

A comparison of surface subdivision algorithms for polygonal meshes

K. Vlachkova and P. Terziev

Citation: *AIP Conf. Proc.* **1487**, 343 (2012); doi: 10.1063/1.4758977

View online: <http://dx.doi.org/10.1063/1.4758977>

View Table of Contents: <http://proceedings.aip.org/dbt/dbt.jsp?KEY=APCPCS&Volume=1487&Issue=1>

Published by the [American Institute of Physics](#).

Additional information on AIP Conf. Proc.

Journal Homepage: <http://proceedings.aip.org/>

Journal Information: http://proceedings.aip.org/about/about_the_proceedings

Top downloads: http://proceedings.aip.org/dbt/most_downloaded.jsp?KEY=APCPCS

Information for Authors: http://proceedings.aip.org/authors/information_for_authors

ADVERTISEMENT



AIP Advances

Submit Now

Explore AIP's new
open-access journal

- Article-level metrics now available
- Join the conversation! Rate & comment on articles

A Comparison of Surface Subdivision Algorithms for Polygonal Meshes

K. Vlachkova and P. Terziev

*Faculty of Mathematics and Informatics, St. Kliment Ohridski University of Sofia, 5 James Bourchier Blvd.,
1164 Sofia, Bulgaria*

Abstract. We present a new program package for interactive implementation and 3D visualization of three fundamental algorithms for surface subdivision. Namely, these are Doo-Sabin algorithm, Catmull-Clark algorithm, and Peters-Reif algorithm. Our work and contributions are in the field of experimental algorithmics and algorithm engineering. We have chosen OpenGL and Qt graphics libraries as our main implementation and visualization tools. Our program analyzes the validity of the loaded mesh and proceeds with valid meshes only. We provide a user friendly interface so that users can load their own data sets. The latter allows wide testing and comparing the results from the implementation of the three algorithms on arbitrary polygonal meshes. The program has also an option for creating new polygonal meshes. We experimented extensively with our package. We compared the behaviour of the three algorithms based on different criteria and using meshes of increasing complexity. The experimental results are presented and analyzed.

Keywords: surface subdivision, polygonal mesh, OpenGL

PACS: 07.05.Tp, 45.10.Na, 02.60.Gf, 89.20.Ff

INTRODUCTION

Subdivision algorithms are one of the most successful modern techniques for modeling smooth free-form shapes, for a detailed discussion see [1, 2, 3, 4]. They allow simple and efficient construction of smooth surfaces of arbitrary topology starting from an initial polygonal *control mesh*. The algorithms 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 is obtained as a limit of a recursive process of subdivision. Because of their simplicity and efficiency, and the ability to obtain high quality images, the subdivision algorithms have applications in various fields: Computer Aided Geometric Design, computer graphics, solid modeling, computer game software, computer animation, aerospace and automotive design, industrial design, architecture, medicine, *etc.* One of the promising applications is in solving problems that arise in applied physics. For example, subdivision surfaces can be used in simulation of thin shells and plates, see [5, 6, 7]. Another important application is in computer animation. A number of commercial 3D animation software packages use subdivision for surface modeling including Autodesk Maya, Pixar's RenderMan and Marionette, NewTek's LightWave 3D, and others.

In this paper we study three of the most popular and used in industry subdivision algorithms that work with arbitrary polygonal meshes: Doo-Sabin algorithm [8], Catmull-Clark algorithm [9], and Peters-Reif algorithm [10]. The main contributions of our work can be summarized as follows:

- We implemented a new software package for interactive visualization, manipulation and comparison of the subdivision surfaces generated by these three algorithms. The package is user friendly and has a large number of features that allow easy testing and experimenting.
- We tested the package and implemented extensive experimentation. The results were analyzed to reveal main differences between the tested algorithms with respect to different criteria including quality, performance, and efficiency.
- We made our package public so that it can be used for educational purposes.

The paper is organized as follows. Section 2 gives a brief description of 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.

