BULGARIAN ACADEMY OF SCIENCES

CYBERNETICS AND INFORMATION TECHNOLOGIES • Volume **13**, No 3

Sofia • 2013 Print ISSN: 1311-9702; Online ISSN: 1314-4081
DOI: 10.2478/cait-2013-0023

An Efficient Data Structure for Representing Trilateral/Quadrilateral Subdivision Surfaces

Jianning Zhu, Minjie Wang[1]*, Zhaocheng Wei, Bin Cao*

School of Mechanical Engineering, Dalian University of Technology, Dalian 116024, Liaoning, China
Emails: 47554159@qq.com mjwang@dlut.edu.cn zhaocheng_wei@foxmail.com 2046@dlut.edu.cn

Abstract: *With the increase of subdivision depths, some problems of common data structures for representing the subdivision surfaces appear, such as excessive computer memory consumption and low efficiency of the data query which restrict the popularization and application of subdivision surfaces in more fields. By utilizing the topological characteristics of the subdivision surface, a two-layer data structure named CELL is presented in order to better realize the piecewise representation of trilateral/quadrilateral subdivision surfaces. The inner structure of CELL represents the subdivision surface patch by using arrays, and the outer structure of CELL represents the topological relations between the subdivision surface patches. Based on Catmull-Clark subdivision scheme, the structural compositions of CELL and the realization mechanism of the subdivision algorithm are proposed. Additionally, sharp and semi-sharp features are constructed, and a primary study on amalgamation of the image/Z-map model and subdivision surface is presented. The results of the experimental and theoretical analysis show the superior performance of CELL with relation to computer memory consumption, data query, subdivision surface computation and algorithm development.*

Keywords: *Geometric modeling, subdivision surface, data structure, mesh compression.*

[1] Corresponding author

1. Introduction

The subdivision surface is a very popular geometric modelling method, which can be used to create smooth surfaces with arbitrary topological structure that has the advantages of strong numerical stability and high execution efficiency. It has been successfully applied to computer graphics, film & TV, animation and other fields. Appropriate data structure is very important for the geometric representation of the subdivision surface, implementation of the subdivision algorithm and development of the application algorithm. The ideal data structure must have the following features: efficient data query, less computer memory consumption and compact topological structure. At present the common data structures of the subdivision surface can meet the application requirements in most of the fields. However, with respect to the industries requiring higher subdivision depth, such as numerical control machining, the rapid increase of the computer memory consumption and the low search efficiency have become the bottleneck of the industrial application of the subdivision surface. Therefore the study on an appropriate data structure of the subdivision surface is of great significance.

So far, some progress has been achieved in the study on the data structure of the subdivision surface. The edge-based data structures (wing-edge data structure [1] and half-edge data structure [2]), being able to express polygon meshes of arbitrary topological structure and applicable to all kinds of subdivision algorithms with a higher neighbourhood search efficiency, are the most common data structures for implementation of the subdivision algorithm at present. Based on the quadripartition feature of Catmull, Clark [1] and Loop [2] subdivision schemes, the quad-tree data structure [5] can be used for multi-resolution editing of the subdivision surface. However, when the subdivision depths get higher because of excessive pointer references, the two data structures above mentioned not only cause the memory consumption rapidly increased, but also become low in neighbourhood search efficiency. In order to realize rapid display of Loop subdivision surface, Pulli and Segal [6] proposed to express Loop subdivision surface by the creation of triangle patch pairs, which, however, are highly restricted to the structure, and adjacency relationship between triangle patch pairs has not been established. Ou and Bin [7] improved Pulli's model and proposed a solution for the case when the sum of triangular patches is odd with a detailed description of the implementation process of Loop subdivision surface. Settgast et al. [8], in his study on the adaptive subdivision of Catmull-Clark subdivision surface, put forward to express the subdivision surface patch by a two-dimensional array to solve the problem of excessive computer memory consumption for subdivision surface. However, definite adjacency relationship between subdivision surface patches was not established, and the problem of how to deal with the boundary and extraordinary vertex was not discussed.

In order to solve the above problems, a two-layer data structure named CELL is created in this paper for better representation of the subdivision surface by using the features of the topological structures of trilateral and quadrilateral subdivision surfaces. The fundamental principle of CELL is to express the subdivision surface patch by an array. With its compact and regular structure CELL has better performance in such aspects, including computer memory consumption, data query, algorithm development, and implementation of the subdivision surface. The organization of this paper is as follows. In Section 2 some common subdivision schemes are reviewed briefly. In Section 3 concepts and definitions of the data structure are proposed. In Section 4 the realization mechanism of a subdivision algorithm based on the data structure is described in details. In Section 5 some experiments are carried out to test the performance of the data structure. In Section 6 some applications of the data structure for a subdivision surface are presented. In Sections 7 and 8 discussions and conclusions can be found.

2. Subdivision surfaces

The subdivision surface is the limit status of an initial control mesh subject to constant subdivision under certain subdivision rules (geometric rule and topological rule). Let M^0 be an open or closed initial control mesh of arbitrary topology, S be a certain subdivision operator and M^n be a control mesh after n times of subdivision. Hence, the standard subdivision can be described as a linear process: M^{n+1}=SM^n, n = 0, 1, 2, 3. There is a wide variety of subdivision surfaces. They can be divided into an approximating subdivision scheme and interpolating subdivision scheme according to whether the limit surface is interpolated with the vertices of the control mesh, or they can be divided into a trilateral subdivision scheme and quadrilateral subdivision scheme according to the shape of the control mesh after subdivision, or they can be divided into a dual subdivision scheme and primal subdivision scheme according to the rules of topological splitting, a dual subdivision scheme is vertex splitting and a primal subdivision scheme is face splitting respectively. Several subdivision schemes widely used at present are listed in Table 1.

