A voxelization modeling algorithm for point-kernel simulations on 3dsMax files

YANG Li-qun[1], LIU Yong-kuo[2], and LI Meng-kun[3]

1. Fundamental Science on Nuclear Safety and Simulation Technology Laboratory, Harbin Engineering University, Harbin 150001, China (hanyangrensheng@163.com)
2. Fundamental Science on Nuclear Safety and Simulation Technology Laboratory, Harbin Engineering University, Harbin 150001, China (lyk08@126.com)
3. Fundamental Science on Nuclear Safety and Simulation Technology Laboratory, Harbin Engineering University, Harbin 150001, China (18845594647@126.com)

Abstract: A voxelization modeling algorithm based on tetrahedral three-dimensional (3D) scan conversion for point-kernel gamma ray calculations is proposed in this paper. The arbitrarily shaped geometries of radiation environment are built by 3dsMax software and the properties of the radiation source and shield material are given directly in the 3DS format file. The voxelization algorithm converts irregular geometry into a compressed voxel model for point-kernel calculation. The algorithm can handle arbitrarily shaped geometries: with or without holes, self-intersecting, manifold or non-manifold. It has fast calculation speed and good shape adaptability.
Keyword: voxelization algorithm; point-kernel simulation; radiation field calculation; 3dsMax

1 Introduction

In order to calculate the gamma ray radiation field, a series of computational tools were developed based on the point-kernel approach, such as PUTZ [1] and QAD [2]. The method of gamma ray calculation has two main steps: implementation of 3D geometries for radiation environment simulation and dose calculation.

The implementation of 3D geometries for radiation environment simulation is a time consuming work. Each computational tool uses textual description and supplies its own scripting language, which makes the creation of geometries a nontrivial undertaking.

This difficulty can be avoided by developing a visual automatic geometric modeling interface or a custom convertor. For example, Borglund *et al.* developed a computer code package that handles geometric descriptions exported from Computer-aided design (CAD) applications for Monte Carlo simulation of particle transport [3]. This package replaces the part of some MC codes that handles geometry information and simplifies the preparation of MC simulations for complex geometric systems with a large number of objects or with complicated shapes. Theis *et al.* proposed a computer code called SimpleGeo, which is a solid modeler that combines Constructive Solid Geometry (CSG) and boundary representations [4]. This hybrid architecture allows for rapid and flexible visualization and creation of solid models for radiation transport codes, but the solid model built is simple.

For existing gamma ray calculation codes, quickly and accurately converting arbitrarily shaped geometries into models that can be used for dose calculations is still a difficult work. A viable method is converting geometries from their continuous geometric representation into a set of voxels that approximates it. For example, Karabassi *et al.* proposed a fast and simple voxelization algorithm based on z-buffer [5]. However, this method can only be applied to a restricted subset of closed solids. Li *et al.* proposed a hardware acceleration voxelization method [6]. The method is based on the technique known as “deep delamination” that uses the stencil buffer to compute a list of voxel layers in each axis direction. But this is a boundary-only voxelization method. Rueda *et al.* proposed a fast CPU-based algorithm for handling a wide variety of polyhedral solids: with or without holes, self-intersecting, manifold or non-manifold [7]. Unlike previous approaches, the algorithm can implement voxelization within a solid.

Received date: November 3, 2018
(Revised data: January 10, 2019)

In order to convert arbitrarily shaped geometries to the models used in point-kernel calculation, a voxelization modeling algorithm based on tetrahedral 3D scan conversion is proposed in this paper. The 3D geometries of radiation environment are built by 3dsMax software and the properties of the radiation source and shield materials are given directly in the 3DS format file. The voxelization algorithm converts irregular geometry into a compressed voxel model for point-kernel calculation. The algorithm can handle complex geometries: with or without holes, self-intersecting, manifold or non-manifold. It has fast calculation speed and good shape adaptability.

The rest of this paper is organized as follows: Section 2 introduces the voxelization modeling algorithm. Section 3 shows the related experiments and results; the paper is concluded in the last section.

2 Methodology

The structure of voxelization algorithm for arbitrarily shaped geometries is shown in Fig.1. The whole process can be divided into six steps.

1. Establish solid models in 3dsMax software.
2. Input the 3DS file in the program and create the minimum bounding box for each solid object.
3. Construct tetrahedrons with origin and triangular faces.
4. Scan each tetrahedron along the *z*-axis to get a series of triangular sections.
5. Voxelize all triangular sections and store the presence value of voxels in the 3D array buffer.
6. Build the voxel model based on the presence value of voxels.

