Adaptive Subdivision of Catmull-Clark Subdivision Surfaces

Jun-Hai Yong[1] and Fuhua (Frank) Cheng[2]

[1] Tsinghua University, yongjh@tsinghua.edu.cn
[2] University of Kentucky, cheng@cs.uky.edu

ABSTRACT

Catmull-Clark subdivision scheme provides a powerful method for building smooth and complex surfaces. But the number of faces in the uniformly refined meshes increases sharply with respect to subdivision depth. This paper presents an adaptive subdivision technique as a solution to this problem. Instead of subdivision depths of mesh faces, the adaptive subdivision process is driven by labels of mesh vertices, which can be viewed as subdivision depths of the surface in the vicinity of the mesh vertices. Smooth transition between faces with different subdivision depths is provided by an unbalanced-subdivision process. The resulting meshes are crack-free, and all the faces are quadrilaterals. Limit surface of the resulting meshes is the same as the original limit surface. Test results show that the number of faces generated in the adaptively refined meshes is one order less than the uniform approach. The proposed technique works for cubic Doo-Sabin subdivision surfaces, non-uniform cubic subdivision surfaces and combined subdivision surfaces as well.

Keywords: Adaptive subdivision, Mesh Generation, Subdivision surfaces

1. INTRODUCTION

Subdivision surfaces have become popular recently in graphical modeling, animation and CAD/CAM [7] [18] because of their stability in numerical computation, simplicity in coding and, most importantly, their capability in modeling/representing complex shape of arbitrary topology. Research work for subdivision surfaces has been carried out in several important areas such as surface evaluation [22], surface trimming [18], Boolean operations [2], and mesh editing [24]. However, the work is far from being complete yet; for instance, research work is still needed for surface tessellation and shape design. The purpose of this paper is to study subdivision surface tessellation problem. A new technique that will significantly reduce the number of faces in the resulting meshes will be presented. The new technique improves the efficiency of subsequent data communication, surface operations (trimming/intersection), rendering and finite element analysis for subdivision surfaces significantly.

Research work for reducing the number of faces in a mesh can be classified into three directions. *Mesh simplification* [1] [9-12] [17] [et al] is the most popular among the three directions over the past decade. The aim is to remove over-sampled vertices and produce approximate meshes with various levels of detail. The second direction focuses on approximating the limit surface by surfaces that we know of, such as a displaced subdivision surface [15] or Nurbs patches [19]. The third one is to apply adaptive or local refinement schemes to areas specified by the user or determined by the application. The resulting mesh should be crack-free and have the same limit surface as the uniformly refined mesh. Kobbelt has presented a red-green triangulation based adaptive refinement scheme for $\sqrt{3}$-subdivision surfaces [14], and Velho and Zorin have presented a bisection based adaptive refinement scheme for 4-8 subdivision [23]. As far as quadrilateral meshes are concerned, Kobbelt has a Y-element based adaptive refinement scheme for interpolatory subdivision surfaces [13] and Sederberg et al. have a T-junction based local refinement scheme for non-uniform rational Catmull-Clark surfaces (NURCCs) [21]. However, Kobbelt's technique [13] may fail to obtain balanced nets for several search attempts, and the resulting mesh in [21] does not maintain a quadrilateral structure.

The technique presented in this paper belongs to the third direction. A label-driven adaptive subdivision technique is presented here. The technique will be presented for the Catmull-Clark subdivision process [3]. But it can be used for cubic Doo-Sabin [8], non-uniform cubic [20], and combined [16] subdivision surfaces as well. One only needs to replace the Catmull-Clark vertex-computing formulas with the corresponding vertex-computing formulas of these schemes. The reason that we use the Catmull-Clark subdivision scheme here is because it is a frequently used subdivision surface generation technique and it is simple for presentation.

Our work is inspired by [4] and [13] which use similar techniques: *unbalanced subdivision* and "Y"-*element*, to avoid crack, respectively. The idea of label-driven subdivision [4] is followed here, that is, labels are assigned to vertices of control mesh and used to control the subdivision process. Illegal faces are removed from the initial mesh using a greedy algorithm. Two subdivision techniques are used in the adaptive subdivision process. The standard Catmull-Clark subdivision (B_sub) is used to reach the desired subdivision depth and an unbalanced subdivision (U_sub) is used to provide smooth, crack-free transition between regions with different subdivision depths. Hence, there is no need to perform unnecessary subdivision on regions that are already flat enough, as the ordinary Catmull-Clark subdivision scheme would do. The adaptively refined meshes converge to the same limit surface as the meshes generated by the standard Catmull-Clark subdivision scheme, but the number of faces in the adaptively refined mesh is one order less than that in the mesh generated by the standard Catmull-Clark subdivision scheme. Therefore, the adaptive refinement scheme can reach the same precision requirement with significantly less faces in the refined mesh. This makes Catmull-Clark subdivision surfaces a practical option in animation, modeling and CAD/CAM applications because the memory problem is no longer an issue when dealing with rendering and tessellation (including finite element generation).

