

INTERACTIVE VISUALISATION AND COMPARISON
OF TRIANGULAR SUBDIVISION SURFACES

Krassimira Vlachkova, Todorka Halacheva

(Submitted by Academician I. Popchev on January 8, 2014)

Abstract

We present a new program package for interactive implementation and 3D visualisation of three fundamental algorithms for triangular surface subdivision. Namely, these are Loop subdivision, Modified Butterfly subdivision, and Kobbelt $\sqrt{3}$ -subdivision. Our work and contributions are in the field of experimental algorithmics and algorithm engineering. We have chosen Java applet application and Java 3D as our main implementation and visualisation tools. This choice ensures platform independency of our package and its direct use without restrictions. The applet can be used for experiments with user's data sets since it works with host file system. The latter allows its usage for research and educational purposes. We have implemented extensive experiments with our package using meshes of different type and increasing complexity. We compared the behaviour of the three algorithms with respect to various criteria. The experimental results are presented and analysed.

Key words: surface subdivision, triangle mesh, mesh regularity

2010 Mathematics Subject Classification: 65D17, 68U07

1. Introduction. Subdivision algorithms are one of the most successful modern techniques for modelling smooth free-form shapes in 3D, [1–4]. They allow simple and efficient construction of smooth surfaces of arbitrary topology starting from an initial polygonal *control mesh*. These algorithms recursively subdivide the control mesh to form a new refined mesh with more *faces*, *edges*, and *vertices* which is topologically equivalent to the original. A smooth surface

This work was partially supported by the Bulgarian Science Fund under Grant No. DFNI-T01/0001 “Effective methods and algorithms for geometric modelling”.

is obtained as a limit of the recursive process of subdivision. Because of their simplicity, efficiency, and high quality results, the subdivision algorithms have applications in various fields. These include CAGD, computer graphics, solid modelling, computer game software, computer animation, aerospace and automotive design, industrial design, architecture, medicine, and others.

In this paper we focus on three subdivision algorithms: Loop subdivision (\mathcal{L}), Modified Butterfly subdivision (\mathcal{MB}), and Kobbelt $\sqrt{3}$ -subdivision (\mathcal{K}). The main contributions of our work can be summarised as follows:

1. We implemented a new software package for interactive visualisation, manipulation and comparison of the subdivision surfaces generated by these three algorithms. The package is user friendly and provides many options for testing and experimentation.
2. We implemented extensive experimental work. The results were analysed to reveal main differences between the tested algorithms with respect to various criteria including quality, performance, and efficiency.

There is a number of general purpose geometric modelling and open source mesh processing packages available on the net that include as routines algorithms for surface subdivision. To our knowledge, these are CGAL [5], MeshLab [6], and OpenFlipper [7]. Their use, however, requires installation on the user's machine as well as special training. We discuss the advantages of using Java applet in Section 3.

The paper is organised as follows: In Section 2 we briefly describe the three algorithms. In Section 3 we provide detailed information about the main features of the package. In Section 4 we present the results from our experimental work and the comparison between the algorithms.

2. Mathematical background. *Subdivision surface* in 3D computer graphics is a polygonal mesh surface generated from an initial mesh through a recursive process that smooths the mesh while increases its density. We refer to the initial mesh as a *control* mesh. A control mesh consists of a set of *vertices*, a set of *edges*, and a set of *faces* plus incidence information defining their interrelations. The geometry of a mesh is defined by the coordinates of the vertices in 3D. A *subdivision scheme* consists of a set of rules for refinement and modification of the control mesh. The number of refinements (levels of subdivision) are controlled by user's requirements and the purposes of subdivision. In a limit subdivision schemes usually produce smooth surfaces with a possible exception of some vertices that are called *extraordinary*. In this paper we consider meshes with all their faces triangles. This is not a restriction since any polygonal mesh can be triangulated straightforwardly by adding edges. The extraordinary vertices for *triangle* meshes are all vertices of degree different from 6. Next we present briefly the subdivision rules for subdivision schemes \mathcal{L} , \mathcal{MB} , and \mathcal{K} . All meshes considered are closed triangle meshes, i. e. without boundary.