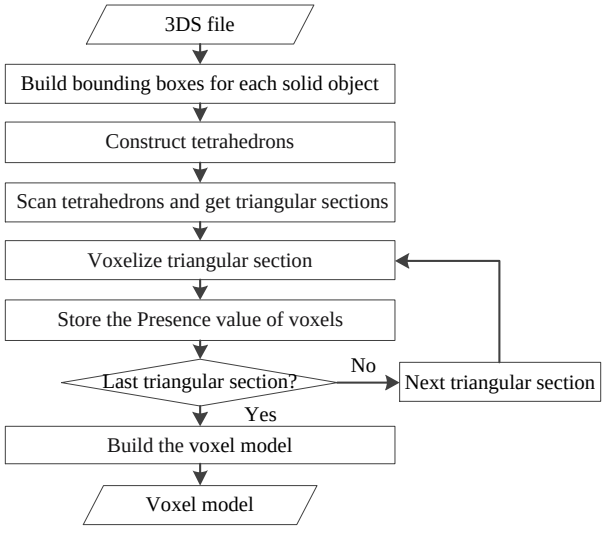


Fig.1 The structure of voxelization modeling algorithm.

2.1 Establish solid model in 3dsMax

The arbitrarily shaped geometries are built visually with exact dimension information of radiation environment in the 3dsMax software.

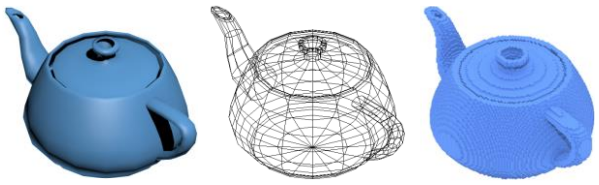
In order to reduce the input process, the solid model is built using a naming format. As shown in Table 1, the keyword for source objects is "SO", and the keyword for shield objects is "SH". The number of the object is written directly behind the keyword. At the same time, the material information of sources and shields are provided to the material ball in the *Material Editor*.

Table 1 3dsMax naming format

Object	Keyword	Object naming format	Material naming format
Source	SO	SO,Source number	Energy,Intensity
Shield	SH	SH,Shield number	Density,Actomic number

The solid model is imported as a 3DS format file. The solid model can be described by a curve mesh, which consists of a series of triangular patches.

2.2 The theoretical basis of voxelization modeling algorithm



(a) (b) (c)

Fig.2 The voxelization of a teapot. (a) Solid model. (b) Curved mesh. (c) Voxel model.

In order to calculate the dose rate of arbitrary geometries, the voxelization modeling algorithm is proposed for the conversion of solid model to voxel model (see Fig.2). The theoretical basis for the solid model voxelization modeling algorithm is the point-in-tetrahedron inclusion test of Feito *et al.* [8]. In this paper, the following definition expresses the inclusion in a more useful way [9]:

Definition. Let G be a solid model and O be the origin of G. Let $T_1, T_2, \cdots, T_n$ be tetrahedrons composed of O and each triangular face of G (See Fig.3). Voxelize all tetrahedrons to get all voxels $V_1, V_2, \cdots, V_n$. The voxel model of G consists of voxels that appear odd times.

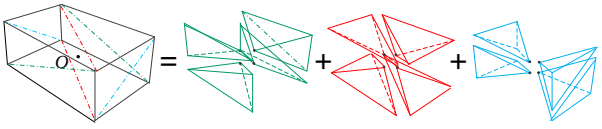


Fig.3 Construct tetrahedrons with origin O and 12 triangular faces of a box solid model.

In order to reduce the computational cost of the program, the center of all the vertices of a solid is chosen as the origin O.

The main idea of the voxelization algorithm is to check whether each voxel belong to the object or not and assign to the voxel a value of odd or even respectively. A voxel with an odd value belongs to the object, while a voxel with an even value does not belong to this object. The definition ensures that the voxelization algorithm can deal with irregular solid models of hollow, perforated, self-intersecting, penetrating and so on.

2.3 Voxelization based on tetrahedral 3D scan conversion

Let $\triangle ABCD$ be an arbitrary tetrahedron of a solid, and w be the width of the voxel. Sort the vertices of the tetrahedron according to the z coordinate. As shown in Fig.4.a, let D (D_x, D_y, D_z) be the vertex with the smallest z coordinate, C (C_x, C_y, C_z) the next, and so on with B (B_x, B_y, B_z) and A (A_x, A_y, A_z). The key steps of tetrahedral 3D scan conversion are given as follows.