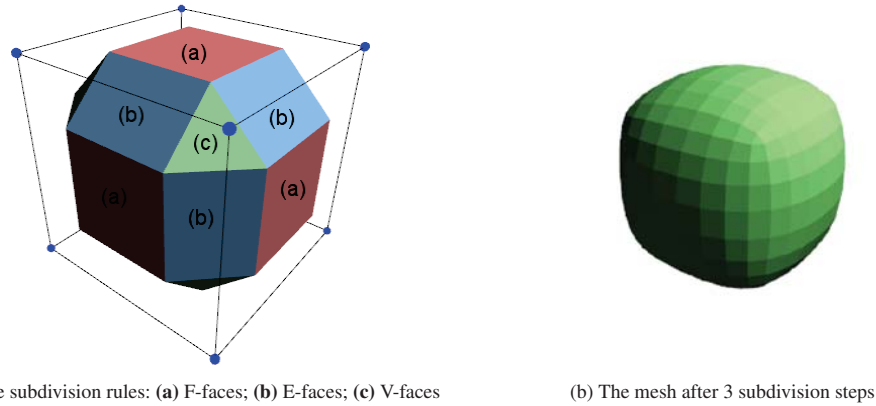


FIGURE 1. Doo-Sabin subdivision for a **cube** as a control mesh

MATHEMATICAL BACKGROUND

A *polygonal mesh* consists of a set of vertices, edges, faces and topological relations between them. A *subdivision scheme* is defined by a set of rules for mesh refinement. Below we present briefly the subdivision rules for Doo-Sabin, Catmull-Clark, and Peters-Reif algorithms. Detailed information about the history and development of the subdivision algorithms can be found, *e.g.*, in [1].

Doo-Sabin subdivision

Doo-Sabin algorithm is proposed by D. Doo and M. Sabin in 1978. This algorithm is a generalization of the Chaikin's corner cutting subdivision algorithm which produces uniform C^1 -quadratic B-spline curves in the limit. For *regular* control meshes - all faces are quadrilaterals, Doo-Sabin subdivision algorithm generates uniform quadratic B-spline surfaces. For irregular closed polygonal meshes of arbitrary topology the algorithm works as follows:

1. For each face with vertices v_i , $i = 1, \dots, n$, where $n \in \mathbf{N}$ is the number of all vertices defining the face, the new vertices v_i^1 , $i = 1, \dots, n$ are computed as an affine combination of the original vertices,

$$v_i^1 = \sum_{j=1}^n a_{ij} v_j, \quad \text{where} \quad a_{ij} = \begin{cases} \frac{n+5}{4n} & \text{for } i = j \\ \frac{1}{4n} \left(3 + 2 \cos \frac{2\pi(i-j)}{n} \right) & \text{otherwise.} \end{cases}$$

2. New faces are constructed by the following rules (see Fig. 1a):
 - (a) *F-faces* are generated by connecting consecutively $v_1^1, v_2^1, \dots, v_n^1$ for every face of the control mesh. The new mesh and the control mesh have equal number of the vertices.
 - (b) *E-faces* are generated using two neighbor faces sharing a common edge. The E-faces are always quadrilaterals.
 - (c) *V-faces* are generated using the E-faces of all edges that start from a common vertex. The number of the new faces is equal to the *degree* of the corresponding vertex., *i.e.*, the number of the edges that start from this vertex.

The generated limit surface is C^1 -continuous. Figure 1 shows an example of Doo-Sabin algorithm's work on a **cube** as a control mesh.