Table 1. Classification of common subdivision schemes

Subdivision scheme	Approximating	Interpolating	Trilateral	Quadrilateral	Primal	Dual
Catmull-Clark [1]	*			*	*	
Loop [2]	*		*		*	
Doo, Sabin [9]	*			*		*
Quad/triangle [10]	*		*	*	*	
Butterfly [11]		*	*		*	

No matter what the subdivision scheme is, according to the local correlation of the geometric rule, a subdivision surface model with complex structure can be divided into several mutually independent subdivision surface patches (SSP). Moreover, each SSP can be computed independently. According to the splitting consistency of the topological rule, each SSP of the same subdivision scheme has the same topological structure. As shown in Fig. 1, Catmull-Clark and Loop subdivision models, shown in Fig. 1(a) and Fig. 1(c) can be divided naturally into several trilateral and quadrilateral SSPs, as shown in Fig. 1(b) and Fig. 1(d) with the same topological structure respectively. Compared with the general polygon mesh, the properties above mentioned can be regarded as topological characteristics of the subdivision surface, which provides the theoretical basis for CELL construction.

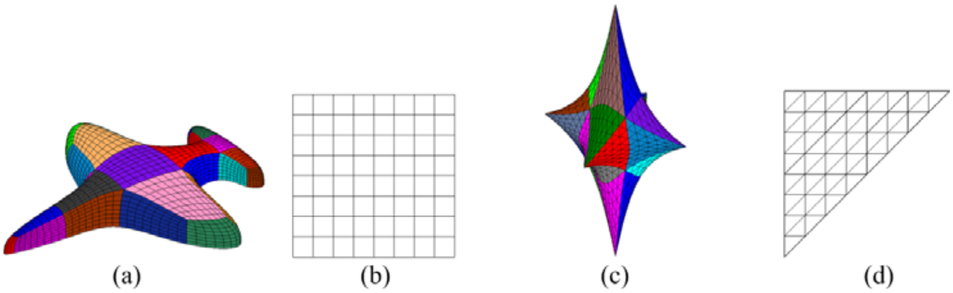

Fig. 1. Piecewise representation for subdivision surfaces

3. Concepts and definitions

By using the topological characteristics of the subdivision surface, a two-layer data structure called CELL is constructed for representing trilateral/quadrilateral subdivision schemes. CELL falls into an inner structure and outer structure, of which the former is composed of: a cell array denoted as A_c, auxiliary "boundary" subdivision structure denoted as A_e and auxiliary "corner vertex" subdivision structure denoted as A_v. As shown in Figs 2 and 3, Fig. 2(a) and Fig. 3(a) show an initial control mesh of Catmull-Clark/Loop subdivision schemes comprised of SSP and relevant neighborhood information, Fig. 2(b), Fig. 3(b), Fig. 2(c) and Fig. 3(c) indicate the relationship between the mesh vertices and CELL.

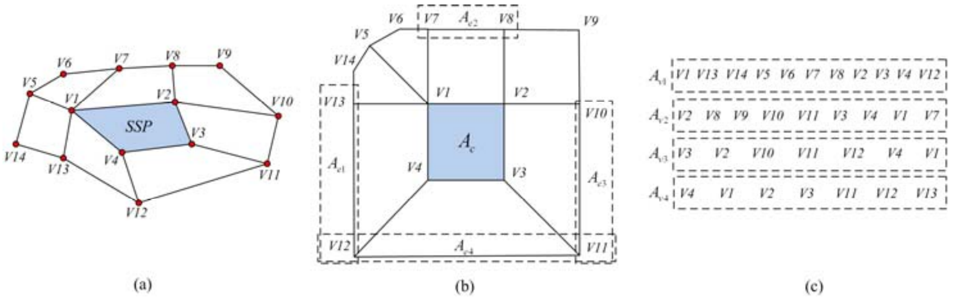
V6
V7
V8
V9
V5
V1
V2
V10
SSP
V3
V14
V13
V4
V11
V12
(a)
V6
V7
A_{e2}
V8
V9
V5
V14
V1
V2
V13
V10
A_c
A_{e1}
V4
V3
A_{e3}
V12
A_{e4}
V11
(b)
A_{v1} V1 V13 V14 V5 V6 V7 V8 V2 V3 V4 V12
A_{v2} V2 V8 V9 V10 V11 V3 V4 V1 V7
A_{v3} V3 V2 V10 V11 V12 V4 V1
A_{v4} V4 V1 V2 V3 V11 V12 V13
(c)

Fig. 2. Components of CELL for Catmull-Clark subdivision scheme

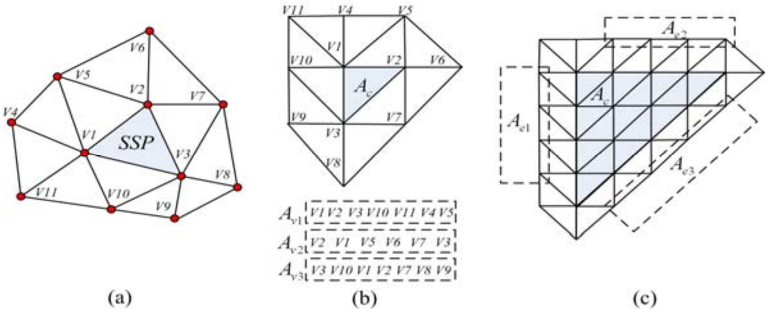

Fig. 3. Components of CELL for Loop subdivision scheme

3.1. Cell array

According to the topological structure of SSP of the quadrilateral subdivision scheme, the two-dimensional array is the best structure to represent SSP. Meanwhile, according to the topological structure of SSP of the trilateral subdivision scheme, array of arrays can be used to represent SSP. Two-dimensional array and array of arrays above mentioned can be collectively called cell array. The goal of creation of the cell array is to represent SSP by recording the data information of the mesh vertices of SSP. The topological information between the geometrical elements (vertex, edge and face) of SSP is implied in the row/column index relations of the cell array.