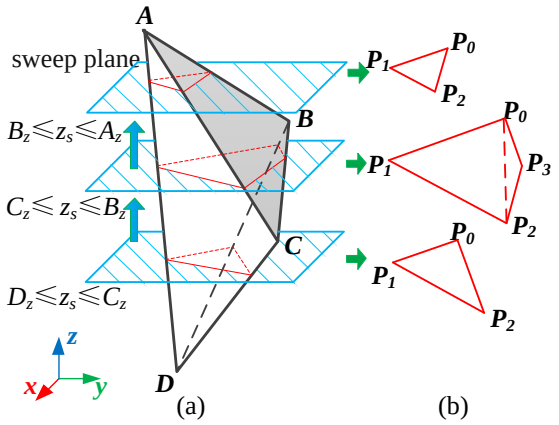


Fig.4 Scan the tetrahedron to get a series of triangles. (a) Scan $\triangle ABCD$ along the z-axis. (b) Triangles of different intervals.

(1) Scan the tetrahedron along the z-axis to get a series of triangular sections.

The scanning plane moves along the z-axis with a scanning pitch w. A series of triangular sections are got when scanning plane scans from $z_s = D_z$ to $z_s = A_z$. Assuming that the z coordinates of the four vertices A, B, C and D are different. When the scanning plane z_s is in the interval $D_z \le z_s \le C_z$ and $B_z \le z_s \le A_z$, the scanning plane and the tetrahedron have three intersection points P_0, P_1 and P_2, and a triangular section $\triangle P_0P_1P_2$ is formed (see Fig.4.b). When z_s is in the interval $C_z \le z_s \le B_z$, there are four intersection points P_0, P_1, P_2 and P_3 between the scanning plane and the tetrahedron, and two triangular sections $\triangle P_0P_1P_2$ and $\triangle P_0P_2P_3$ are formed.

When the scanning plane z_s=z_i intersects the side AB, the intersection of AB is calculated as:

$$\frac{x_i - B_x}{A_x - B_x} = \frac{y_i - B_y}{A_y - B_y} = \frac{z_i - B_z}{A_z - B_z} \qquad (1)$$

then

$$\begin{cases} x_i = B_x + (A_x - B_x)\dfrac{z_i - B_z}{A_z - B_z} \\ y_i = B_y + (A_y - B_y)\dfrac{z_i - B_z}{A_z - B_z} \end{cases} \qquad (2)$$

For the other sides, the intersections are calculated by a similar method.

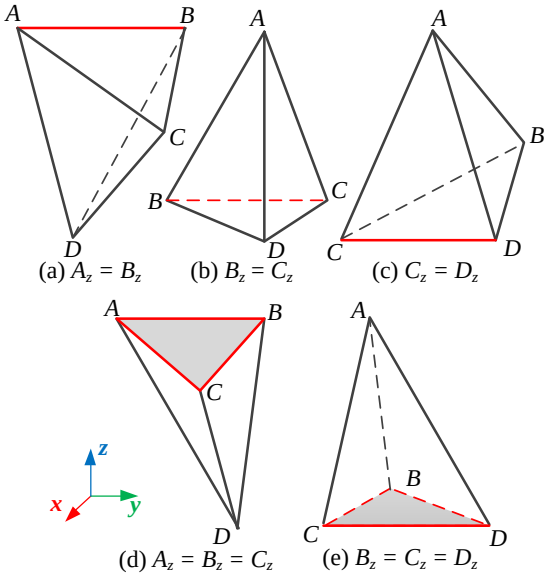


Fig.5 Different situations of tetrahedrons.

If the z-coordinates of two vertices are equal (see Fig.5.a-c), the calculation method of intersections is the same as above. If the z-coordinates of three vertices are equal and the scanning plane is coincident with the three vertices (see Fig.5.d-e), the triangular section consisting the three vertices is used directly in the next step.

(2) Voxelize all triangular sections and build the voxel model based on the presence value of voxels.

Build the 3D array buffer based on minimum bounding box of the object. The 3D array buffer has the same dimensions as the voxel space. It is used to store the presence values of the relevant voxel.

