Proceedings of Algoritmy 2005
pp. 309-318

CENTERED CUBIC LATTICE METHOD COMPARISON*

JAN PATERA† AND VÁCLAV SKALA‡

Abstract. The Centered Cubic Lattice method is used for an iso-surface extraction. This paper focuses on its comparison with other fundamental methods (Marching Cubes, Marching Tetrahedra) that are also used to extract an iso-surface from regular volume data. Iso-surfaces that are generated using above mentioned methods and mathematical data are compared due to area and volume approximation. The Hausdorff and RMS distances are also utilized for comparisons. Also all possible cell tessellations into 6 tetrahedra are discussed and analyzed. We think that such a comparison can be helpful to decide which of tested methods should be chosen when we have concrete demands concerning the iso-surface. Also newly developed methods can be compared with existing ones in the similar way.

Key words. Computer graphics, Iso-surface, Comparison , Error, Hausdorff distance, Volume data

AMS subject classifications. 65D18, 65D15, 65G99, 28A78

1 Introduction. In the recent period of time volume data have started to play a significant role in many scientific areas and are spread across many professions. In medical field, various devices, such as Computer Tomography (CT) scanners, Magnetic Resonance Imaging (MRI) scanners produce volume data. The volume data are also produced as a result of mathematical or physical simulations and experiments and researchers need to visualize such data.

There are two main techniques for the volume data visualization. The first approach is based on a volume rendering (ray-tracing-like methods), the second one on a surface rendering (iso-surface-extraction-like methods). The volume rendering methods are complex and work with the whole volume data. This paper is concentrated on surface rendering methods that visualizes surfaces stored in the volume data (so called iso-surfaces). The extracted iso-surface is determined by a threshold value. All the points on the iso-surface have their value equal to the threshold.

The field of the iso-surface extraction is quite large. There are many approaches used for the iso-surface extraction such as view-dependent techniques, parallel or distributed approaches, external memory (or sometimes called I/O) techniques, multiresolution (LOD) based extractions and others. In general, we can describe the iso-surface generation and visualization process with the following steps:

1. Search for all active cells (cells that are intersected by the iso-surface)
2. The iso-surface and its normal vectors approximation within these cells (e.g. by a triangle set)
3. Iso-surfaces visualization (visualization of a set of triangles; different iso-surfaces can be visualized with different colors depending on a selected threshold, etc.)

The first phase of the iso-surface extraction can be accelerated using a wide set of speed up algorithms [7], [9], [11], [12], [18] or [19]. However, we are interested not that much in speed of the extraction process but in properties of the output set of triangles.

There are many various methods for the iso-surface generation and each such a method generates generally different approximation of a searched iso-surface for a given threshold. There is probably no way how to compare the resulting iso-surfaces to each other unless we know how the iso-surface should look like. Therefore we try to compare generated iso-surfaces produced by different methods.

---

* Department of Computer Sciences and Engineering, Faculty of Applied Sciences, University of West Bohemia, Univerzitní 22, Plzeň, Czech Republic, (hopatera | skala)@kiv.zcu.cz

† supported by Grant No.: MSM 235200005

‡ supported by project NoE – 3DTV PLT 511568

A comparison can be made with respect to the volume data. When we generate the volume data using some mathematical or physical model, we are able to gain some additional information concerning the object that is utilized to make a comparison more informative and objective. As additional information, we assume e.g. possibility to compute area or volume of such an object. For real data sets, when we do not have any additional information concerning a scanned object, we can just use general approaches for comparison, e.g. Hausdorff distance or root mean square (RMS) distance.

This paper is organized in the following way. At first, compared methods are described. Afterwards, we will explain used approaches for the comparison and how the data are generated. The last two sections are devoted to the methods comparisons, error analysis and conclusions.

2 Methods Descriptions.

2.1 Marching Cubes. There are many kinds of volume data. From simulations, we often get unstructured volume data. On the other hand from medical imaging the output data is structured one. We aimed at comparison of iso-surface generation methods that are used for structured data, especially for regular grids. Compared methods do not differ in the kind of used interpolation but only in the way they divide a cell into tetrahedra. The well-known method is Marching Cubes (MC) method that was firstly published by Lorensen and Cline [13]. At first, we will describe this method because the main principle of other compared methods is similar.