3.2. Auxiliary subdivision structure

The SSP cannot perform a subdivision operation only with its own data information. Some auxiliary subdivision structures must be constructed to provide the necessary data information for assisting the successful subdivision of SSP. According to different effects of the auxiliary subdivision structures, the auxiliary subdivision structures can be classified into two types: auxiliary boundary subdivision structure and auxiliary corner vertex subdivision structure. Based on one-dimensional array, the auxiliary boundary subdivision structure can be constructed by recording the data information about a column of vertices adjacent to a “boundary” of SSP. Based on one-dimensional array, the auxiliary corner vertex subdivision structure can be constructed by recording the data information about a “corner vertex” of SSP and its 1-neighborhood.

3.3. Outer structure of CELL

The goal of creation of the outer data structure is to realize inter-SSP data search for algorithm development and implementation of the subdivision surface by using the topological relations between SSPs. Since the topological relations between SSPs remain unchanged after subdivision, they can be expressed by recording the topological structure of the initial control mesh. In order to cope with the arbitrary topological structure of the initial control mesh, half-edge and winged-edge data structures can be used as an outer structure of CELL.

4. Implementation of Catmull-Clark subdivision algorithm

In view of the typicality of Catmull-Clark subdivision algorithm, Catmull-Clark subdivision algorithm is taken as an example to describe the realization mechanism of a subdivision algorithm based on CELL. The essence of CELL is to achieve piecewise representation of the subdivision surface. The subdivision of CELL also follows the common subdivision rules. As shown in Fig. 4, the hollow square dots are the new vertices and the circle dots are the original vertices. A new face-vertex as shown in Fig. 4(a) is computed as the average of all original vertices defining the face. A new edge-vertex as shown in Fig. 4(b) is computed as the average of the original vertices defining the edge and two new face-vertices of the faces sharing the edge. A new vertex-vertex as shown in Fig. 4(c) is computed by the linear combination of the original vertex and its 1-ring neighbours. A new boundary edge-vertex as shown in Fig. 4(d) is computed as the average of the original vertices defining the boundary edge. A new boundary vertex-vertex as shown in Fig. 4(e) is computed by the original boundary vertex and other two boundary vertices defining the boundary edge.

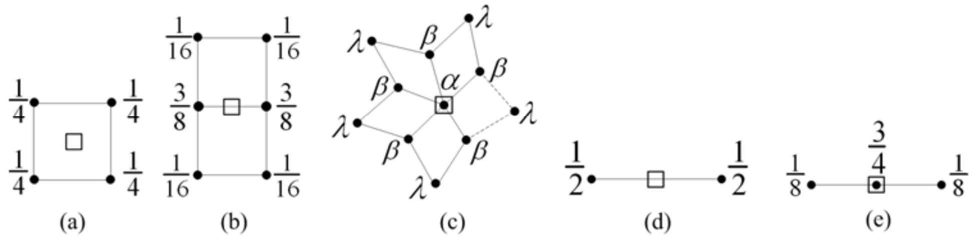

Fig. 4. Catmull-Clark subdivision masks

4.1. Subdivision of a cell array

According to different topological positions of the new vertices, new vertices generated from A_c, the subdivision can be divided into two categories: an internal new vertex and a boundary new vertex. The basic types of the internal new vertex mainly include the new face-vertex, new edge-vertex and new vertex-vertex. The basic types of the boundary new vertex mainly include the new boundary edge-vertex, new boundary vertex-vertex and new corner-vertex. As shown in Fig. 5, a big hollow dot is the original vertex of A_c. The solid circle dot, solid square dot and solid triangular dot are the new face-vertex, internal new vertex-vertex and internal new edge-vertex respectively. The hollow triangular dot, hollow square dot and hollow circle dot are the boundary new edge-vertex, internal new vertex-vertex and new corner-vertex respectively.