Let $\triangle abc$ be a triangular section of a tetrahedron. Assuming that the y coordinates of three vertices a (a_x, a_y, z_s), b (b_x, b_y, z_s) and c (c_x, c_y, z_s) are different. Sort

the vertices of $\triangle abc$ in y coordinate. Let c be the vertex with the smallest y coordinate, b the next, and so on with a. $\triangle abc$ is scanned along the y axis with a scanning pitch w. The scan line $y_s = y_i$ begins at $y_s = c_y$ and ends at $y_s = a_y$. When the scan line intersects the side ac, the x coordinate of the intersection is calculated as:

$$x_i = c_x + (a_x - c_x)\frac{y_i - c_y}{a_y - c_y} \qquad (3)$$

For the other sides, a similar approach is used.

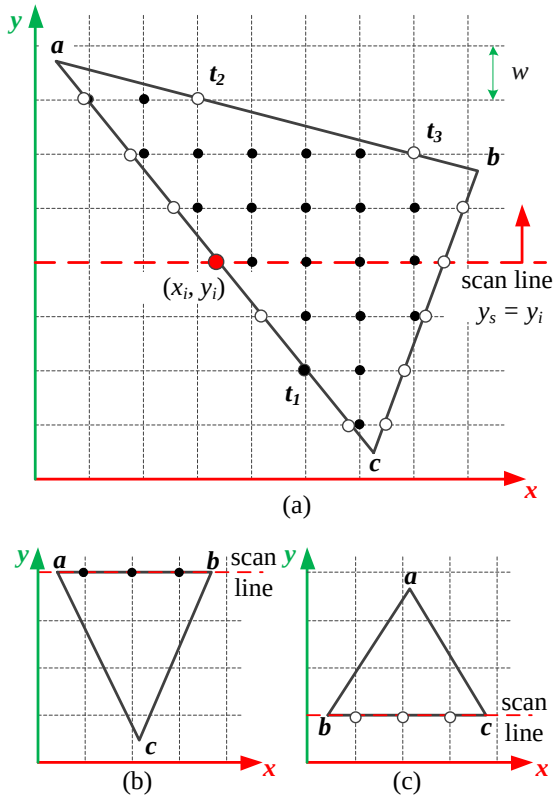


Fig.6 Voxelize triangular section. (a) Scan line intersects edge ac at (x_i, y_i). (b) $y_s = a_s = b_s$. (c) $y_s = b_s = c_s$.

After all intersections are obtained, the presence value of the relevant voxel inside the triangle is incremented by one in the 3D array buffer. When a voxel coincides with an intersection point and locates on the left side of Δabc as t_1 in Fig.6.a, the presence value of voxel plus 1. Conversely, if the intersection is on the right side as the point t_2 and point t_3 in Fig.6.a, the presence value plus 0.

When $y_s = a_s = b_s$, the presence values of coincident voxels plus 1 (see Fig.6.b). When $y_s = b_s = c_s$, the presence values of coincident voxels plus 0 (see Fig.6.c).

The odd-number value voxels that occur in the 3D array buffer belong to the solid, and the opposite occurs when the number is even. Build of the voxel model by the odd-number value voxels.

2.4 Compress the voxel model

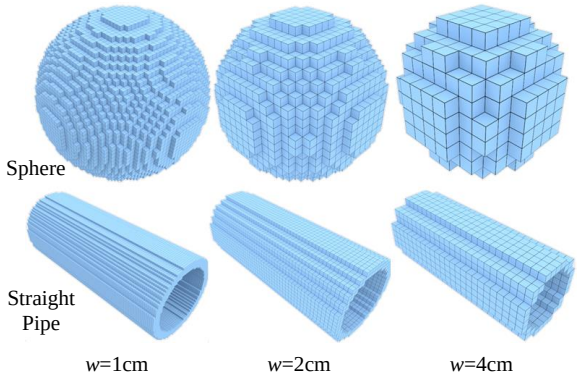


Fig.7 Voxel models for sphere and straight pipe when w = 1cm, w = 2cm and w = 4cm.

In order to calculate the radiation dose of a voxelized source, the voxels are used as point kernels for point-kernel simulation. As shown in Fig.7, the larger the width of the voxel, the greater the simulated error. In order to improve the accuracy of the radiation dose calculation, it is necessary to reduce the width of the voxel, which leads to an increase in calculation time. In order to improve computational efficiency while maintaining accuracy, it is desirable to compress voxels to reduce the number of voxels used for dose assessment. The basic idea of voxel compression is to compress adjacent voxels into a series of depth elements (dexel) [10].