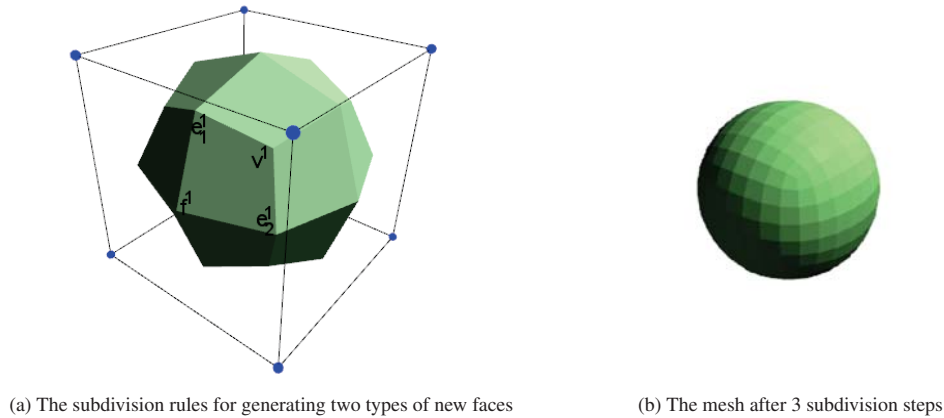


FIGURE 2. Catmull-Clark subdivision for a **cube** as a control mesh

Catmull-Clark subdivision

Catmull-Clark algorithm is proposed by E. Catmull and J. Clark in 1978 and is a generalization of the uniform C^2 -cubic B-spline curves. For regular control meshes, Catmull-Clark subdivision algorithm generates uniform cubic B-spline surfaces. For irregular closed polygonal meshes of arbitrary topology the algorithm works as follows (see Fig. 2a):

1. Compute the *f-point* for each face. This is the centroid of the face, *i.e.*, the average of all vertices defining the face.
2. Compute the *e-point* for each edge. This is the average of the two endpoints of the edge and the two *f*-points of the faces sharing the edge.
3. Compute the *v-point* v^1 for each vertex v as follows:

$$v^1 = \frac{1}{n}Q + \frac{2}{n}R + \frac{n-3}{n}v,$$

where Q is the average of the *f*-points of all faces surrounding v , R is the average of the midpoints of all edges with endpoint v , and n is the degree of v .

4. The new faces are generated. The vertices of each new face form a sequence of the following type

$$v^1 \rightarrow e_1^1 \rightarrow f^1 \rightarrow e_2^1 \rightarrow v^1,$$

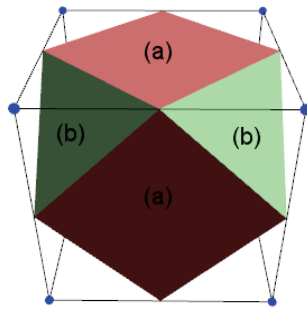
where the *e*-points e_1^1 and e_2^1 refer to the two neighboring edges with endpoint v^1 , which belong to the face with *f*-point f^1 (see Fig. 2a). Note that all faces after the first subdivision level are quadrilaterals.

Figure 2 shows an example of Catmull-Clark algorithm's work on a **cube** as a control mesh. The generated limit surface is C^2 -continuous except at *extraordinary* points - of degree that differs from 4. At such points the limit surface is only C^1 -continuous.

Peters-Reif subdivision

Peters-Reif algorithm is proposed in 1997 by J. Peters and U. Reif who called it "the simplest scheme" because of its extremely simple subdivision rules. For any control mesh, in the limit the algorithm produces regular planar polygons. They are tangent planes to the limit surface which is C^1 -continuous. The algorithm works as follows:

1. Compute the new vertices. These are the midpoints of all edges.
2. Form the new faces. They are of two types (see Figure 3a):
 - (a) Face inscribed to an old face. Its vertices are the midpoints of the edges of the old face.

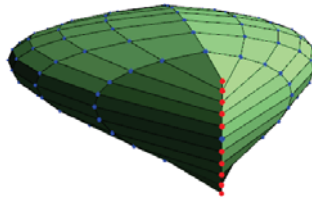


(a) The subdivision rules for generating new faces

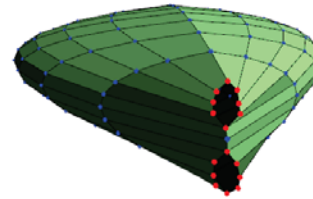


(b) The mesh after 3 subdivision steps

FIGURE 3. Peters-Reif subdivision for a **cube** as a control mesh



A mesh that looks closed...



