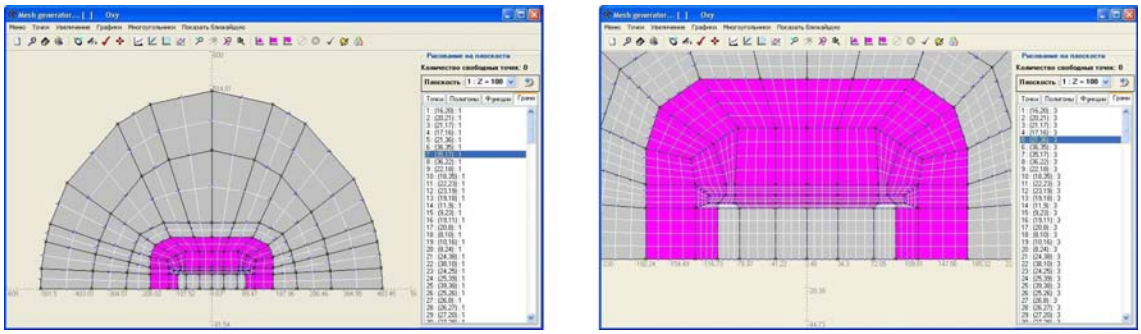

Figure 7: Fragmentation on the base hyperplane for the dipole magnet

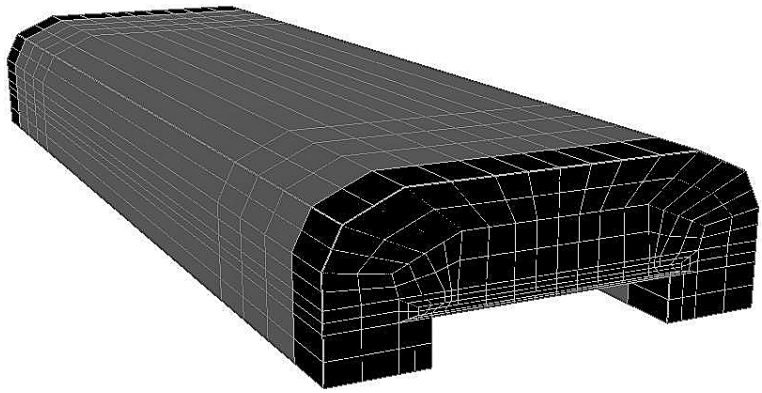

Figure 8: Three-dimensional view of a half a dipole magnet

mesh generation method is similar to the existing ones. However, certain distinctions should be outlined. In particular algorithms for fragmentation of macro blocks into micro blocks are improved via creating a database of standard fragmentations. A user-friendly interface for defining input geometry is created. Visual control on all stages of generation is available. In particular the quality of the final fragmentation can be visually controlled. The proposed generator can be used as a preprocessor for solving a large spectrum of problems which are based on the finite elements method.

References

[1] O.C.Zienkiewicz. "The finite element method in engineering science". MCgraw-Hill, London, 1971.

[2] A Survey of Unstructured Mesh Generation Technology. Steven J. Owen Department of Civil and Environmental Engineering, Carngie Mellon University, Pittsburgh, PA. and ANSYS Inc., Canonsburg, PA.

[3] T.D.Blacker, J.L.Mitchiner, L.R.Phillips and Y.T.Lin. "Knowledge System Approach to Automated Two-Dimensional Quadrilateral Mesh Generation". Computers in Engineering, 1988.

[4] George P.L. "Automatic Mesh Generation: Application to Finite Element Methods", 1991.

[5] M.C.Lee and M.S.Joun. "General Approach to Automatic Generation of Quadrilaterals on Three-Dimensional Surfaces", Communications in Numerical Methods in Engineering, 1998.

[6] Vector Field, OPERA code, http://www.vectorfields.com/.

[7] W.A.Cook and W.R.Oakes, "Mapping Methods for Generating Three- Dimensional Meshes", 1982.