Scheme 1. Loop subdivision	
<i>Step 1.</i>	For each edge $e = (v_1, v_2)$ compute a new edge point $v_e = 3/8(v_1 + v_2) + 1/8(v_l + v_r)$, where v_l and v_r are the third vertices of the faces incident to the edge e .
<i>Step 2.</i>	For each vertex v compute a new vertex $v' = (1 - n\alpha_d)v + \alpha \sum_{j=1}^d v_j$, where v_1, \dots, v_d are the vertices adjacent to v and $\alpha_d = \frac{1}{d} \left(\frac{5}{8} - \left(\frac{3}{8} + \frac{1}{4} \cos \frac{2\pi}{d} \right)^2 \right)$.
<i>Step 3.</i>	Replace each face $f = (v_1, v_2, v_3)$ with four new faces $f_1 = (v'_1, v_{e_3}, v_{e_2})$, $f_2 = (v'_2, v_{e_1}, v_{e_3})$, $f_3 = (v'_3, v_{e_2}, v_{e_1})$, and $f_4 = (v_{e_1}, v_{e_2}, v_{e_3})$.

2.1. Loop subdivision. \mathcal{L} was proposed by LOOP [8] in 1987. The scheme is a generalisation of the binary subdivision of the three-directional quartic box splines. It works as follows, see Fig. 1.

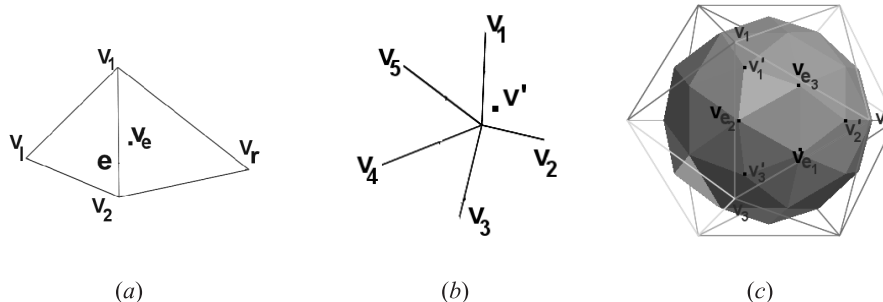


Fig. 1. (a) illustrates Step 1 of \mathcal{L} ; (b) illustrates Step 2 where new vertex v' is computed for a vertex v with $d = 5$; (c) illustrates Step 3 for input control mesh *icosahedron*

The limit surface generated by \mathcal{L} is C^2 -continuous except at extraordinary vertices where it is only C^1 -continuous. \mathcal{L} is an *approximating* scheme since the limit surface does not contain the vertices of the control mesh.

2.2. Modified Butterfly subdivision. Butterfly subdivision was initially proposed by DYN et al. [9] in 1990. The scheme is a generalisation of the 4-point curve subdivision. A tension parameter w is used to control how close is the limit surface to the control mesh. The obtained limit surface is C^1 -continuous except at extraordinary vertices of degree $d = 3$ and $d > 7$. An extension of the algorithm was proposed in 1993 by Dyn et al. [10]. However, the problem with the smoothness at the extraordinary vertices remained. In 1996, ZORIN et al. [11] presented an improved modification of the algorithm known as \mathcal{MB} . The scheme

for regular meshes is the same as in [10] but in addition the authors propose rules for the extraordinary vertices that guarantee C^1 -continuous limit surface for arbitrary triangle meshes. The scheme works as follows:

Scheme 2. Modified Butterfly subdivision	
<i>Step 1.</i>	For each edge $e = (v_1, v_2)$ with endpoints of degree 6 compute a new vertex $v_e = (1/2 - w)(v_1 + v_2) + (1/8 + 2w)(v_4 + v_8) + (-1/16 - w)(v_3 + v_5 + v_7 + v_9) + w(v_6 + v_{10})$, see Fig. 2(a)
<i>Step 2.</i>	For each edge $e = (v, v_1)$ with extraordinary vertex v and adjacent to v vertices v_1, \dots, v_d , compute new vertex $v_e = \sum_{j=1}^d \alpha_j v_j$ as follows, see Fig. 2(b) : $d \geq 5, \alpha_j = (1/4 + \cos \frac{2\pi(j-1)}{d} + 1/2 \cos \frac{4\pi(j-1)}{d})/d, j = 1, \dots, d;$ $d = 3, \alpha_1 = 5/12, \alpha_2 = \alpha_3 = 1/12;$ $d = 4, \alpha_1 = 3/8, \alpha_3 = -1/8, \alpha_1 = \alpha_3 = 0.$
<i>Step 3.</i>	For each edge $e = (v_1, v_2)$ with extraordinary vertices v_1 and v_2 compute two new vertices for each of them using Step 2. Then take its average as a new vertex v_e .
<i>Step 4.</i>	Replace each face $f = (v_1, v_2, v_3)$ with four new faces $f_1 = (v_1, v_{e_3}, v_{e_2}), f_2 = (v_2, v_{e_1}, v_{e_3}), f_3 = (v_3, v_{e_2}, v_{e_1}),$ and $f_4 = (v_{e_1}, v_{e_2}, v_{e_3})$, see Fig. 2(c).