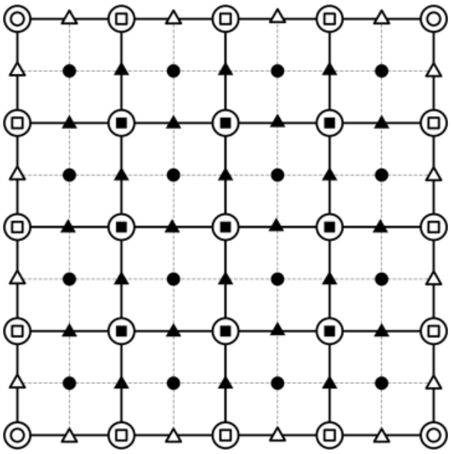

Fig. 5. Basic types of new vertices for subdivision of a cell array

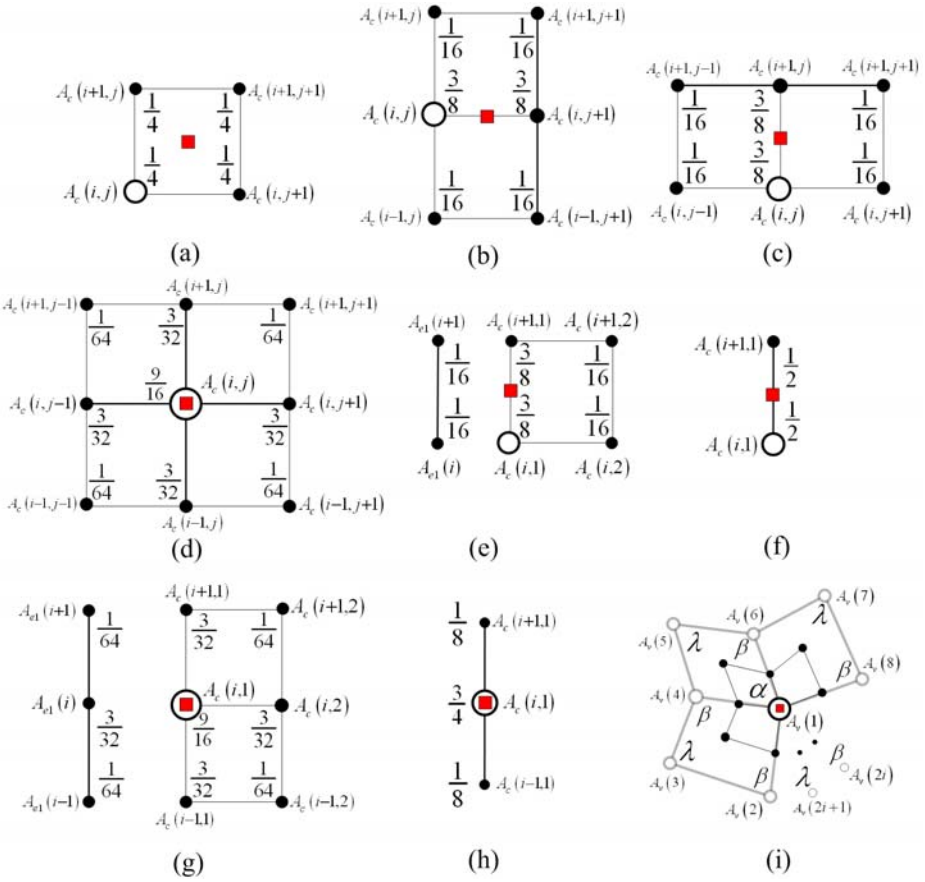

Fig. 6. Catmull-Clark subdivision masks for a cell array

After n times of subdivision, the new vertices in A_c total up to $(2^n+1)^2$, including 4^{n-1} new face vertices, $(2^{n-1}-1)^2$ internal new vertex vertices, $2^{2n-1}-2^n$ internal new edge vertices, $2^{n+1}-4$ boundary new vertex vertices, 2^{n+1} boundary new edge vertices and 4 new corner vertices. As shown in Fig. 5, after 3 times of subdivision, the new vertices in A_c total up to 81, including 16 new face vertices, 9 internal new vertex vertices, 24 internal new edge vertices, 12 boundary new vertex vertices, 16 boundary new edge vertices and 4 new corner vertices. From the viewpoint of programming implementation, a concept of the “reference point” is introduced. The reference points refer to the vertices at the particular positions before A_c subdivision. During A_c subdivision, the geometric calculation of the new vertices and their topological positions are closely related to the reference points. In Fig. 6 are shown the subdivision masks for computing the new face-vertex, internal new edge-vertex I and II, internal new vertex-vertex, fake and real boundary new edge-vertex, fake and real boundary new vertex-vertex, and new corner-vertex, the hollow circle dots are the reference points, the square dots are the new vertices and the circle dots are the original vertices. From the viewpoint of geometric calculation of new vertices, the new vertices of A_c still follow the basic subdivision rule. The subdivision mask as shown in Fig. 6(a) is essentially the subdivision mask of the new face-vertex. The subdivision masks as shown in Fig. 6(b), Fig. 6(c) and Fig. 6(e) are essentially the subdivision mask of the new edge-vertex. The subdivision masks as shown in Fig. 6(d), Fig. 6(g) and Fig. 6(i) are essentially the subdivision mask of the new vertex-vertex. The subdivision masks as shown in Fig. 6(f), Fig. 6(h) are essentially the subdivision masks of the new boundary edge-vertex and new boundary vertex-vertex respectively.

Assuming the subdivided A_c as A_{cs} and the reference point as $A_c(i, j)$, it is very important to determine the topological position of the new vertex in the process of A_c subdivision. As shown in Fig. 5, the full lines show the topological structure of A_c and the dotted lines show the topological structure of A_{cs}. If the size of A_c is $K.K$, the size of A_{cs} is $(2K–1).(2K–1)$. And based on that, the topological position of each type of a new vertex can be deduced from the reference point as shown in Fig. 6. For example, as shown in Fig. 6(b), if the reference point is indicated by $A_c(i, j)$, the new edge-vertex can be indicated by $A_{cs}(2i–1, 2j)$. The relations between the geometric calculation/topological position of the new vertices and the reference points are shown as follows:

The new face-vertex expressed as a square dot in Fig. 6(a):

$$A_{cs}(2i, 2j) = \frac{1}{4}\Big(A_c(i, j) + A_c(i+1, j) + A_c(i, j+1) + A_c(i+1, j+1)\Big). \tag{1}$$