The input volume data consists of samples organized into a regular 3D Cartesian grid. From such a grid, we can easily obtain a set of cells. The cell has in this case a cube shape and consists of eight corresponding samples from two adjacent sample planes. Four samples are from the first plane and four samples are from the second plane. MC method processes sequentially all the cells that can be found in volume data. The iso-surface, which we are looking for, is specified by a threshold value.

Each cell is processed separately. Firstly, the cell *index* is computed. The cell has eight vertices, let us name them from A to H, and each vertex has its data value. Depending on a selected threshold the vertex is assigned a binary value $index = \mathrm{ABCDEFGH}_B$. Each bit of the *index* is 0 when the data value in the corresponding vertex is less than the threshold and 1 otherwise.

Based on the *index*, we are able to distinguish 256 cases how the iso-surface can intersect the cell, because each vertex can be inside or outside of the iso-surface. When the *index* is 0 or 255 the cell is not intersected by the iso-surface, otherwise such a cell is called an active cell. The purpose of the *index* will be described later. For an active cell, normal vectors are computed in all its vertices using symmetric or asymmetric difference of data samples.

Each *index* represents a different case how the iso-surface can intersect the cell. All these cases can be tabularized and easily triangulated using linear interpolation. The vertices of triangles lay on the cell edges. Note, that vertices of triangles are interpolated only on the cell edges, this will not be true for other methods. Maximum of four triangles per the cell is needed to approximate the iso-surface. For each triangle vertex a normal vector is computed from normal vectors in the cell vertices (using linear interpolation).

Each cell face is shared by another cell. Due to such sharing, the iso-surface is continuous among adjacent cells. Note that there can be ambiguous faces at which the triangulation proposed by [13] will produce holes. There are few approaches how to avoid the holes. Ambiguous cases can be detected and a special triangulation can be applied [17]. The cells can be divided into tetrahedra and resulting simplices

triangulated in a little bit different way as we will describe in the next section. Other approaches are out of the scope of this paper, see [2], [3], [6], [14], [15], [16].

The algorithm complexity of MC method is *O(N)*, where *N* is the number of all cells. The following two described algorithms are dealing with tetrahedral tessellations of the cube cell.

2.2 Marching Tetrahedra. Marching Tetrahedra (MT) method is based on the same principle as MC method. The significant difference is that the cube cell is furthermore split into tetrahedra. There are two main possibilities how to do that.

First the cell is divided into six tetrahedra (MT6), [16], [23] or [24]. This is probably the most often used division of the cube cell. Kolcun [10] investigated all the cube cell tessellations into six tetrahedra and defined a surface representation of the tessellation schemes, which uniquely describes each particular division scheme. The five non-equivalent tetrahedra (with respect to the rotation) which can be created using the cube cell vertices are shown in Fig. 1.

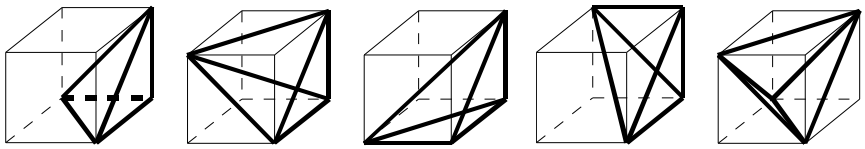


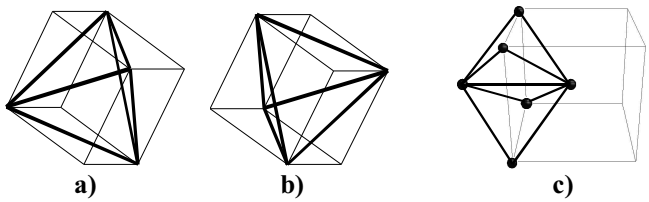
FIG. 1 - *Five non-equivalent tetrahedra within the cell*

The number of all possible tetrahedra which can be constructed from eight cell vertices is following [10]:

$$\binom{8}{4} - planar\ \ quadrilaterals = \frac{8 \cdot 7 \cdot 6 \cdot 5}{4 \cdot 3 \cdot 2 \cdot 1} - (6 + 6) = 58 \tag{1}$$