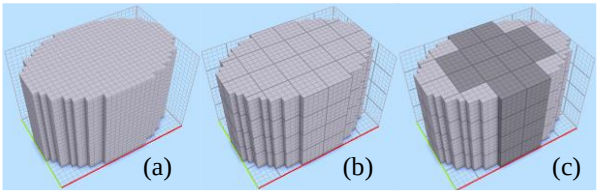


Fig.8 Compress the voxel model. (a) Build bounding box. (b) Divide the bounding box into a series of cubes. (c) Determine whether the cube is filled with full voxels.

The method of voxel compression is shown in Fig.8. At first, build the bounding box of the voxel model (see Fig.8.a). Next, divide the bounding box into a series of cubes according to the dexel width (see Fig.8.b). In the end, determine whether the cube is filled with full voxels. If so, these voxels in cube are compressed into a dexel. The new compressed model is consisted of compressed dexels and uncompressed voxels.

3 Experiments and results

In this paper, a voxelization algorithm has been proposed for converting the arbitrarily shaped geometries of radiation environment into voxel models. In order to verify the accuracy and validity of the algorithm, a series of simulation experiments were designed in this section. The first simulation experiment was run to illustrate the geometric modeling capability of the algorithm. The second simulation experiment was run to verify the validity of the voxel model compression. All simulation experiments presented in this work were tested on a Core i5 3.33GHz processor with 3.49GB of RAM memory. All the algorithms have been implemented in C ++, using the same compiler and optimizations.

3.1 Basic geometries voxelization modeling

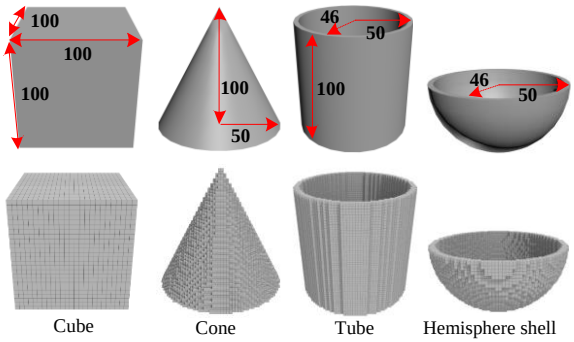


Fig.9 Four basic geometries used in experiments. The upper model is the original model and the lower model is the voxel model. The dimensions are in cm.

Since complex geometries consist of basic geometries, four basic geometries were used to test the geometric modeling capability of the voxelization algorithm. The four geometries are cube, cone, tube and hemispherical shell, where the convex surface, concave surface, flat surface and their combination were considered. Since the larger the voxel width, the larger the simulation error, we set the voxel width of the four basic geometries to 4 cm to produce a large error situation.

Table 2 Compare the volume between original models and voxel models

Basic geometries	Model volume (cm^3)		Deviation (%)	Time (ms)
	Original	Voxel		
Cube	1000000	986560	1.34	22
Cone	261799.38	259136	-1.02	24
Tube	120637.16	121600	0.80	28
Shell	57939.35	58880	1.62	41

The four basic geometries with specific dimensions and the voxel models are shown in Fig.9. It can be observed that the voxel model can accurately simulate the original model. The modeling results are shown in Table 2. It can be seen that after voxelization, the relative volume deviation between original models and voxel models is less than 1.7%, indicating that the algorithm can accurately simulate the original model. In addition, we can see that the modeling time is less than 42 ms, which shows that the algorithm can convert the solid model into voxels in real time.

3.2 Voxel model compression

In the point-kernel calculation, voxels and dexels can be used directly as point kernel. Obviously, compressing voxels can effectively reduce the number of point kernels and improve the efficiency of point-kernel calculation. In order to demonstrate the efficiency of the voxel compression method presented in this paper, we performed the voxel compression on the cube and cone used in the previous experiment.

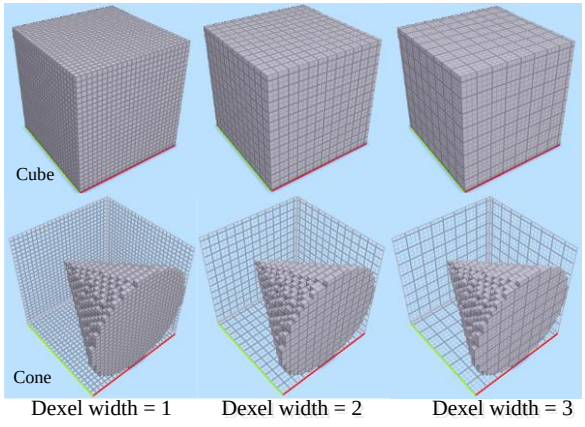


Fig.10 Compress the voxel model of cube and cone. The voxel width is 4 cm, and the dexel width is 1, 2 and 3.