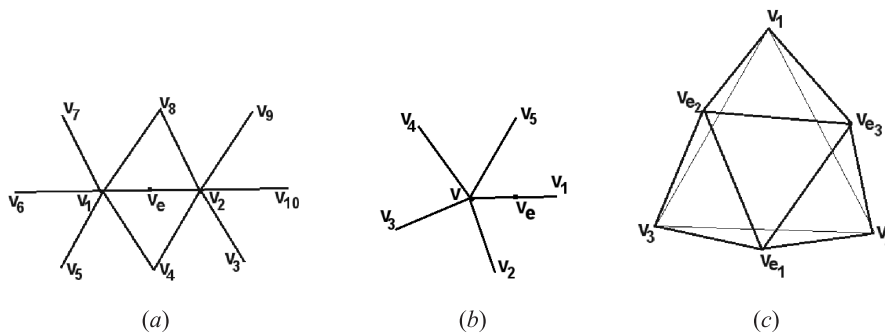


Fig. 2. (a) illustrates Step 1 of \mathcal{MB} where new vertex v_e is computed for an edge with endpoints of degree 6; (b) illustrates Step 2 where new vertex v_e is computed for an edge with extraordinary vertex v of degree $d = 5$ as endpoint; (c) illustrates Step 4 where the old face $f = (v_1, v_2, v_3)$ is replaced with four new faces

\mathcal{MB} is an *interpolating* scheme since the limit surface contains the vertices of the control mesh.

2.3. Kobbelt $\sqrt{3}$ -subdivision. Scheme \mathcal{K} was proposed by KOBBELT [12] in 2000. It works in three steps, see Fig. 3.

Scheme 3. Kobbelt $\sqrt{3}$ -subdivision	
<i>Step 1.</i>	For each face $f = (v_1, v_2, v_3)$ compute its center $m_f = \frac{1}{3}(v_1 + v_2 + v_3)$ and create three new faces $f_1 = (v_2, v_3, m_f)$, $f_2 = (v_3, v_1, m_f)$, and $f_3 = (v_1, v_2, m_f)$, see Fig. 3(a).
<i>Step 2.</i>	For each vertex v compute new vertex $v' = (1 - \alpha_d)v + \frac{\alpha_d}{d} \sum_{j=1}^d v_j$, where v_1, \dots, v_d are vertices adjacent to v and $\alpha_d = \frac{1}{9} \left(4 - 2 \cos \frac{2\pi}{d} \right)$. Replace v with v' .
<i>Step 3.</i>	Replace each old edge with the edge between the centers of the adjacent triangles, see Fig. 3(b), 3(c).

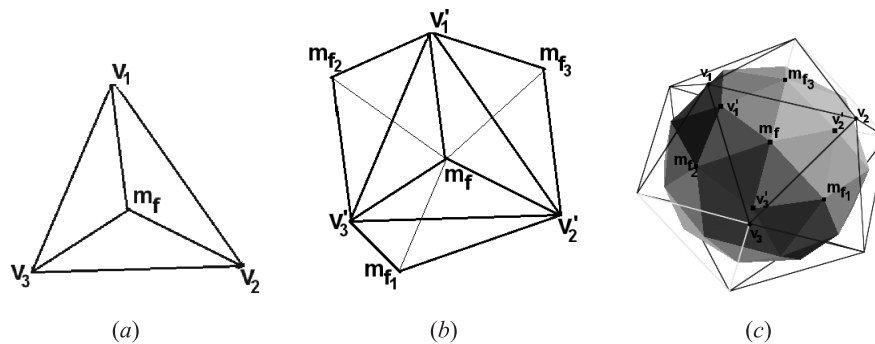


Fig. 3. (a) illustrates Step 1 of \mathcal{K} ; (b) illustrates Step 3; (c) illustrates Step 3 for input control mesh *icosahedron*

The limit surface is C^2 -continuous except at extraordinary vertices where it is C^1 -continuous. \mathcal{K} is an approximating scheme.