The internal new edge-vertex I expressed as a square dot in Fig. 6(b):

$$\begin{gathered} A_{cs}\left(2i–1, 2j\right) = \frac{3}{8}\Big(A_c(i, j)+A_c(i, j+1)\Big)+ \\ +\frac{1}{16}\Big(A_c(i–1, j)+A_c(i+1, j)+A_c(i–1, j+1)+A_c(i+1, j+1)\Big). \end{gathered} \tag{2}$$

The internal new edge-vertex II expressed as a square dot in Fig. 6(c):

$$\begin{gathered} A_{cs}\left(2i, 2j–1\right) = \frac{3}{8}\Big(A_c(i, j)+A_c(i+1, j)\Big)+ \\ +\frac{1}{16}\Big(A_c(i, j–1)+A_c(i, j+1)+A_c(i+1, j–1)+A_c(i+1, j+1)\Big). \end{gathered} \tag{3}$$

The internal new vertex-vertex expressed as a square dot in Fig. 6(d):

$$\begin{gathered} A_{cs}\left(2i–1, 2j–1\right) = \frac{9}{16}A_c\left(i, j\right) + \frac{3}{32}\begin{pmatrix} A_c(i+1, j)+A_c(i–1, j) \\ +A_c(i, j+1)+A_c(i, j–1) \end{pmatrix}+ \\ +\frac{1}{64}\begin{pmatrix} A_c(i+1, j–1)+A_c(i+1, j+1) \\ +A_c(i–1, j–1)+A_c(i–1, j+1) \end{pmatrix}. \end{gathered} \tag{4}$$

The fake boundary new edge-vertex expressed as a square dot in Fig. 6(e):

(5) $$A_{cs}\left(2i,1\right)=\frac{3}{8}\left(A_c\left(i,1\right)+A_c\left(i+1,1\right)\right)+$$
$$+\frac{1}{16}\left(A_c\left(i,\,2\right)+A_c\left(i+1,\,2\right)+A_{e1}\left(i\right)+A_{e1}\left(i+1\right)\right).$$

The real boundary new edge-vertex expressed as a square dot in Fig. 6(f):

(6) $$A_{cs}\left(2i,1\right)=\frac{1}{2}\left(A_c\left(i,1\right)+A_c\left(i+1,1\right)\right).$$

The fake boundary new vertex-vertex expressed as a square dot in Fig. 6(g):

(7) $$A_{cs}\left(2i-1,1\right)=\frac{9}{16}A_c\left(i,1\right)+\frac{3}{32}\begin{pmatrix}A_c\left(i+1,1\right)+A_c\left(i-1,1\right)\\+A_c\left(i,\,2\right)+A_{e1}\left(i\right)\end{pmatrix}+$$
$$+\frac{1}{64}\begin{pmatrix}A_c\left(i+1,\,2\right)+A_c\left(i-1,\,2\right)\\+A_{e1}\left(i+1\right)+A_{e1}\left(i-1\right)\end{pmatrix}.$$

The real boundary new vertex-vertex expressed as a square dot in Fig. 6(h):

(8) $$A_{cs}\left(2i-1,1\right)=\frac{3}{4}A_c\left(i,1\right)+\frac{1}{8}\left(A_c\left(i+1,1\right)+A_c\left(i-1,1\right)\right).$$

The new corner-vertex expressed as a square dot in Fig. 6(i):

(9) $$A_{cs}\left(i,j\right)=\alpha A_v\left(1\right)+\beta\sum_{i=1}^{m}A_v\left(2i\right)+\lambda\sum_{i=1}^{m}A_v\left(2i+1\right),$$

where m is the valance of corner-vertex, $\beta=\frac{3}{2m^2},,$ $\lambda=\frac{1}{4m^2},$ $\alpha=1-m\beta-m\lambda$.

4.2. Subdivision of auxiliary subdivision structure

To ensure successful updating of A_c, corresponding updating mechanisms are required for A_e and A_v. The basic types of new vertices of A_e include a new face-vertex and a new edge-vertex. And the basic types of the new vertices of A_v include a new face-vertex, a new edge-vertex and a new vertex-vertex. As shown in Fig. 7, the hollow circle dot is the original vertex of A_e, the solid circle dot and solid triangular dots are the new face-vertex and new edge-vertex. The hollow square dots are the original vertices of A_v. The solid circle dot, solid triangular dot and solid square dot are the new face-vertex, new edge-vertex and new vertex-vertex. After n times of subdivision, the new vertices in A_e total up to 2^n+1, including 2^{n-1} new face vertices and $2^{n-1}+1$ new edge vertices. The new vertices in A_v total up to $2m+1$ (m is the valence of the corresponding corner vertex), including m new face vertices, m new edge vertices and 1 new vertex. In Fig. 8 are the subdivision masks for computing the new face-vertex, new edge-vertex and subdivided A_v, the hollow circle dots are the reference points for computing new vertices.