The remaining part of the paper is arranged as follows. A brief review of the Catmull-Clark subdivision scheme and the definitions of subdivision depth and labels are given in Section 2. The label-driven subdivision process, including B_sub and U_sub, is presented in Section 3. The illegal face elimination process is presented in Section 4. The proof that the adaptively refined meshes converge to the same limit surface is shown in Section 5. Implementation issues and test results are shown in Section 6. Concluding remarks are given in Section 7.

2. PROBLEM FORMULATION AND DEFINITIONS

Given a control mesh of arbitrary topology, the goal here is to construct a sequence of adaptively refined meshes that would converge to the same *Catmull-Clark subdivision surface*, but with much fewer vertices and faces than what one would get in the traditional Catmull-Clark subdivision process. The mesh refining process will be driven by *labels* of mesh vertices. We need a few definitions first.

2.1 Catmull-Clark Subdivision Surfaces

Given a control mesh, a *Catmull-Clark subdivision surface* (CCSS) is generated by iteratively refining the control mesh [3]. The iteratively refined control meshes converge to a limit surface. The limit surface is called a *subdivision surface* because the mesh refining process is a generalization of the uniform bicubic B-spline surface subdivision technique. Therefore, CCSSs include uniform B-spline surfaces and piecewise Bezier surfaces as special cases. It is known now that CCSSs include non-uniform B-spline surfaces and NURBS surfaces as special cases as well [20]. The Catmull-Clark mesh refining process will also be called the *Catmull-Clark subdivision*, or simply the *subdivision step* subsequently. The *valence* of a mesh vertex is the number of mesh edges adjacent to the vertex. A mesh vertex is called an *extra-ordinary vertex* if its valence is different from four.

Mesh faces of a CCSS generated after one iteration of the subdivision step are always quadrilaterals. The number of extra-ordinary vertices remains the same after one iteration of the subdivision step as well. Therefore, after at most two iterations of the subdivision step, each face has at most one extra-ordinary vertex. We shall assume that all the mesh faces considered subsequently are quadrilaterals and each of them has at most one extra-ordinary vertex. A mesh face with an extra-ordinary vertex will be called an *extra-ordinary face*.

2.2 Subdivision Depth and Labels

The given control mesh will be referred as $\mathbf{M}_0$ and the limit surface of $\mathbf{M}_0$ will be referred as $\mathbf{S}$. For each positive integer k, $\mathbf{M}_k$ refers to the control mesh obtained after applying the Catmull-Clark subdivision k times to $\mathbf{M}_0$. For each interior face $\mathbf{f}$ of $\mathbf{M}_k$, there is a corresponding patch $\mathbf{s}$ on the limit surface $\mathbf{S}$. $\mathbf{f}$ and $\mathbf{S}$ can be parametrized on the same parameter space $\mathbf{D}=[0,1]$ X$[0,1]$ [22]. $\mathbf{f}$ is a bilinear *rule surface* and $\mathbf{s}$ is a uniform bicubic B-spline surface patch. The *error* of a mesh face $\mathbf{f}$ is defined by

$$\xi_{\mathbf{f}}=max_{(u,v)\in\mathbf{D}}\,||\,\mathbf{f}(u,v)-\mathbf{s}(u,v)\,||,$$

where $\mathbf{s}$ is the corresponding patch of $\mathbf{f}$ on the limit surface $\mathbf{S}$.

Given an $\varepsilon>0$, we shall assume that each interior face $\mathbf{f}$ of $\mathbf{M}_0$ has been assigned a *subdivision depth* $d_{\mathbf{f}}$ so that if $\mathbf{f}$ is recursively subdivided $d_{\mathbf{f}}$ times, then errors of the resulting mesh faces are all smaller than ε. The subdivision depth of an interior face of $\mathbf{M}_0$ can be computed using a technique presented in [5] [6]. The *label* of an interior vertex $\mathbf{v}$ of $\mathbf{M}_0$, denoted $L(\mathbf{v})$, is defined as the maximum of adjacent interior faces' subdivision depths, i.e.,

$$L(\mathbf{v})=max\{\ d_\mathbf{f}\ |\ \mathbf{f} \text{ is an interior face of } \mathbf{M}_0 \text{ and } \mathbf{v} \text{ is a vertex of } \mathbf{f}\ \}. \qquad (1)$$

Figure 1(a) shows subdivision depths of interior faces of a control mesh and (b) shows corresponding labels of the interior vertices.