...but in fact is not: it has coincident vertices.

FIGURE 4. An invalid mesh that looks closed

(b) Face formed by the midpoints of the edges that are incident to a fixed vertex.

Figure 3 shows an example of Peters-Reif algorithm's work on a **cube** as a control mesh.

DESCRIPTION OF THE PACKAGE

We implemented a program package for interactive 3D visualization and comparison of the three considered algorithms. Our main purpose was to visualize, compare, and analyze the behavior of Doo-Sabin, Catmull-Clark, and Peters-Reif subdivision algorithms for different closed polygonal meshes of arbitrary topology. In addition the package was used for evaluation of the efficiency of the three algorithms.

The program uses the multiplatform graphics libraries OpenGL [11, 12] and Qt [13] as our main implementation and visualization tools. OpenGL is probably the most used in industry application programming interface for computer graphics. This is due to its wide accessibility and compatibility with different operating systems and computer platforms.

The user interface is intuitive and allows changing of the main visualization elements as lighting, shading, coloring, *etc.* The main features of our package are:

- The control mesh can be chosen either from a drop-down list, or can be loaded from an external file. Most of the polygonal meshes on the list are downloaded from the Internet site [14].
- The level of subdivision can be controlled. Moreover, after the initial mesh is chosen, a dialog window showing the required memory allocations appears so that the user can estimate a priori the maximum subdivision level.
- The program analyzes the loaded mesh for correctness and proceeds with a valid mesh only. Otherwise the incorrect vertices are marked in red. Note that a mesh that is not closed is recognized as invalid. An example of invalid mesh is shown in Figure 4.

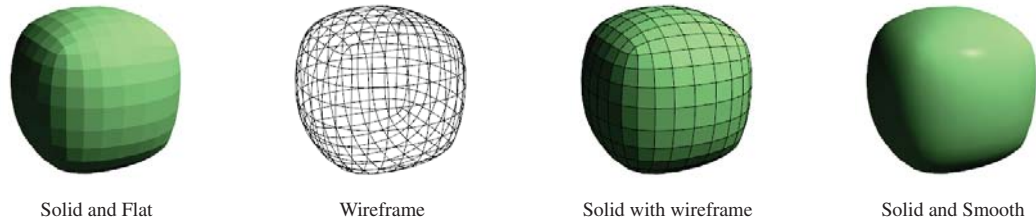


FIGURE 5. The different visualization modes

TABLE 1. Comparison of Doo-Sabin, Catmull-Clark, and Peters-Reif subdivision algorithms using different criteria.

Criterion	Doo-Sabin	Catmull-Clark	Peters-Reif
Stationary scheme	yes	yes	yes
Approximating scheme	yes	yes	yes
Primal (face split)	no	yes	no
Dual (vertex split)	yes	no	yes
Smoothness	C^1	C^2	C^1

Number of vertices, edges, and faces for a mesh of genus g			
Initial number	After one step		
v (vertices)	$2e$	$2e+2-2g$	e
e (edges)	$4e$	$4e$	$2e$
f (faces)	$2e+2-2g$	$2e$	$e+2-2g$

- The program visualizes the meshes in three modes: *Solid*, *Wireframe*, and *Solid with wireframe*, with options for visual adjustments of the color, points size and mesh thickness. *Wireframe* mode is more convenient to demonstrate and analyze the subdivision process while *Solid* and *Solid with wireframe* modes are appropriate to study and compare the shape and the smoothness of the obtained surfaces. *Solid* and *Solid with wireframe* have also an additional option for a choice between two light types: *Flat* and *Smooth*, see Figure 5.
- The surfaces can be edited interactively by rotation, translation and resizing using the mouse with option for synchronizing the transformations between different views.
- The mesh can be visualized using one, two or all three algorithms simultaneously on the screen with an option for showing the next step vertices, edges, and faces which allow better visual comparison of the results.
- The current mesh can be exported as a file. This option, together with the interactivity of the vertices, can be used to create new meshes. For example, we can apply first Doo-Sabin algorithm, then move some of the vertices, and then apply Catmull-Clark algorithm.