3. Overview of our work. We implemented a program package for interactive 3D visualisation and comparison of \mathcal{L} , \mathcal{MB} , and \mathcal{K} subdivision schemes as an interactive Java applet using Java 3D graphics library. Based on extensive experimentation we analysed the behaviour of the three considered schemes for various closed triangle meshes of arbitrary topology.

3.1. Why Java applet. *Java applet* is a compiled Java program runned by web browser. When activated the executable file is downloaded automatically from a web server to the user's system. Then it is executed by a Java Virtual Machine (JVM).

Our choice of implementation and visualisation tools is motivated by the following main advantages of using Java applets.

Java applets are supported by most web browsers and they are platform independent. Java applets can be executed in all popular OS including Microsoft Windows, Unix, Mac OS and Linux. Thus the distribution of a program through the Internet is easier.

Most web browsers cache applets, so when returning to a previously visited web page, the applet loads quickly. Moreover, applet's performance improves with multiple executions within one session of the browser. The only condition for running the applet is that the user's system must have Java plug-in and Java 3D installed which is easy since Java is freeware. This is an advantage over using similar visualisation packages available on the Internet. For example, using Matlab requires installed Matlab software and moreover the user has to be familiar with Matlab programming.

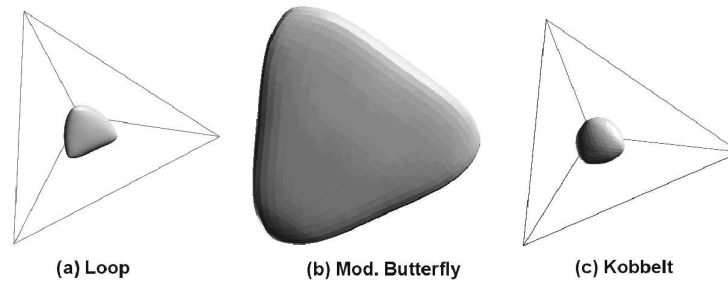
A disadvantage is that despite the use of Java archive (jar) files, the jar file can still be too large which significantly slows down its downloading and execution. This is one of the reasons why we didn't include in our Java applet more triangular subdivision algorithms.

3.2. Description of the package. The main features of our package are:

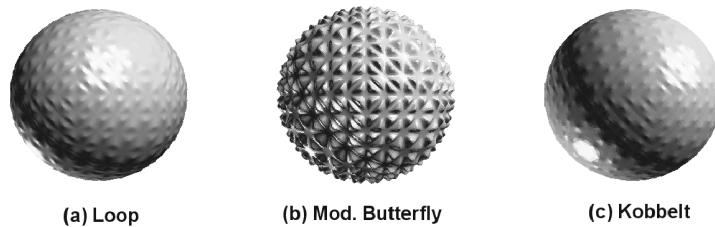
- The initial mesh can be chosen either from four drop-down lists, or can be loaded from an external file since our applet is signed to allow access to the user's local file system. Most of the preinstalled meshes on the lists are downloaded from [13–16].
- The package works with closed polygonal meshes of arbitrary topology. In case the input mesh is not a triangle mesh, the program automatically triangulates it using the Java 3D class *Triangulator* and then proceeds.
- The level of subdivision can be controlled manually.
- The applet visualises meshes in *Wireframe* and *Shaded* modes with additional options *Show All Steps* and *Show Initial Mesh*. *Wireframe* mode is more convenient to demonstrate and analyse the subdivision process while *Shaded* is appropriate to study and compare the shape and the smoothness of the obtained surfaces.
- The surfaces can be edited interactively by rotation, translation and resizing.
- An additional slider *Surface tension* allows the adjustment of the tension parameter w for \mathcal{MB} in the interval $[-1, 1]$.
- The generated mesh at each subdivision level can be exported as .obj file. So the user has the possibility to choose which algorithm to use at each subdivision level and even to switch between different algorithms during the creation of a single complex surface.

The package is accessible through the applet and can be tested at the Internet address [17]. A detailed user guide including complete description of package features and their control as well as results from our experimental work can be found in [18].