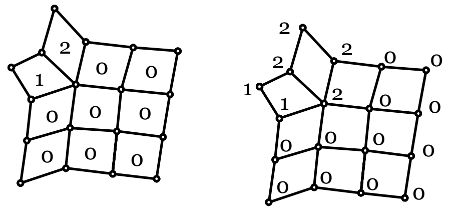

Fig. 1. (a) Subdivision depths of interior faces and (b) labels of interior vertices.

3. LABEL-DRIVEN ADAPTIVE SUBDIVISION

The adaptive subdivision process is driven by vertex labels and is performed on individual mesh faces. After each subdivision step, labels are assigned to the newly generated vertices so they can drive the next subdivision step. The adaptive subdivision process stops when labels of all the mesh vertices are zero. In the following, $\mathbf{M}_k$, k=1, 2, ..., stand for the meshes generated by the adaptive refinement process Variables without a bar refer to those of $\mathbf{M}_{k\text{-}1}$, and variables with a bar refer to those of $\mathbf{M}_k$.

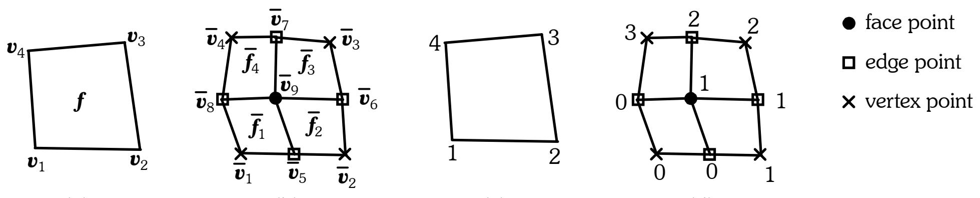

Fig. 2. Balanced subdivision: (a) before; (b) after. Labels: (c) before; (d) after.

3.1 B_sub: Balanced Catmull-Clark Subdivision

The adaptive subdivision of $\mathbf{M}_{k\text{-}1}$, $k\geq 1$, is performed as follows. If a given face has two or more positive vertex labels, a *balanced Catmull-Clark subdivision*, called *B_sub* for short, is performed on that face. A B_sub is a standard Catmull-Clark subdivision, but only those new vertices that are directly associated with the given face are used to construct new faces for that face. For instance, in Figure 2(a), if $\mathbf{v}_1$, $\mathbf{v}_2$, $\mathbf{v}_3$, and $\mathbf{v}_4$ are the vertices of a face $\mathbf{f}$ in $\mathbf{M}_{k\text{-}1}$, then nine of the new vertices created by a standard Catmull-Clark subdivision are directly associated with this face. These nine vertices include a *face point* $\bar{\mathbf{v}}_9$, four *vertex points* $\bar{\mathbf{v}}_1$, $\bar{\mathbf{v}}_2$, $\bar{\mathbf{v}}_3$, $\bar{\mathbf{v}}_4$, and four *edge points* $\bar{\mathbf{v}}_5$, $\bar{\mathbf{v}}_6$, $\bar{\mathbf{v}}_7$, $\bar{\mathbf{v}}_8$. A B_sub uses these nine points to form four new faces $\bar{\mathbf{f}}_1$, $\bar{\mathbf{f}}_2$, $\bar{\mathbf{f}}_3$, and $\bar{\mathbf{f}}_4$ (see Figure 2(b)). However, the creation of the new vertices and faces is symbolic only, coordinates of the new vertices will not be computed at this moment yet These new vertices will be marked with an "UPDATE" to indicate that they will be computed at a later stage.

A *label* will be assigned to each of the new vertices. The label of a new *vertex point* is defined as follows:

$$L(\bar{\mathbf{v}}_i)=max\{\ 0, L(\mathbf{v}_i)\text{-}1\ \}, \qquad i=1,2,3,4,$$

i.e., if label of the original vertex is positive, label of the new vertex point is one less than the label of the original vertex. Otherwise, label of the new vertex point is set to zero. The label of a new edge point is equal to the minimum of adjacent vertex points' labels. That is,

$$L(\bar{\mathbf{v}}_i)=min\{\ L(\bar{\mathbf{v}}_{i\text{-}4}), L(\bar{\mathbf{v}}_{i\text{-}3})\ \}, \qquad i=5,6,7,8,$$

where i-3 is modulo 4 when i=8. Label of the new face point depends on labels of adjacent edge points. If the labels of the adjacent edge points are all zero, label of the new face point is set to zero. Otherwise, label of the new face point is set to the minimum of the positive edge point labels, i.e.,