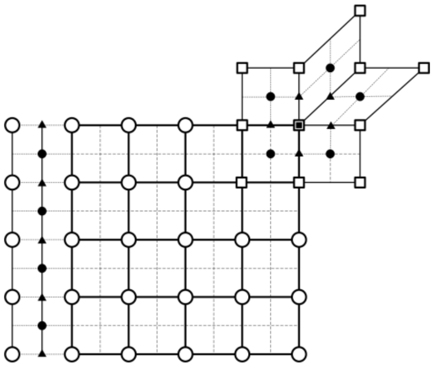

Fig. 7. Basic types of new vertices for subdivision of auxiliary subdivision structure

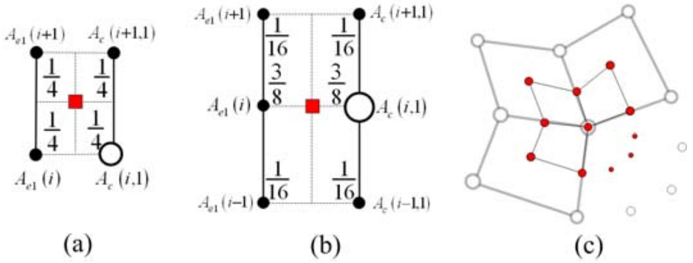

Fig. 8. Catmull-Clark subdivision masks for auxiliary subdivision structure

Assuming the subdivided A_e as A_{es} and subdivided A_v as A_{vs}, the relations between the geometric calculation/topological position of the new vertices and the reference points are shown as follows:

The new face-vertex expressed as a square dot in Fig. 8(a):

$$A_{e1s}(2i) = \frac{1}{4}\Big(A_c(i,1)+A_c(i+1,1)+A_{e1}(i)+A_{e1}(i+1)\Big). \tag{10}$$

The new edge-vertex expressed as a square dot in Fig. 8(b):

$$\begin{gathered} A_{e1s}(2i-1) = \frac{3}{8}\Big(A_c(i,1)+A_{e1}(i)\Big)+ \\ +\frac{1}{16}\Big(A_c(i+1,1)+A_c(i-1,1)+A_{e1}(i-1)+A_{e1}(i+1)\Big). \end{gathered} \tag{11}$$

A_{vs} expressed as circle dots in Fig. 8(c):

$$A_{vs} = M_m A_v, \tag{12}$$

where A_v and A_{vs} are column vectors of ordered vertices formed by corner vertices and their 1-neighborhood points before and after the subdivision respectively; M_m is the subdivision matrix deduced based on the valences m of the corner vertices and the column vectors of the ordered vertices.

5. Algorithm test

Some experiments are carried out to implement the subdivision algorithm and evaluate the performance of CELL in computer memory consumption and the computing time of SSP. We have implemented our approach on Windows 7 64-bit operating system using Matlab7.13 software and run on an i5-3570k processor PC with 4GB RAM.

5.1. Implementation of Catmull-Clark subdivision surface

As shown in Fig. 9, Fig. 9(a) and Fig. 9(c) show the initial control mesh of Catmull-Clark subdivision surface, Fig. 9(b) and Fig. 9(d) show the corresponding results of Catmull-Clark subdivision algorithm implementation by using CELL. The experiment results show that either open or closed Catmull-Clark subdivision surfaces can be represented well by CELL.

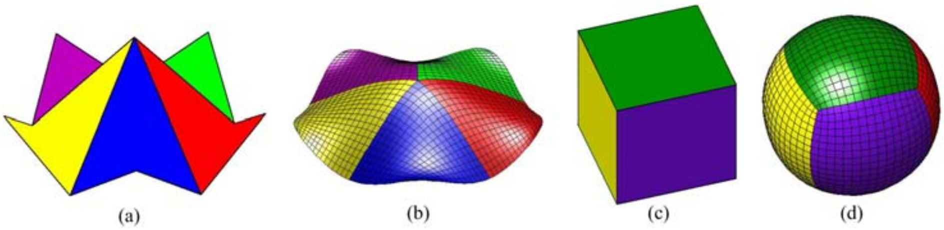

Fig. 9. Implementation of Catmull-Clark subdivision surface

5.2. Computer memory consumption

Since SSP is the data element of CELL, SSP can be used as a test model for testing the performance of CELL with relation to Computer Memory Consumption (CMC). Under different conditions of the subdivision time, the test results of computer memory consumption for implementation of Catmull-Clark SSP by CELL are recorded in Table 2.

Table 2. Computer memory consumption

Subdivision times	1	2	3	4	5	6	7	8	9	10
CMC: measured in Mega-Bytes	0.00003	0.0001	0.0003	0.001	0.004	0.016	0.066	0.264	1.052	4.202

5.3. Computing time

In the same way, SSP is used as a test model for testing the performance of CELL with respect to the Computing Time (CT). Under different conditions of the subdivision time, the test results of the computing time for implementation of Catmull-Clark SSP by CELL are recorded in Table 3.

Table 3. Computing time

Subdivision times	1	2	3	4	5	6	7	8	9	10
CT: measured in seconds	0.001	0.004	0.007	0.013	0.026	0.054	0.167	0.609	2.367	9.603

6. Algorithm development

The data structure plays an important role in developing relevant algorithms of subdivision surface. By virtue of simple intra-SSP topological relationship and explicit inter-SSP topological structure, CELL not only reduces the difficulty in algorithm implementation and cuts the algorithm development cycle, but also enables good readability and portability in program codes and easy program maintenance.

6.1. Implementation of Loop, Doo-Sabin and Quad/triangle subdivision algorithms

In addition to Catmull-Clark subdivision algorithm, CELL is also applied to implement Loop, Doo-Sabin and Quad/triangle subdivision algorithms. The key to subdivision algorithm implementation is the creation of corresponding A_c. In respect of quadrilateral subdivision scheme, the corresponding A_c can be created by using a two-dimensional array. For a trilateral subdivision algorithm, the corresponding A_c can be created by using array of arrays. The creation methods for A_e and A_v are essentially the same. As shown in Fig. 10, Fig. 10(a), Fig. 10(c) and Fig. 10(e) show the initial control meshes of Loop, Doo-Sabin, Quad/triangle subdivision surface, Fig. 10(b), Fig. 10(d) and Fig. 10(f) show the subdivision models of Loop, Doo-Sabin and Quad/triangle subdivision surface.