The planar quadrilaterals are all six cell faces and six diagonal planar cuts of the cube, these have to be subtracted because they do not represent a tetrahedron. Kolcun [10] also counts all possible cube cell tessellations into six tetrahedra using incidence matrix and a graph theory. The total number of all tetrahedral tessellations of the cube cell into six tetrahedra is 72. An example of one particular division scheme is shown in Fig. 11. Not all 72 division schemes are suitable for the iso-surface extraction. Some of them do not maintain continuity among adjacent cells (same problem as in MT5, see bellow).

The second possibility is that the cell is divided into five tetrahedra (MT5) [8], [16], [23]. There are several ways how the cube cell can be divided into five tetrahedra (e.g. Fig. 2.a and Fig. 2.b). When using the five tetrahedra division scheme, it is necessary to alternate two different splitting schemes. Otherwise, the continuity of the extracted iso-surface will not be maintained properly among adjacent cells because each cell would use interpolation at a different face diagonal within shared faces.



a) **b)** **c)**

FIG. 2 – *a), b) MT5 tetrahedra division schemes, c) CCL division scheme*

After the cell is split into tetrahedra (each has four vertices), the *index*=$ABCD_B$ for each tetrahedron is computed separately and tetrahedron is processed separately in the similar way as the cube cell in the MC method. There are only 16 possibilities how the tetrahedron can be intersected with iso-surface. These methods generate at most two triangles per tetrahedron.

Five or six tetrahedra decompositions introduce new edges at which vertices of triangles are to be interpolated. For five tetrahedra the interpolation will be held on face diagonals of the cube cell, for six tetrahedra both face and internal diagonals are used. If we look at the five tetrahedra division, there is one tetrahedron with different shape and size. For some six tetrahedra splitting scheme all the tetrahedra can be the same.

2.3 Centered Cubic Lattice. The last method that will be compared to previously mentioned ones is the Centered Cubic Lattice (CCL) method, see [5]. This method is little bit different, because it splits the cube cell into 24 tetrahedra.

The difference is that the resulting tetrahedra are partially shared among adjacent cells and a new data value is introduced to the center of gravity of the processed cell, as outlined in Fig. 2.c. There are several ways how to compute the value of the central sample, e.g. the arithmetic mean or the weighted mean.

Each tetrahedron is then processed separately in the same way as in MT6 or MT5 methods. Similarly to previous methods this kind of splitting of the cell also introduces new edges at which the interpolations will be made. These are edges among adjacent central points. In this division scheme, all the 24 tetrahedra dimensions are the same.

There are also other possible decompositions of the cube cell, e.g. [20] that decomposes parallelepiped cell into two tetrahedra and one octahedron or [10] where the cell is split into four pentahedra. These techniques were not included into this comparison study.

3 Comparisons Approaches. This section covers basic approaches that we used for comparison purposes.

3.1 Root Mean Square Distance. We use the Root Mean Square (RMS) of computed distances between two extracted iso-surfaces. At first, we define a distance between a point p (on the surface S) and a surface S' (with points p') as [1]

$$d(p, S')=min||p-p'||, \tag{2}$$

for all p' from S'. RMS distance in discrete case is then defined as in [21]

$$RMS(S, S') = \frac{\sqrt{x_1^2 + ... + x_n^2}}{n}, \tag{3}$$

where n is a number of points of a mesh S', x_i (where $i=1..\ n$) represents the distance of corresponding point p_i' from S, $x_i=d(p_i', S)$. We compare S' to S.

Note that RMS is not symmetrical. We do not use symmetrical RMS distance in our tests, thus it is not defined here. This measurement is computed with METRO tool [4].