$$L(\bar{\mathbf{v}}_9)=0, \qquad \text{if } L(\bar{\mathbf{v}}_5)=L(\bar{\mathbf{v}}_6)=L(\bar{\mathbf{v}}_7)=L(\bar{\mathbf{v}}_8)=0,$$

or

$$L(\bar{\mathbf{v}}_9)=min\,\{\, L(\bar{\mathbf{v}}) \mid \bar{\mathbf{v}} \in \{\, \bar{\mathbf{v}}_5, \bar{\mathbf{v}}_6, \bar{\mathbf{v}}_7, \bar{\mathbf{v}}_8 \,\} \text{ and } L(\bar{\mathbf{v}})\neq 0 \,\}, \quad \text{otherwise.}$$

If the labels of vertices $\mathbf{v}_1$, $\mathbf{v}_2$, $\mathbf{v}_3$, and $\mathbf{v}_4$ in Figure 2(a) are 1, 2, 3 and 4, respectively (Figure 2(c)), then labels of the new vertices after a B_sub are the ones shown in Figure 2(d).

Note that if the label of a new vertex point is zero then labels of adjacent edge points are both zero, and label of the new face point is zero only if labels of all the adjacent edge points are zero. Therefore, a new face can never have two positive labels and two zero labels, be the positive labels next to each other or not. The label pattern of a new face is either all-positive, all-zero, one-zero, or one-positive.

3.2 U_sub: Unbalanced Catmull-Clark Subdivision

If a face has only one positive vertex label, an *unbalanced Catmull-Clark subdivision*, called *U_sub* for short, is performed on that face with respect to the vertex with positive label. A U_sub generates three new faces, instead of four, using six of the nine vertices used by B_sub and an old vertex. The six new vertices include a new face point, two new edge points and three new vertex points.

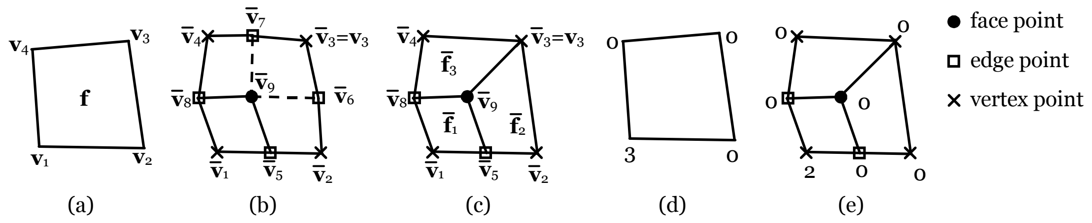

Fig. 3. Unbalanced Catmull-Clark subdivision with respect to $\mathbf{v}_1$: (a) before subdivision; (b) auxiliary structure stored after subdivision; (c) structure output after subdivision. Labels: (d) before; (e) after.

In Figure 3(a), if label of $\mathbf{v}_1$ is the only positive label of face $\mathbf{f}$, then the new face point $\bar{\mathbf{v}}_9$, the new vertex point $\bar{\mathbf{v}}_1$ and adjacent vertex points $\bar{\mathbf{v}}_2$ and $\bar{\mathbf{v}}_4$, and the adjacent new edge points $\bar{\mathbf{v}}_5$ and $\bar{\mathbf{v}}_8$ are used with $\mathbf{v}_3$ to form three new faces $\bar{\mathbf{f}}_1$, $\bar{\mathbf{f}}_2$ and $\bar{\mathbf{f}}_3$, as shown in Figure 3(c). The new edge points $\bar{\mathbf{v}}_6$ and $\bar{\mathbf{v}}_7$ are not used in the new face construction process. But these points and the *auxiliary structure* shown in Figure 3(b) will also be computed and recorded; these points are not only needed in the computation of the new vertex points $\bar{\mathbf{v}}_2$ and $\bar{\mathbf{v}}_4$, they are needed in the computation of the vertices of $\mathbf{M}_{k+1}$ as well. However, coordinates of these points and the other six new points will not be computed at the moment yet. Like in a B_sub, these vertices, except $\mathbf{v}_3$, will be marked with an "UPDATE" to indicate that they will be evaluated later. For consistency of notation, we will use $\bar{\mathbf{v}}_3$ to represent $\mathbf{v}_3$ in $\mathbf{M}_k$ subsequently. This will not cause any problem at all because $\bar{\mathbf{v}}_3=\mathbf{v}_3$ will not be marked with an "UPDATE" and, consequently, will not be evaluated as a "vertex point" at a later stage. Labels of all the new vertices are zero except $\bar{\mathbf{v}}_1$ whose label equals the old label minus one. Namely,