The package is available and can be downloaded at the Internet address http://www.fmi.uni-sofia.bg/fmi/companal/krassivl/Plamen_Terzиеv/Subdivision.zip.

EXPERIMENTAL WORK AND COMPARISON OF THE ALGORITHMS

We compared the behaviour of Doo-Sabin, Catmull-Clark, and Peters-Reif subdivision algorithms using different criteria. The results are presented in Table 1. Below we discuss our observations and conclusions with respect to each of the criteria.

Type of subdivision rules: *primal (face split) or dual (vertex split)*. All three algorithms are based on *stationary* subdivision schemes, *i.e.*, the subdivision rules do not change during the subdivision process. The algorithms differ according to which elements of the mesh are split. With the *face split* rule at each subdivision step a new edge point is added for every edge and every old face is subdivided to new faces of the same type. With the *vertex split* rule at each step every vertex is replaced by a new face. Subdivision schemes using the face split rule are *primal*, while schemes using the vertex split rule are *dual*. Catmull-Clark algorithm is of a primal type (face split). Doo-Sabin and Peters-Reif

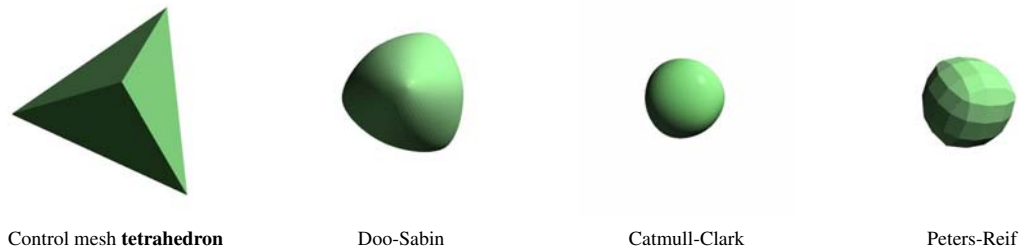


FIGURE 6. Control mesh **tetrahedron** after 5 subdivision steps

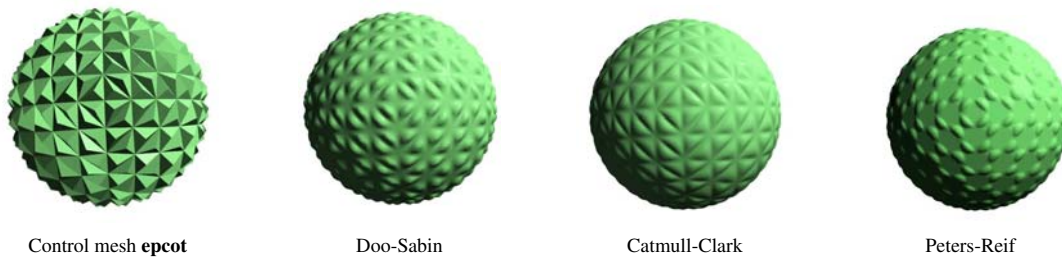


FIGURE 7. Control mesh **epcot** after 5 subdivision steps

algorithms are of a dual type (vertex split). Note that the mesh generated after one step of Doo-Sabin algorithm is dual of the mesh generated after one step of Catmull-Clark algorithm.

Interpolating or approximating scheme. The scheme is *interpolating* if the limit surface passes through the initial *control points*. Otherwise the scheme is *approximating*. Doo-Sabin, Catmull-Clark, and Peters-Reif are approximating schemes. The approximating surfaces shrink during the subdivision. The shrinkage effect is quite obvious in simple models with small number of faces, *e.g.*, tetrahedron or octahedron (see Figure 6). The approximating schemes produce visually smooth, good looking surfaces and for this reason they are generally the preferable modeling 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 Figure 7). A drawback of the approximating schemes is that the user can not foresee the shape of the limit surface simply by looking at the control mesh.