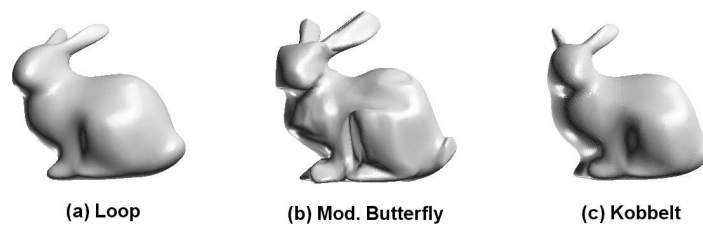
4. Experimental work and comparison of the algorithms. Using our package we have compared the behaviour of the algorithms with respect to various criteria. Below we discuss our observations and conclusions.



A: Control mesh *tetrahedron* after 5 subdivision steps



B: Control mesh *epcot* after 3 subdivision steps



C: Control mesh *bunny* after 4 subdivision steps

Fig. 4. Various meshes with increasing complexity

Type of scheme: interpolating or approximating. \mathcal{L} and \mathcal{K} are *approximating* schemes while \mathcal{MB} is *interpolating*. There are substantial differences between the shape of the surfaces obtained with these two types of schemes. The approximating schemes shrink the surface during the subdivision. The shrinkage effect is quite obvious in simple models with small number of faces, e.g. tetrahedron or octahedron, see Fig. 4A.

The approximating schemes produce visually smooth, good looking surfaces and for this reason they are generally the preferable modelling tool in computer animation. Even if the control mesh is dense and with many creases, the scheme

has a tendency to smooth them out, see Fig. 4Ba, 4Bc. The main drawback of the approximating schemes is that the user can not foresee the shape of the limit surface by looking at the control mesh. In contrast, the major benefit of the interpolating schemes is that given a control mesh, it is easy to control the shape of the limit surface since it remains close to the original mesh. A drawback is the appearing unpleasant undulation, rippling or bulging especially for control meshes with sharp features, see Fig. 4Bb. On the other hand, the interpolating schemes preserve more details and sharp features of the model, see Fig. 4Cb.

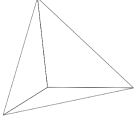
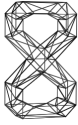


Computational costs. After one level of subdivision the number of faces using \mathcal{L} and \mathcal{MB} increases four times while using \mathcal{K} the number of faces increases three times. So one subdivision step of \mathcal{K} is usually faster compared to one step of \mathcal{L} and \mathcal{MB} . On the other hand, \mathcal{K} produces larger faces and the refined mesh has a poorer quality. But the slower increase of the number of faces allows better control of the details level and makes \mathcal{K} suitable for adaptive subdivision. Our experiments were performed on a Windows 7 system with IntelCore i7-3630 CPU and 2.4-GHz processor, 12-GB RAM. The times to perform one level of subdivision of \mathcal{L} , \mathcal{MB} , and \mathcal{K} on a mesh with 200 faces were .0822 ms, .3834 ms, and .0599 ms, respectively. The difference in the speeds increases with the increase of the subdivision level and mesh complexity, see Table 1.

Regularity of the generated meshes. Regular meshes consisting of good quality triangles are preferable in most applications. The quality of a triangle can be measured by the aspect ratio r/R where r and R denote the radii of the inscribed and circumscribed circles, respectively. The *regularity* of a mesh is the minimal aspect ratio of all triangles. We have compared numerically the quality of the meshes produced by the three algorithms. We tracked the number of equilateral triangles, regularity, and the number of bad triangles. The computational results for meshes of increasing complexity *tetrahedron*, *double torus*, *bunny*, and *king* are presented on Table 1. The experiments show that the percentage of the equilateral triangles of maximum aspect ratio $1/2$ is highest for meshes generated by \mathcal{L} and lowest for meshes generated by \mathcal{MB} . The regularity is lowest for meshes generated by \mathcal{MB} . With the increase of the subdivision level and mesh complexity the regularity for \mathcal{L} is getting better than the regularity for \mathcal{K} , see Table 1. A visual comparison of the regularity was created using Meshlab [6] and can be seen in [18].

5. Conclusions and future work. We have implemented a large number of experiments on meshes of increasing complexity and analysed the results with respect to various criteria. Our comparison results can be useful in deciding which of these three algorithms is best suited for specific applications. Currently, our package works with closed meshes only. Our intention is to further enhance the package to work with arbitrary polygonal meshes and to compare numerically other important features of the generated meshes.