As shown in Fig.10, the voxel width of voxel model is 4 cm, and the dexel width is 1 (no compression), 2 and 3. The number of point kernels under different compression conditions is shown in Table 3. After compression, the number of point kernels of the cube decreased from 15415 to 3039, and the number of point kernels of the cone decreased from 4049 to 1214. The compression results demonstrate the effectiveness of the voxel compression method.

Table 3 Number of point kernels in different conditions

Shapes	Dexel width	Point kernels' number		
		Voxel	Dexel	Total
Cube	1	15415	0	15415
	2	2215	1650	3865
	3	2563	476	3039
Cone	1	4049	0	4049
	2	809	405	1214
	3	1565	92	1657

4 Conclusions

A voxelization modeling algorithm based on tetrahedral 3D scan conversion for point-kernel gamma ray calculations is proposed. The geometric capability of algorithm was verified by simulating basic geometries, which include convex surface, concave surface, flat surface and their combination. In the voxel compression experiment, compressing voxels can effectively reduce the number of point kernels.

Compared with the existing similar methods, the major advantage of this algorithm is that it can visually perform geometric modeling for arbitrary shape 3D geometries using the modeling capabilities of 3dsMax without writing the textual description. Using the voxelization modeling method, the proposed algorithm can automatically convert arbitrary shape solid models to voxel models, which makes the modeling more flexible. The algorithm can handle arbitrarily shaped geometries: with or without holes, self-intersecting, manifold or non-manifold. It has fast calculation speed and good shape adaptability.

Acknowledgements

This research work was funded by Decommissioning of Nuclear Facilities and Radioactive Waste Management Research, Fundamental Science on Nuclear Safety and Simulation Technology Laboratory, Harbin Engineering University. Project supported by the Natural Science Foundation of Heilongjiang Province, China (Grant NO.A2016002), the technical support project for Suzhou Nuclear Power Research Institute(SNPI)(NO.029-GN-B-2018 -C45-P.0.99-00003), the Foundation of Science and Technology on Reactor System Design Technology Laboratory (HT-KFKT-14-2017003) and the project of Research Institute of Nuclear Power Operation (No.RIN180149-SCCG).

References

[1] INGERSOLL, D.T.: User's manual for PUTZ: a point-kernel photon shielding code. Oak Ridge National Lab, 1986.

[2] ZHANG, L., Li, C., ZHANG, Y., and ZHANG, C.: Gad-cga:an improved version of qad-cg (a point kernel code for neutron and gamma-ray shielding calculations with combinatorial geometry technique), Nuclear Power Engineering, 1988.

[3] BORGLUND, N., ERIKSSON, J., FUMERO, E., KJÄLL, P., MÅRTENSSON, L., and ORSHOLM, C., *et al.*: Geometry package for monte carlo simulations on cad files, Nuclear Inst & Methods in Physics Research A, 2004, 525(1), 417-420.

[4] THEIS, C., BUCHEGGER, K. H., BRUGGER, M., FORKEL-WIRTH, D., ROESLER, S., and VINCKE, H.: Interactive three-dimensional visualization and creation of geometries for monte carlo calculations, Nuclear Inst & Methods in Physics Research A, 2006, 562(2), 827-829.

[5] KARABASSI, E. A., PAPAIOANNOU, G., and THEOHARIS, T.: A fast depth-buffer-based voxelization algorithm, Journal of Graphics Tools, 1999, 4(4), 5-10.

[6] Li, W., Fan, Z., Wei, X., and KAUFMAN, A.: Flow simulation with complex boundaries, Gpu Gems, 2005.

[7] RUEDA, A. J., SEGURA, R. J., FEITO, F. R., MIRAS, J. R. D., and OGYAR, C.: Voxelization of solids using simplicial coverings. Václav Skala - UNION Agency, 2004.

[8] FEITO, F. R., and TORRES, J. C.: Inclusion test for general polyhedral, Computers & Graphics, 1997, 21(1), 23-30.

[9] OGÁYAR, C.J., RUEDA, A.J., SEGURA, R.J., and FEITO, F.R.: Fast and simple hardware accelerated voxelizations using simplicial coverings, Visual Computer, 2007, 23(8), 535-543.

[10] VAN HOOK, T.: Real-time shaded NC milling display, Conference on Computer Graphics and Interactive Techniques, SIGGRAPH. DBLP, 1986, Vol.20, 15-20.