Smoothness of the limit surface. The limit surfaces generated by Catmull-Clark subdivision are C^2 -continuous everywhere except at the extraordinary points where they are C^1 -continuous. The limit surfaces generated using Doo-Sabin and Peters-Reif subdivision are only C^1 -continuous. Hence Catmull-Clark algorithm generates surfaces that look visually more pleasant than the surfaces generated with Doo-Sabin and Peters-Reif algorithms which often produce unevenness and creases (see Figure 8). Moreover, the algorithms produce visually unpleasant surfaces if the mesh is triangulated first. In Figure 9 are shown the surfaces generated after 5 subdivision steps of Catmull-Clark algorithm applied on quad and triangulated model of digit 6, respectively.

Computational complexity. After one level of subdivision the number of the faces using Doo-Sabin and Catmull-Clark algorithms increases approximately four times while using Peters-Reif algorithm the number of the faces increases approximately twice. Hence one subdivision step of Peters-Reif algorithm is usually faster compared to one step of Doo-Sabin and Catmull-Clark algorithms. On the other hand, Peters-Reif algorithm produces larger faces and the refined mesh has a poorer quality.

Efficiency. Our experiments were performed on a Windows7 system with IntelCore i7-940 2.93-GHz processor (12-GB RAM) and NVidia Quadro FX 3800 (16 GB RAM) GPU. The computational results of the surface approximation for meshes of increasing complexity (**tetrahedron**, **cube**, **star**, and **epcot**) are presented in Table 2. The experiments show that Peters-Reif algorithm is faster than Doo-Sabin and Catmull-Clark algorithms. Doo-Sabin and Catmull-Clark have almost the same speed. Catmull-Clark is a little slower than Doo-Sabin. The difference in the speeds increases with the increasing of the subdivision level and mesh complexity, see Table 2.

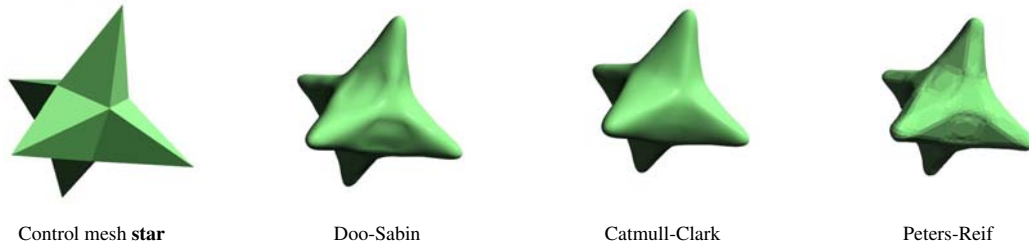


FIGURE 8. Control mesh **star** after 5 subdivision steps

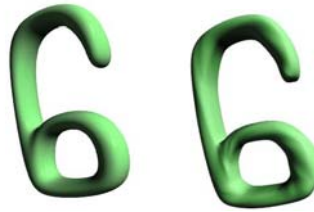






FIGURE 9. The algorithms produce visually unpleasant surfaces if the mesh is triangulated first: Catmull-Clark after 5 subdivision steps for quad and triangulated model of digit 6

TABLE 2. Comparison of the computational costs of Doo-Sabin, Catmull-Clark, and Peters-Reif subdivision algorithms for meshes of increasing complexity: **tetrahedron**, **cube**, **star**, and **epcot**.