$$L(\bar{\mathbf{v}}_i)=0, \qquad \text{if } i=2, 3, 4, 5, 8, 9,$$

and

$$L(\bar{\mathbf{v}}_i)=L(\mathbf{v}_i)-1, \qquad \text{if } i=1.$$

If labels of vertices $\mathbf{v}_1$, $\mathbf{v}_2$, $\mathbf{v}_3$, and $\mathbf{v}_4$ in Figure 3(a) are 3, 0, 0 and 0, respectively (Figure 3(d)), then labels of the new vertices after a U_sub with respect to $\mathbf{v}_1$ are the ones shown in Figure 3(e). B_sub with respect to other vertices can be defined similarly.

After all the faces of $\mathbf{M}_{k\text{-}1}$ are processed, vertices marked with an "UPDATE" in $\mathbf{M}_k$ are computed using the Catmull-Clark subdivision scheme to find their new coordinates in $\mathbf{M}_k$. Vertices of $\mathbf{M}_k$ not marked with an "UPDATE" will be inherited from $\mathbf{M}_{k\text{-}1}$ directly. Note that the vertices of $\mathbf{M}_{k\text{-}1}$ required in the computation process for the new vertices are available because they are either explicitly generated in a B_sub or a U_sub, or are stored with an auxiliary structure for some U_sub (see Figure 3(b)). Setting up the status of a vertex with the mark "UPDATE" is necessary because whether a vertex should be inherited or updated depends on all its adjacent faces.

The adaptive subdivision process stops when labels of the mesh vertices are all zero. The resulting mesh not only satisfies the required precision for each face without doing excessive refinement, but also provides smooth, crack-free transition between faces with different subdivision depths. An example is shown in Figure 4(b). The input faces and their labels are shown in 4(a).

4. ELIMINATION OF ILLEGAL FACES

A mesh face **f** of $\mathbf{M}_0$ is called an *illegal face* if two adjacent vertices of **f** have positive labels and the other two adjacent vertices have zero labels. For instance, in Figure 4(c), two adjacent faces of the extra-ordinary vertex are illegal faces. A mesh refined by the adaptive subdivision process is not guaranteed to be crack free if it contains illegal faces. Consider again the case depicted in Figure 4(c). If the adaptive subdivision process is performed on this mesh then a crack would be generated between each illegal face with an adjacent face once the new face points and edge points are connected with the corresponding vertex points (see Figure 4(d)). Note that the adaptive subdivision process does not generate new illegal faces in the refining process. This is because if the label of a new face point or a new vertex point is zero then labels of its adjacent vertices in the refined mesh are all zero. Therefore, to avoid cracks in the refined meshes, one only needs to make sure illegal faces contained in the initial mesh $\mathbf{M}_0$ are removed before the adaptive subdivision process starts.

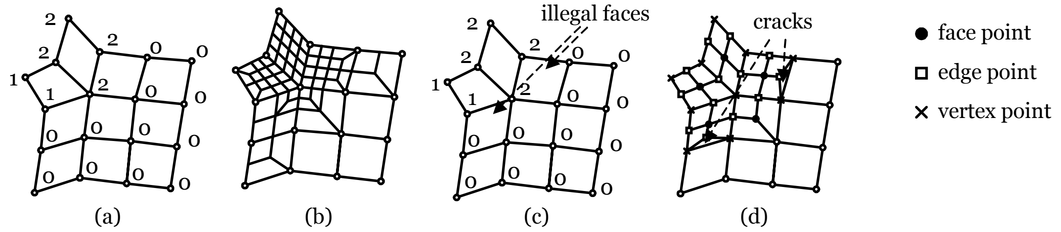

Fig. 4. (a) A mesh without illegal faces, (b) smooth, crack-free transition provided by U_sub, (c) a mesh with illegal faces, (d) cracks resulted from subdivision (only one iteration of subdivision is carried out).

The easiest way to remove illegal faces in $\mathbf{M}_0$ is to set all the zero labels to 1. But this would unnecessarily increase the number of faces generated in the final mesh. A better way is to eliminate all the illegal faces with as few zero labels changed to 1 as possible. This is in general not easy to achieve because although the process of changing the label of a vertex from 0 to 1 can make some illegal faces legal, it also can make some legal faces become illegal. Therefore, back tracking is constantly needed in the illegal face elimination process. An illegal face elimination technique for rectangular grids is presented in [4]. It removes the illegal faces by setting zero labels on every other diagonal line to 1. This approach does not work for meshes of arbitrary topology. In the following, a greedy algorithm is presented.