3.2 Hausdorff Distance. As mentioned before, we use also Hausdorff distance [1] for comparisons. Hausdorff distance between two surfaces S and S' is defined as

$$d_H(S, S')=max\ d(p, S'), \tag{4}$$

for all points p on the surface S. Note the important thing that Hausdorff distance is not symmetrical $d(S,S')\neq d(S',S)$. When we call $d(S,S')$ a forward and $d(S',S)$ a backward distance, a symmetrical Hausdorff distance [1] can be defined as

(5) $$d_{SH}(S,S')=max(d(S,S'),\ d(S',S)).$$

The symmetrical difference provides better error measurement for two surfaces. In this case, we also utilized a METRO software tool (described in [4]) for accurate computation of Hausdorff distance of two discrete surfaces (triangle meshes). The METRO tool was generally used to compare original mesh with its simplified (e.g. decimated) version. We use it for comparison of two iso-surfaces, each generated with different method.

Both the distances RMS and Hausdorff are calculated according to some source mesh. As such a mesh, we use a mesh generated with MC method (the simplest one) that do not divide a cell into tetrahedra.

3.3 Mathematical Data. If we know the object equation and its dimensions, we are able to compute some additional information concerning the object, such as its surface area, object volume, etc. We believe that these properties are worth to compute, because they can help us to differentiate among the quality of examined methods.

Surface area – the iso-surface is generated by an extraction method in a form of a set of triangles. We compute the total area as a sum of all triangles areas. Then we can compute the area of the mathematical object and compare it with the iso-surface area obtained. For special objects such as sphere, we are able to track the error behaviour dependency on the sphere radius changes (threshold changes).

Volume enclosed with the iso-surface – for basic objects the volume is computed using appropriate mathematical formula. The volume enclosed with the iso-surface is computed in the following manner (for tetrahedra methods only, [22]). There are three cases for each tetrahedron that can occur:

1. The whole tetrahedron is outside of the iso-surface – does not affect the total volume computation.
2. The whole tetrahedron is inside of the iso-surface – the whole tetrahedron contributes to the total volume. The tetrahedron volume is computed easily.
3. The tetrahedron is intersected with the iso-surface – we have to compute the part of the tetrahedron which is inside of the iso-surface. As there are at most two triangles generated per tetrahedron, these triangles form two small tetrahedra with appropriate tetrahedron vertex and we are able to compute the volume of the tetrahedron part which contributes to the total volume.

The two mentioned geometric properties are the main aspects that we used for extraction methods output comparison. The obtained results are shown in the next section.

4 Results. Let us mention and describe the data sets we used for comparisons and give the reasons why we chose them. We used mainly mathematical objects that were generated within volume data, see Fig. 3. The brief description of used data sets follows in upcoming paragraphs.

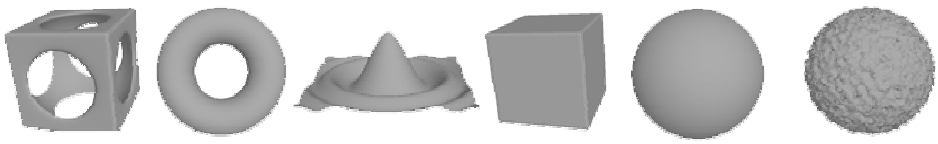


FIG. 3 - *Objects (csph, torus, sombrero, cube, sphere and noisedsph)*

4.1 Used Objects. Here, we would like to mention how the testing data are generated from basic mathematical objects (given by their mathematical equation).

Sphere – a sphere is an example of an object that we use to follow the error behaviour dependency on the sphere radius. The modified sphere equation is used for data generation

$$F(x,y,z) = \sqrt{(x-s_X)^2 + (y-s_Y)^2 + (z-s_Z)^2} - r\,, \tag{6}$$

where x, y and z are samples coordinates, s_X, s_Y and s_Z are the sphere centre coordinates, r is the sphere radius and $F(x,y,z)$ is a corresponding sample value. This equation assigns data value to all the volume data samples. The sphere is then represented with a zero threshold iso-surface. The samples that are inside of the sphere have negative values, on the sphere surface zero values and samples placed out of the sphere have positive values. The sample value represents the distance of the sample from the sphere surface. The radius was 25 in our experiments.

Cell edge has a length 1 for our purposes. The object dimensions (e.g. radius, edge length) are then related to a cell edge length.