T a b l e 1

Comparison of percentage of equilateral triangles, regularity (minimal aspect ratio), number of bad triangles, and computational costs in ms of \mathcal{L} , \mathcal{MB} , and \mathcal{K} for meshes of increasing complexity: *tetrahedron*, *double torus*, *bunny*, and *king*

Mesh	Alg.	Subd. level	# of \triangle	% of eq. \triangle	Regularity	# of bad \triangle	Comp. costs
		0	4	25	.4142	3	
	L.	1	16	6.25	.2511	6	.0075
		5	4 096	31.45	.0938	3	.3329
	M.B.	1	16	6.25	.3293	3	.0143
		5	4 096	18.70	.1132	3	.3365
	$\sqrt{3}$	1	12	50	.1800	3	.0108
		5	972	31.48	.0958	1	.0866
		0	116	0	.1360	4	
	L.	1	1 856	1.19	.2069	2	.0500
		5	118 784	10.89	.2207	1	12.9876
	M.B.	1	464	1.72	.1491	2	.1101
		5	118 784	7.09	.0913	1	13.0863
	$\sqrt{3}$	1	348	9.77	.1956	1	.0344
		5	28 188	9.67	.2253	1	2.2822
		0	200	12.5	.0255	1	
	L.	1	800	25	.0091	1	.0822
		5	204 800	28.32	.0283	1	25.7828
	M.B.	1	800	12.38	.0145	1	.3834
		5	204 800	13.02	.0007	1	29.7751
	$\sqrt{3}$	1	600	21.67	.0098	1	.0599
		5	48 600	27.28	.0286	1	4.3842
		0	624	8.97	.0036	8	
	L.	1	2 496	7.05	.0050	4	.2319
		5	638 976	13.66	.0206	4	76.7748
	M.B.	1	2 496	5.77	.0470	4	.2869
		5	638 976	4.38	.0000	4	102.5647
	$\sqrt{3}$	1	1 872	7.05	.0024	8	.1723
		5	151 632	12.47	.0021	4	54.4312

Acknowledgements. We thank Hristo Tonchev for creating some of the test models.

REFERENCES

- [1] FARIN G., J. HOSCHEK, M.-S. KIM (Eds.). Handbook of Computer Aided Geometric Design, Amsterdam, Elsevier, North-Holland, 2002.
- [2] WARREN J., H. WEIMER. Subdivision Methods for Geometric Design: A Constructive Approach, San Francisco, Morgan Kaufmann, 2002.
- [3] PETERS J., U. REIF. Subdivision Surfaces, Berlin, Springer, 2008.
- [4] ANDERSSON L.-E., N. STEWART. Introduction to the Mathematics of Subdivision Surfaces, Philadelphia, SIAM, 2010.
- [5] CGAL, Computational Geometry Algorithms Library, <http://www.cgal.org>
- [6] Meshlab, Visual Computing Lab-ISTI-CNR, <http://meshlab.sourceforge.net>
- [7] OpenFlipper, <http://www.openflipper.org>
- [8] LOOP C. T. Smooth Subdivision Surfaces Based on Triangles. Master's thesis, University of Utah, Department of Mathematics, 1987.
- [9] DYN N., D. LEVIN, J. A. GREGORY. ACM Trans. on Graphics, **9**, 1990, No 2, 160–169.
- [10] DYN N., S. HED, D. LEVIN. In: Workshop on Computational Geometry (eds A. Conte et al.), Singapore, World Scientific, 1993, 97–118.
- [11] ZORIN D., P. SCHRÖDER, W. SWELDENS. In: SIGGRAPH'96, 23rd Int. Conf. on Computer Graphics and Interactive Techniques, New York, ACM Press, 1996, 189–192.
- [12] KOBBELT L. In: SIGGRAPH'00, 27th Int. Conf. on Computer Graphics and Interactive Techniques, New York, ACM Press, 2000, 103–112.
- [13] <http://www.the3dstudio.com>
- [14] <http://people.sc.fsu.edu/~jburkardt/data/data.html>
- [15] <http://www.holmes3d.net/graphics/offfiles>
- [16] <http://graphics.stanford.edu/data/3Dscanrep>
- [17] <http://www.fmi.uni-sofia.bg/fmi/companal/krassivl/TodorkaHalacheva/Subdivision.html>
- [18] <http://www.fmi.uni-sofia.bg/fmi/companal/krassivl/TodorkaHalacheva/UserGuide.pdf>

*Faculty of Mathematics and Informatics
Sofia University St. Kliment Ohridski
5, J. Bourchier Blvd
1164 Sofia, Bulgaria
e-mail: krassivl@fmi.uni-sofia.bg
todorka.halacheva@gmail.com*