The algorithm eliminates the illegal faces via a *connection supporting graph* **G**. The vertices of **G** are those of the illegal faces whose labels are zero, called *illegal vertices* for short. The edges of **G** are those of $\mathbf{M}_0$ that connect vertices of **G**, i.e., illegal vertices. The algorithm repeatedly selects a vertex from **G**, changes its label to 1 and then updates **G** accordingly. This process continues until **G** is empty. The updating process includes removing vertices from **G** which are no longer illegal and adding new illegal vertices into **G**. During each cycle, the greedy algorithm will try to remove as many old vertices from **G** and introduce as few new vertices into **G** as possible. This is achieved by using the following rule in selecting a vertex from **G** to change label. Let $D(\mathbf{v})$ denote the degree of **v** in **G** and let $N(\mathbf{v})$ be the number of new vertices introduced into **G** if the label of **v** is changed from 0 to 1. If the number of $D(\mathbf{v})=1$ vertices is not zero then, in the pool of vertices which are adjacent to a $D(\mathbf{v})=1$ vertex, select any one with a minimum $N(\mathbf{v})$ among those with maximum $D(\mathbf{v})$. Otherwise, select any vertex with a minimum $N(\mathbf{v})$ among vertices of **G** with maximum $D(\mathbf{v})$.

5. LIMIT SURFACE OF ADAPTIVELY REFINED MESHES

We prove in this section that the adaptively refined meshes converge to the same lime surface as the uniformly refined meshes. We need a few properties in the proof.

First note that, from Figures 2 and 3, a face in $\mathbf{M}_k$ ($k \geq 1$) that requires further refinement, be it a B_sub or a U_sub, is always defined by a vertex point, two edge points and a face point. Such a face can be illustrated by the example given in Figure 5. Figure 5(a) shows faces in $\mathbf{M}_{k-1}$ and Figure 5(b) shows the result after a B_sub on the face $\mathbf{v}_1\ \mathbf{v}_4\ \mathbf{v}_5\ \mathbf{v}_6$ in Figure 5(a). If a U_sub is performed on $\mathbf{v}_1\ \mathbf{v}_4\ \mathbf{v}_5\ \mathbf{v}_6$, the example shown in Figure 5(b) is the auxiliary structure. The difference between the auxiliary structure and the output structure is, in the output structure, there is a diagonal edge (see Figure 3(c)). But such edge does not affect the analysis of the convergence to the limit surface. It should be pointed out that Figure 3 can be used for both ordinary faces and extraordinary faces. If N=4, the face is an ordinary face. Otherwise, it is an extraordinary face. According to the rules of adaptive subdivision process, a vertex marked with "UPDATE" is either a vertex in a *balanced Calmull-Clark subdivision structure* or a vertex in an *unbalanced Calmull-Clark subdivision structure*. Therefore, we only need to consider faces where a B_sub or a U_sub is to be performed.

Two properties of vertex labels will be proved below. These properties are quite obvious. But since they are needed in the proof of the theorem, we name them as lemmas for convenience of reference.

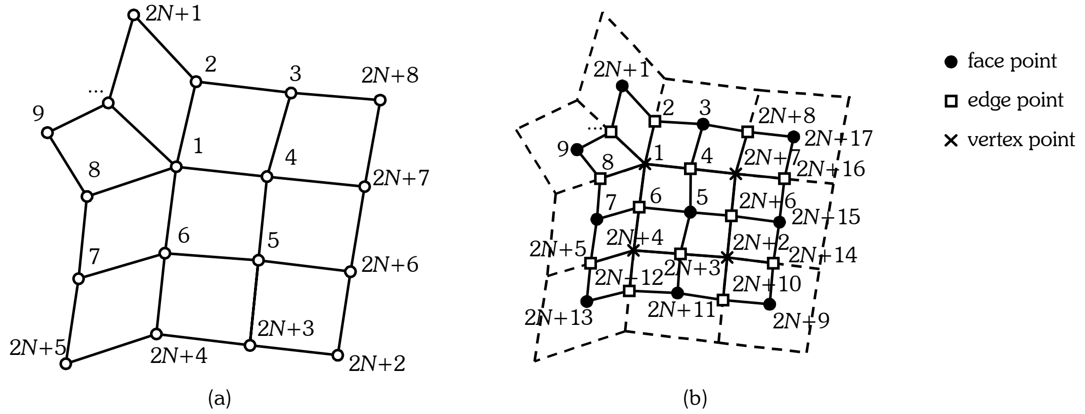

Fig. 5. Ordering of control points of a CCSS patch [22] (a) before subdivision, (b) after subdivision.