Noised sphere – (noisedsph) to study the influence of the noise on the shape of the output set of triangles we generate a noised sphere. The random noise is introduced (added) to all samples of the volume data. The size of the noise is given in percentage from the sphere radius size. We used radius 25 and 10% noise.

Cube – we use this kind of an object to follow the behaviour and properties of the iso-surface on edges. We will show the iso-surface behaviour mainly visually. Data are generated similarly as in the previous case using the distance of the sample from the closest face, edge or vertex (in other words the sample distance from the cube). The inner, on surface and outer samples have the negative, zero and positive values respectively. Cube was generated using $a=b=c=42$.

Cube minus sphere – (csph) such an object was constructed to combine both properties of the sphere ($r=25$) and the cube ($a=b=c=42$). The generation of it is a little bit complicated. At first, the cube is generated in the volume data. Afterwards, the values of all samples that are closer to the sphere than to the cube are modified to the new value.

Torus – is the next object. Torus is defined with the following equation [21]

$$F(x,y,z) = \sqrt{\left(r_1 - \sqrt{(x-s_X)^2 + (y-s_Y)^2}\right)^2 + (z-s_Z)^2} - r_2\,, \tag{7}$$

where x, y and z are samples coordinates, s_X, s_Y and s_Z are the torus centre coordinates, r_1 is a torus main radius, r_2 is a torus secondary radius and $F(x,y,z)$ is a corresponding sample value. The samples values are negative, zero or positive as well. Torus dimensions are $r_1=20$ and $r_2=42$ in our case.

Sombrero – is the last mathematically generated object we use. It is a surface defined with the mathematical equation (taken from Derive mathematical program)

$$F(x,y,z) = y - \frac{r\cos(s((x-s_X)^2 + (z-s_Z)^2))}{t + (x-s_Z)^2 + (z-s_Z)^2}\,, \tag{8}$$

where x, y and z are sample coordinates, s_X, s_Y and s_Z are the sombrero centre coordinates, and $F(x,y,z)$ is a corresponding sample value and r, s and t are constants

modifying the shape of the function. Parameters of the sombrero, we used, are *i=12*, *j=0.25* and *k=3*. The following section covers the obtained results.

4.2 Performed Tests and Results. This section describes the obtained results using comparison approaches and data sets that were mentioned in previous sections.

The more tetrahedra is used the larger area is extracted for all tested objects that have edges, see Fig. 4. The results in Fig. 4 are relative due to mathematically computed area and volume (hence 1 for mathematical results). For objects like torus (does not have edges) the results were approximately the same as for a sphere. We think that for the area approximation purposes the best choice is MC method without division of the cube cells into tetrahedra.

From the view of volume approximation (the volume enclosed by the iso-surface) the MT5 method is in most cases slightly better than MT6 method and both methods are approaching to the original volume from below, see Fig. 4. The CCL method in the other hand in most cases approaches to mathematically computed volume from above. The MC method is not included in the volume comparison chart, because it is hard to compute the volume enclosed with the iso-surface (due to 256 cases).

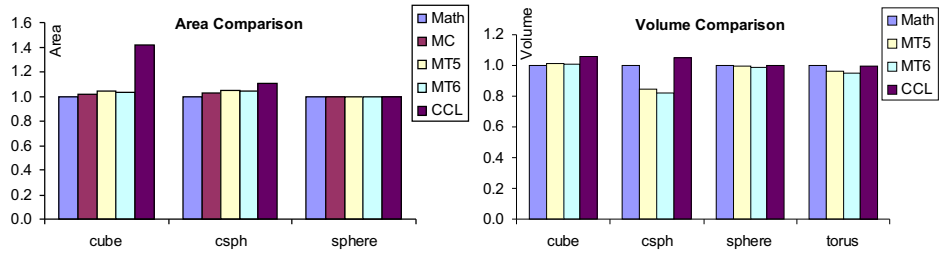


FIG. 4 - *Area and volume comparison (relative to mathematical volume)*

If we use Hausdorff distance a sphere and a sombrero give approximately similar results compared to torus. From the view of Hausdorff distance the MT6 method gives the worst results for all tested objects, see Fig. 5. As you can see for noisedsph object the CCL method is the best choice. The overall best method in this case is probably MT5 method because it does not generate as much triangles as CCL method and gives reasonable results.