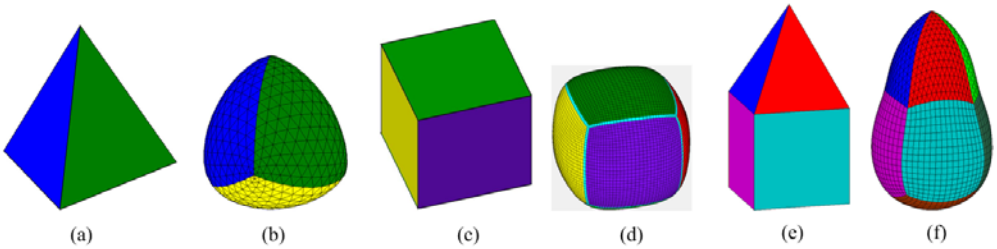

Fig. 10. Implementation of Loop, Doo-Sabin and Quad/triangle subdivision surface

6.2. Establishment of sharp and semi-sharp features

Smooth surface can be created by a subdivision algorithm, but real objects often have sharp and semi-sharp features. In order to enhance the modelling capability of the subdivision surface, Hoppe et al. [12] proposed methods to establish sharp and semi-sharp features for the subdivision surface. Sharp and semi-sharp features of the subdivision surface can also be established efficiently by CELL. As shown in Fig. 11, Fig. 11(a) shows an initial control mesh; Fig. 11(b) shows a Catmull-Clark subdivision surface without sharp and semi-sharp features, Fig. 11(c) shows a Catmull-Clark subdivision surface with semi-sharp features, Fig. 11(d) shows a Catmull-Clark subdivision surface with a sharp feature.

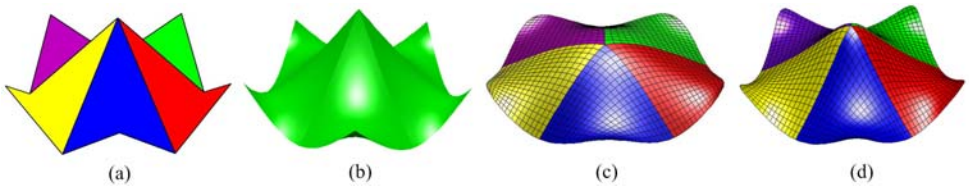

Fig. 11. Sharp and semi-sharp features

6.3. Amalgamation of image/Z-map models with subdivision surface

CELL records the information of the subdivision surface patch via a two-dimensional array, and image and Z-map models also record the information of the image and graph through a two-dimensional array. In order to enhance the modeling capability of the subdivision surface, an attempt is made to study the amalgamation of the image/Z-map models with Catmull-Clark subdivision surface. As shown in Fig. 12, Fig. 12(a) shows a character image; Fig. 12(c) shows a graphic image, and Fig. 12(b) and Fig. 12(d) show subdivision surfaces combined with an image. As shown in Fig. 13, Fig. 13(a) and Fig. 13(c) show Z-map models, Fig. 13(b) and Fig. 13(d) show subdivision surface amalgamated with Z-map model.