<u>Lemma 1:</u> Let $\overline{\mathbf{v}}_1$, $\overline{\mathbf{v}}_4$, $\overline{\mathbf{v}}_5$ and $\overline{\mathbf{v}}_6$ be the vertices of a mesh face in $\mathbf{M}_k$ ($k \geq 1$), as the one shown in Figure 5(b), where a B_sub is to be performed. If $\overline{\mathbf{v}}_1$ is generated as a vertex point during the kth subdivision step, then the label of $\overline{\mathbf{v}}_1$, $L(\overline{\mathbf{v}}_1)$, must be positive.

Proof. A B_sub is performed on a mesh face only if at least two of its vertex labels are positive. Therefore, at least two of the given face's labels: $L(\overline{\mathbf{v}}_1)$, $L(\overline{\mathbf{v}}_4)$, $L(\overline{\mathbf{v}}_5)$ and $L(\overline{\mathbf{v}}_6)$, are positive. Besides, this face must be the result of a B_sub. Consequently, $\overline{\mathbf{v}}_4$ and $\overline{\mathbf{v}}_6$ must be edge points generated during the kth subdivision step. Since the label of an edge point is smaller than or equal to the label of its adjacent vertex point, we must have $L(\overline{\mathbf{v}}_1) \geq L(\overline{\mathbf{v}}_4)$ and $L(\overline{\mathbf{v}}_1) \geq L(\overline{\mathbf{v}}_6)$. Hence, we must have $L(\overline{\mathbf{v}}_1)>0$ for, otherwise, we would have $L(\overline{\mathbf{v}}_1)=L(\overline{\mathbf{v}}_4)=L(\overline{\mathbf{v}}_6)=0$, conflicting to the fact that the face has at least two positive labels. This completes the proof of **Lemma 1**.

<u>Lemma 2:</u> Let $\overline{\mathbf{v}}_1$, $\overline{\mathbf{v}}_4$, $\overline{\mathbf{v}}_5$ and $\overline{\mathbf{v}}_6$ be the vertices of a face in $\mathbf{M}_k$ ($k \geq 1$), as the one shown in Figure 5(b), where a U_sub is to be performed. If $\overline{\mathbf{v}}_1$ is a vertex point generated during the kth subdivision step, then we must have $L(\overline{\mathbf{v}}_1)>0$ or $L(\overline{\mathbf{v}}_5)>0$.

Proof. Since a U_sub is to be performed on this face, one of the labels of the face: $L(\overline{\mathbf{v}}_1)$, $L(\overline{\mathbf{v}}_4)$, $L(\overline{\mathbf{v}}_5)$ or $L_v(\overline{\mathbf{v}}_6)$, must be positive. Suppose the face is the result of a subdivision performed on the face $\mathbf{v}_1\ \mathbf{v}_4\ \mathbf{v}_5\ \mathbf{v}_6$ as the one shown in Figure 5(a). Then $\overline{\mathbf{v}}_4$ and $\overline{\mathbf{v}}_6$ must be edge points and $\overline{\mathbf{v}}_5$ must be a face point generated during the subdivision process. Note that no matter the subdivision performed on $\mathbf{v}_1\ \mathbf{v}_4\ \mathbf{v}_5\ \mathbf{v}_6$ is a B_sub or U_sub, we have $L(\overline{\mathbf{v}}_1) \geq L(\overline{\mathbf{v}}_4)$ and $L(\overline{\mathbf{v}}_1) \geq L(\overline{\mathbf{v}}_6)$. Consequently, we can not have $L(\overline{\mathbf{v}}_4)>0$ or $L(\overline{\mathbf{v}}_6)>0$. This completes the proof.

We are ready to prove the main result of this section now.

<u>Theorem 1:</u> The adaptively refined meshes converge to the same limit surface as the meshes uniformly refined by the standard Catmull-Clark subdivision scheme.

Proof. Since the adaptive subdivision scheme uses the same formulas as the standard Catmull-Clark subdivision scheme to calculate new vertices, to prove this theorem, we only need to prove that, in the adaptive subdivision process, the vertices required in the new vertex computing formulas are always available and valid. The proof will be based on mathematical induction. For the first subdivision step, vertices required in the new vertex computing formulas are vertices of the initial mesh. Therefore, the induction is obviously satisfied. Assume the induction holds for the (k-1)st subdivision step, i.e., all the vertices required in the new vertex computing formulas for the (k-1)st subdivision step are available and valid. We need to prove that all the vertices needed in the new vertex computing formulas for the *k*th subdivision step are available and valid as well.