The next test uses the RMS distance computation. Note that RMS and Hausdorff distances are related to the MC method. For a sphere and a torus the obtained results were slightly less than results for a sombrero. Again, when the object has edges the CCL method is the worst from the view of RMS distance, see Fig. 5. For noisedsph object the CCL method gives the best results. We suppose that the central cell sample value computation (using arithmetic mean) filters data a little bit.

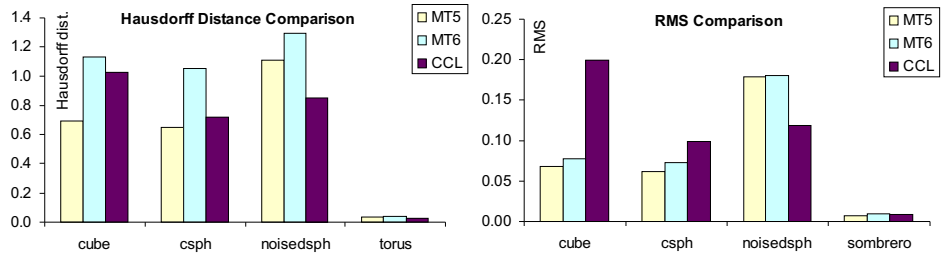


FIG. 5 - *Hausdorff and RMS distances histograms*

In Fig. 6 you can see that the choice of the method (or division scheme in MT6 method) has a great impact on how objects edges will be extracted. For different division schemes we obtain highly different iso-surface approximations of object edges.

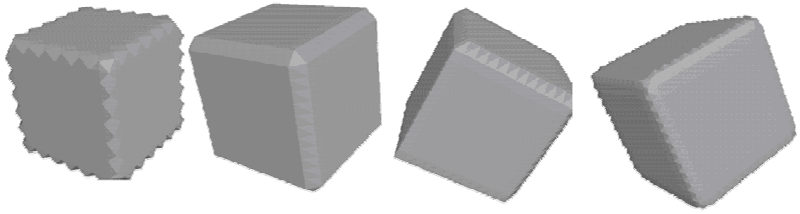


FIG. 6 - *Iso-surface on edges (MT5, MC, MT6, CCL)*

4.3 The MT6 Additional Tests. Next paragraph deals with the cell division schemes into six tetrahedra. We have tested all 72 division schemes from [10]. As we stated not all of them are suitable for the use in iso-surface extraction process. There are only 22.2% division schemes that maintain continuity among adjacent cells from all possible division schemes. These are 1, 6, 9, 17, 23, 25, 31, 35, 38, 44, 47, 50, 57, 62, 65 and 72 (schemes labeling is the same as in [10]).

The iso-surface area and the volume enclosed with the extracted iso-surface can be computed. We have performed several tests concerning not only the area and volume error. We have also measured the average minimal and maximal angles within the extracted iso-surface to get rid of acute and obtuse angles in the resulting iso-surface. These tests were performed to pick up the best overall division scheme for MT6 method. Mathematical objects sphere and cpsh were used as well as real data set syn_64.vol. The division scheme number 38 (Fig. 8) gave the best overall results.

Now, have a look at the area of the extracted iso-surface when a threshold is changing and MT6 method with the picked division scheme is used, see Fig. 7.

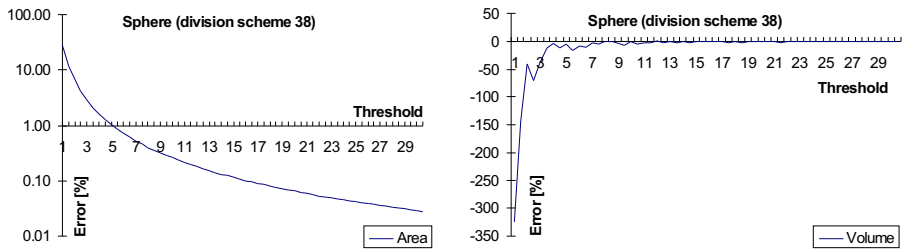


FIG. 7 – *Area and volume errors behaviour*

A relative area error is defined in a following way

$$S_{error} = \frac{S - S_{tr}}{S} * 100 \ \ [\%], \tag{9}$$

where S_{tr} is an area of the extracted iso-surface and S is mathematically computed volume of the sphere. The volume that is enclosed with the extracted iso-surface can be also computed and the relative volume error is defined similarly as