Computational costs in ms				
Control mesh	Subd. level	Doo-Sabin	Catmull-Clark	Peters-Reif
	1	0.01885	0.02100	0.01169
	2	0.05648	0.05896	0.01770
	3	0.15188	0.15724	0.02791
	4	0.49387	0.49838	0.05173
	5	2.05886	2.04334	0.088203
	total	2.77994	2.77892	0.19724
	1	0.03111	0.03408	0.01702
	2	0.09275	0.09398	0.02908
	3	0.26746	0.26491	0.05083
	4	0.97538	1.00970	0.09060
	5	4.16854	4.00928	0.14027
	total	5.53523	5.41195	0.32779
	1	0.21027	0.22752	0.11856
	2	0.77094	0.76089	0.19251
	3	3.09434	3.00573	0.35888
	4	12.7053	12.0062	0.69687
	5	67.9744	55.9502	1.42880
	total	84.7553	71.95050	2.79561
	1	3.07818	2.97268	1.44242
	2	12.3815	11.9505	2.87408
	3	65.7667	56.0691	5.53061
	4	318.758	248.859	11.5932
	5	1327.72	980.746	23.9201
	total	1727.70	1300.60	45.3604

CONCLUSIONS AND FUTURE WORK

In this paper we presented a program package for interactive implementation and 3D visualization of three fundamental algorithms for surface subdivision: Doo-Sabin algorithm, Catmull-Clark algorithm, and Peters-Reif algorithm. The package is made user friendly and possesses a wide range of features for experimenting with the three algorithms. We performed a large number of experiments on meshes of increasing complexity and analyzed the results with respect to different criteria. Given the importance of surface modeling and simulation techniques in practice, it is important to better understand and engineer software packages based on different algorithms for surface manipulation. The work presented here is an initial step in this direction. Our intention is to further enlarge the package by adding new subdivision algorithms and new features to it. A challenging idea is to add an automatic mode to the package that would be able to choose a particular algorithm or even a combination of algorithms depending on the user defined criteria.

ACKNOWLEDGMENTS

This work was partially supported by Sofia University Science Fund Grant No. 132/2012. The experiments were performed at the Centre for Simulation, Business Processes and 3D Visualization (SimPro) in Sofia University.

REFERENCES

1. M. Sabin, "Subdivision surfaces," in *Handbook of Computer Aided Geometric Design*, edited by G. Farin, J. Hoschek, and M.-S. Kim, Elsevier, North-Holland, 2002, pp. 309–325.
2. J. Warren, and H. Weimer, *Subdivision Methods for Geometric Design: A Constructive Approach*, Morgan Kaufmann, San Francisco, 2002.
3. J. Peters, and U. Reif, *Subdivision Surfaces*, Springer, Berlin, 2008.
4. L.-E. Andersson, and N. Stewart, *Introduction to the Mathematics of Subdivision Surfaces*, SIAM, Philadelphia, 2010.
5. F. Cirak, M. Ortiz, and P. Schröder (2000) *International Journal for Numerical Methods in Engineering* **47**(12), 2039–2072.
6. F. Cirak, and Q. Long, "Advances in subdivision finite elements for thin shells," in *New Trends in Thin Structures: Formulation, Optimization and Coupled Problems*, edited by P. de Mattos Pimenta, and P. Wriggers, Springer, Wien-New York, 2010, pp. 205–227.
7. S. Green, G. Turkiyyah, and D. Storti, D., "Subdivision-based multilevel methods for large scale engineering simulation of thin shells," edited by K. Lee, and N. M. Patrikalakis, *Proceedings of the 7th ACM Symposium on Modeling and Application (SM-02)*, ACM Press, New York, 2002, pp. 265–272.
8. D. Doo, and M. Sabin (1978) *Computer-Aided Design*, **10**(6), 356–360.
9. E. Catmull, and J. Clark (1978) *Computer-Aided Design* **10**(6), 350–355.
10. J. Peters, and U. Reif (1997) *ACM Trans. Graph.*, **16**(4), 420–431.
11. OpenGL, <http://www.opengl.org>
12. D. Shreiner, and The Khronos OpenGL ARB Working Group, *OpenGL Programming Guide (The Red Book), 7th edn, The Official Guide to Learning OpenGL Version 3.0 and 3.1*, Addison-Wesley, Boston, 2010
13. Qt, <http://qt-project.org>
14. .OFF files, <http://www.holmes3d.net/graphics/offfiles>