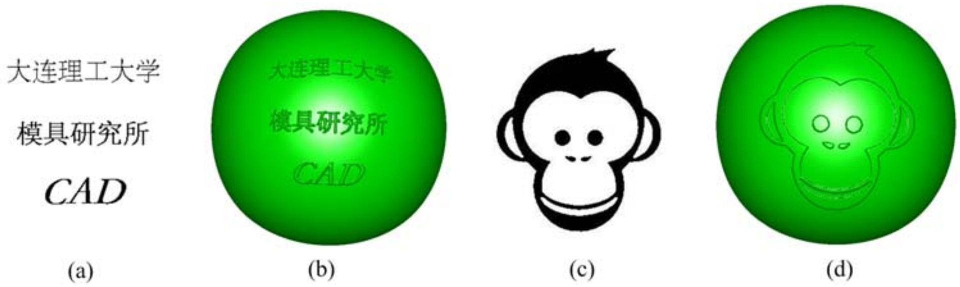

Fig. 12. Amalgamation of an image with subdivision surface

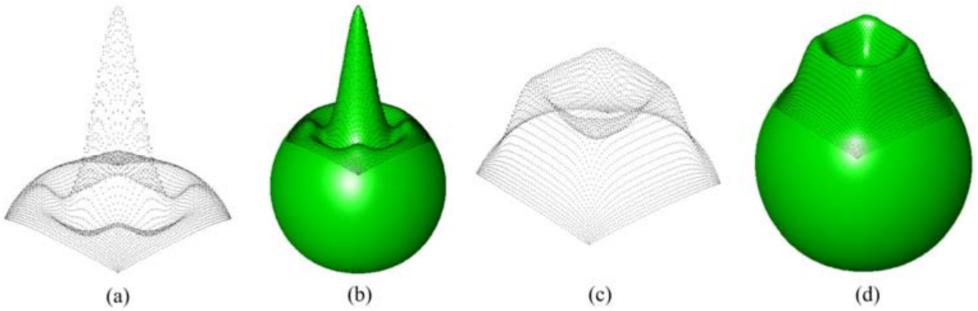

Fig. 13. Amalgamation of Z-map model with subdivision surface

7. Discussions

7.1. Computer memory consumption

With regard to the subdivision surfaces, the computer memory consumption is mainly used to store the geometrical information of the subdivision surface vertices and the information of the topological structure among mesh vertices, edges and faces. The latter part accounts for a large percent of memory consumption. The computer memory consumption of the subdivision surface is increased exponentially with the increase of the subdivision times. However, CELL has a significant advantage in saving the computer memory consumption. Based on CELL, the topological information of the subdivision surface can be divided into intra-SSP topological information and inter-SSP topological information. The topological information of the intra-SSP structure is naturally represented by the row/column indexing relationship of a cell array. While the topological information of inter-SSP structure can be evolved from the topological structure of the initial control mesh of subdivision surface. Foremost, its volume is generally small and does not expand with the increase of the subdivision times.

7.2. Data query

Data query is the most basic operation used in the implementation or analysis of the subdivision surface or in the development of some applications based on subdivision surfaces. CELL has obvious advantages in terms of zone, content and speed of data query. If you want to search mesh elements within a neighboring zone of a mesh element (vertex, edge or face), or mesh elements having particular adjacency relationship with it, the required mesh elements can be obtained through the row/column index of A_c. The search can be completed in constant time as long as the search zone does not exceed this A_c. If the search zone is beyond this A_c, the desired information can be found in other A_c areas through the inter-SSP topology information.

7.3. Implementation of subdivision surface

Based on CELL, the global subdivision of the subdivision surface can be transformed into local subdivision of CELL. Each SSP is subdivided by using the same subdivision pattern; no determination of a vertex valence is required; the boundary conditions are decided within a specific limited zone. So, the subdivision surface can be rapidly implemented. The topological relationship between the mesh elements can be certainly established upon the subdivision algorithm completion.

8. Conclusion

By utilizing the topological characteristics of the subdivision surface, CELL is proposed for representing trilateral/quadrilateral subdivision surfaces. Certain excellent features, such as local subdivision, pattern subdivision, compact topological structure and local natural parameterization make CELL better than other data structures in terms of data storage, search efficiency, subdivision computing speed, algorithm development and maintenance. Within the broad applicability of CELL, Catmull-Clark, Loop, Doo-Sabin and Quad/triangle subdivision schemes can be efficiently implemented. The application of CELL not only establishes sharp and semi-sharp features of the subdivision surface, but also realizes the amalgamation of the image/Z-map model and subdivision surface, so that the modelling capability of the subdivision surface can be improved by using CELL. The results of the algorithm test and theoretical analysis show that CELL has excellent performance with respect to computer memory consumption, data query and implementation of the subdivision surface. By making full use of CELL, it will be appropriate for generalization and application of the subdivision surface in more fields.

Acknowledgements. This research is supported by the National Natural Science Foundation of China under Grant No 51205040 and China Postdoctoral Science Foundation No 2012M510800. These supports are greatly appreciated.

References

1. Catmull, E., J. Clark. Recursively Generated B-Spline Surfaces on Arbitrary Topological Meshes. – Computer-Aided Design, Vol. **10**, 1978, No 6, 350-355.
2. Loop, C. Smooth Subdivision Surfaces Based on Triangles. M. S. Thesis, Dept. Mathematics, University of Utah, Utah, USA, 1987.
3. Kraemer, P., D. Cazier, D. Bechmann. Extension of Half-Edges for the Representation of Multiresolution Subdivision Surfaces. – The Visual Computer, Vol. **25**, 2009, No 2, 149-163.
4. Campagna, S., L. Kobbelt, H. P. Seidel. Directed Edges-A Scalable Representation for Triangle Meshes. – Journal of Graphics Tools, Vol. **3**, 1998, No 4, 1-11.
5. Weiler, K. Edge-Based Data Structures for Solid Modeling in Curved-Surface Environments. – IEEE Computer Graphics and Applications, Vol. **5**, 1985, No 1, 21-24.
6. Pulli, K., M. Segal. Fast Rendering of Subdivision Surfaces. – In: Proc. of the Eurographics Workshop on Rendering Techniques, Porto, Portugal, 1996, 61-70.
7. Ou, Shiqi, Hongzan Bin. A Compact Data Structure for Implementing Loop Subdivision. – International Journal of Advanced Manufacturing Technology, Vol. **29**, 2006, No 11, 1151-1158.
8. Settgast, V., K. Müller, C. Fünfzig, D. W. Fellner. Adaptive Tesselation of Subdivision Surfaces. – Computers and Graphics, Vol. **28**, 2004, No 1, 73-78.
9. Doo, D., M. Sabin. Behaviour of Recursive Division Surfaces Near Extraordinary Points. – Computer-Aided Design, Vol. **10**, 1978, No 6, 356-360.
10. Stam, J., C. Loop. Quad/Triangle Subdivision. – Computer Graphics Forum, Vol. **22**, 2002, No 1, 79-85.
11. Dyn, N., D. Levine, J. A. Gregory. A Butterfly Subdivision Scheme for Surface Interpolation with Tension Control. – ACM Transactions on Graphics, Vol. **9**, 1990, No 2, 160-169.
12. Hoppe, H. et al. Piecewise Smooth Surface Reconstruction. – In Proc. of the 21st Annu. Conf. on Computer Graphics and Interactive Techniques, Orlando, Florida, 1994, 295-302.