$$V_{error} = \frac{V - V_{tr}}{V} * 100 \ [\%] . \tag{10}$$

As it can be observed from Fig. 7, with increasing threshold the error is getting smaller. That is due to the fact that with increased threshold the extracted sphere is bigger and the amount of active cells is also higher thus the sphere is sampled with higher number of samples.

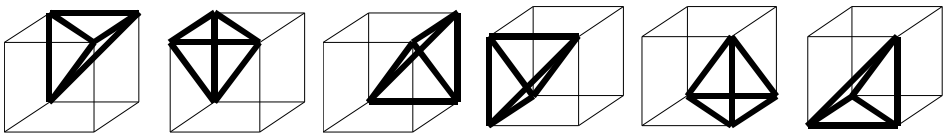


FIG. 8 – *Division scheme number 38*

4 Conclusions. We compared fundamental methods for the iso-surface extraction utilizing Hausdorff distance, RMS distance, iso-surface area and volume.

Hausdorff distance is in fact the biggest distance between two compared surfaces (extreme distance). Hausdorff distance can be also used, in our opinion, to compare iso-surfaces that are extracted from real data sets when using different methods for their extraction but there is a need to have some relative iso-surface.

In general, we are more interested in an average distance between two surfaces (the RMS distance). In the case of RMS distance, the CCL method generally gives worse results than other tested methods.

Interesting is that a volume of objects is approximated with the similar difference no matter of method used except for csph object.

The division number 38 from [10] gives the best overall results, thus we would recommend its use in MT6 method.

The quality of the extracted set of triangles for noised sphere was in general bad. We think that for noised data the CCL method is the most suitable.

It is also important to realize that for real data we do not know the exact area or volume of the object, thus the comparison can not be made.

As a future work we would like to examine also mentioned division schemes that are not included in this paper (two tetrahedra plus one octahedron [20] or four pentahedra [10]).

Acknowledgements. We want to thank to Dr. Ivana Kolingerová for her huge help and support during preparation of this paper. Our thanks also belong to Mgr. Alexej Kolcun, CSc. for his helpful consultation in the field of our interest.

REFERENCES

[1] N. ASPERT, D. SANTA-CRUZ AND T. EBRAHIMI, *Mesh Measuring Errors Between Surfaces Using The Hausdorff Distance*, In Proceedings of the IEEE International Conference in Multimedia and Expo (ICME), Lausanne, Switzerland, August 26-29, Vol. 7 (2002), pp. 705-708.

[2] K. S. BONNEL, M. A. DUCHAINEAU, D. R. SCHIKORE, B. HAMANN AND K. I. JOY, *Material Interface Reconstruction*, IEEE Trans. on Visualization and Comp. Graphics, Vol. 9(4) (2003), pp. 500-511.

[3] P. CIGNONI, F. GANOVELLI, C. MONTANI AND R. SCOPIGNO, *Reconstruction of Topologically Correct and Adaptive Trilinear Isosurfaces*, Computers & Graphics, Vol. 24(3) (2000), pp. 399-418.

[4] P. CIGNONI, C. ROCCHINI AND R. SCOPIGNO, *Metro: Measuring Error on Simplified Surfaces*, Computer Graphics Forum, Blackwell Publishers, Vol. 17(2) (1998), pp. 167-174.

[5] S. L. CHAN AND E. O. PURISIMA, *A New Tetrahedral Scheme for Iso-surface Generation*, Computers & Graphics, Elsevier Science Limited, Vol. 22(1) (1998), pp. 82-90.

[6] E. V. CHERNYAEV, *Marching Cubes 33: Construction of Topologically Correct Isosurfaces*, Institute for High Energy Physics, Moscow, Russia, Report CN/95-17, 1995.

[7] M. GILES AND R. HAIMES, *Advanced interactive visualization for CFD*, Computing Systems in Engineering, Vol. 1(1) (1990), pp. 51-62.
[8] M. HALL AND J. WARREN, *Adaptive Polygonalization of Implicitly Defined Surfaces*, IEEE Computer Graphics and Applications, Vol. 10(6) (1990), pp. 33-42.
[9] T. ITOH, Y. YAMAGUCHI AND K. KOYAMADA, *Fast Isosurface Generation Using the Volume Thinning Algorithm*, IEEE Transactions on Visualization and Computer Graphics, Vol. 7(1) (2001), pp. 32-46.
[10] A. KOLCUN (SUPERVISOR: R. BLAHETKA): *Preprocessing for Application of FEM Method in Geomechanical Tasks (in Slovak)*, Dissertation, Ústav geoniky Ostrava, Akademie věd ČR, 1999.
[11] M. VAN KREVELD, R. VAN OOSTRUM, C. BAJAJ, V. PASCUCCI AND D. SCHIKORE, *Contour Trees and Small Seed Sets for Iso-surface Traversal*, In Proceedings 13th Annual Symposium Computational Geometry, 1997, pp. 212-220.
[12] Y. LIVNAT, S. G. PARKER AND CH. R. JOHNSON, *Fast Iso-surface Extraction Methods for Large Imaging Data Sets*, Center for Scientific Computing and Imaging, Department of Computer Science, University of Utah, Salt Lake City, USA, 1999.
[13] W. E. LORENSEN AND H. E. CLINE, *Marching Cubes: A High Resolution 3D Surface Construction Algorithm*, Computer Graphics, Vol. 21(4) (1987), pp. 163 - 169.
[14] A. LOPEZ AND K. BRODLIE, *Improving the Robustness and Accuracy of the Marching Cubes Algorithm for Isosurfacing*, IEEE Transactions on Visualization and Comp. Graph., Vol. 9(1) (2003), pp. 16-29.
[15] B. K. NATARAJAN, *On Generating Topologically Consistent Isosurfaces from Uniform Samples*, The Visual Computer, Vol. 11 (1994), pp. 52-62.
[16] P. Ning and J. Bloomenthal, *An Evaluation of Implicit Surface Tilers*, Computer Graphics and Applications, Vol. 13(6), (1993), pp. 33-41.
[17] W. SCHROEDER, K. MARTIN AND B. LORENSEN, *The Visualization Toolkit*, 2nd Edition, Prentice Hall PTR, ISBN 0-13-954694-4, 1998.
[18] H. W. SHEN, CH. D. HANSEN, Y. LIVNAT AND CH. R. JOHNSON, *Isosurfacing in Span Space with Utmost Efficiency (ISSUE)*, IEEE Visualization 96, 1996, pp. 287-294.
[19] H. SHEN AND C. R. JOHNSON, *Sweeping simplicies: A Fast Iso-surface Extraction Algorithm for Unstructured Grids*, Proc. of Visualisation 95, IEEE Computer Society Press, Los Alamos, CA, 1995.
[20] T. TAKAHASHI AND T. YONEKURA, *Isosurface Construction from a Data Set Sampled on a Face-Centered-Cubic Lattice*, Proceedings of ICCVG 2002, Vol. 2 (2002), pp. 754-763.
[21] E. W. WEISSTEIN, *MathWorld*, A Wolfram Web Resource, http://mathworld.wolfram.com.
[22] V. SKALA AND A. B. TERRADES, *Two Methods for Iso-Surface Extraction from Volumetric Data and Their Comparison*, Machine Graphics & Vision, Poland Academy of Sciences, Volume 9 (2000), ISSN 1230-0535, pp. 149-166.
[23] M. KREJZA AND V. SKALA, *Iso-surface Extraction and Visualization in Modular Environment System*, Computational Geometry Workshop Proceedings, Kocovce, STU Bratislava, Slovakia (2000), ISBN 80-227-14587-7, pp. 68-75.
[24] V. SKALA, *Precision of Iso-surface Extraction from Volume Data and Visualization*, Algoritmy 2000, International Conference, Slovakia (2000), ISBN: 80-227-1391-0, pp. 368-378.