The new vertices that have to be computed in the *k*th subdivision step are those marked with "UPDATE" in the subdivision process. Each of such vertices is the result of a B_sub or a U_sub, as shown in Figures 2 and 3. Consider a new vertex generated by subdividing the face $\bar{\mathbf{f}} = \bar{\mathbf{v}}_1 \bar{\mathbf{v}}_4 \bar{\mathbf{v}}_5 \bar{\mathbf{v}}_6$, as the one shown in Figure 5(b) (Figure 5(b) now acts as $\mathbf{M}_{k-1}$), where $\bar{\mathbf{v}}_1$ is a vertex point and $\bar{\mathbf{v}}_5$ is a face point generated in the previous subdivision step. The corresponding vertex of $\bar{\mathbf{v}}_1$ in $\mathbf{M}_{k-2}$ is $\mathbf{v}_1$ in Figure 5(a). If the subdivision process performed on $\bar{\mathbf{f}}$ is a B_sub, then before the subdivision we must have $L_v(\bar{\mathbf{v}}_1)>0$, according to **Lemma 1**. However, the condition that $L(\bar{\mathbf{v}}_1)$ is positive means that a subdivision, B_sub or U_sub, has been performed on all the adjacent faces of $\mathbf{v}_1$ in the (*k*-1)st subdivision step, and $\bar{\mathbf{f}}$ is the result of a B_sub. Therefore, vertices $\bar{\mathbf{v}}_i$, $i=1, 2, ..., 2N+8$, would be marked "UPDATE" and computed in the (*k*-1)st subdivision step. But these are the vertices required in the new vertex computation process for the face $\bar{\mathbf{f}}$ in a B_sub. Therefore, all the vertices required in the new vertex computation process for $\bar{\mathbf{f}}$ in a B_sub are available and valid.

If the subdivision process performed on $\bar{\mathbf{f}}$ is a U_sub, then according to **Lemma 2**, we know that either $L(\bar{\mathbf{v}}_1)>0$ or $L(\bar{\mathbf{v}}_5)>0$. If $L(\bar{\mathbf{v}}_1)>0$ (i.e., $L(\bar{\mathbf{v}}_5)=0$), then all the adjacent faces of $\mathbf{v}_1$ in Figure 5(a) would be subdivided by a U_sub or a B_sub and $\bar{\mathbf{f}}$ is the result of a U_sub or a B_sub. No matter the subdivision process performed on face $\mathbf{v}_1 \mathbf{v}_4 \mathbf{v}_5 \mathbf{v}_6$ in Figure 5(a) is a U_sub or a B_sub, vertices $\bar{\mathbf{v}}_i$, $i=1, 2, ..., 2N+1, 2N+3, 2N+4, ..., 2N+8$, would be marked "UPDATE" and computed. These are the vertices required in the new vertex computation process for a U_sub performed on $\bar{\mathbf{f}}$ with respect to $\bar{\mathbf{v}}_1$.

If $L(\bar{\mathbf{v}}_5)>0$, then $\bar{\mathbf{f}}$ is the result of a B_sub in the (*k*-1)st subdivision step. Since $L(\bar{\mathbf{v}}_1)=0$, a subdivision might not be performed on the adjacent faces $\mathbf{v}_1 \mathbf{v}_8 \mathbf{v}_9 \mathbf{v}_{10}, ..., \mathbf{v}_1 \mathbf{v}_{2N} \mathbf{v}_{2N+1} \mathbf{v}_2$ of $\mathbf{v}_1$ in Figure 5(a), but a subdivision would be performed on the two adjacent faces $\mathbf{v}_1 \mathbf{v}_6 \mathbf{v}_7 \mathbf{v}_8$ and $\mathbf{v}_1 \mathbf{v}_2 \mathbf{v}_3 \mathbf{v}_4$ in the (*k*-1)st subdivision step to ensure $\bar{\mathbf{v}}_4$ and $\bar{\mathbf{v}}_6$ are shared by faces on both sides the shared edges. Therefore, vertices $\bar{\mathbf{v}}_i$, $i=1, 2, ..., 7, 8, 2N+2, 2N+3, ..., 2N+8$, would be marked "UPDATE" in the (*k*-1)st subdivision step and computed at the end of the (*k*-1)st subdivision step. But these are the vertices one needs in computing the new vertices for a U_sub on $\bar{\mathbf{f}}$ with respect to $\bar{\mathbf{v}}_5$. Hence, in either case, we have shown the vertices required in the new vertex computation process for $\bar{\mathbf{f}}$ in a U_sub are available and valid. Thus, the theorem is proved.

6. IMPLEMENTATION AND RESULTS

For a given initial mesh $\mathbf{M}_0$, the label-driven adaptive subdivision method proposed in this paper calculates the subdivision depth for each face in $\mathbf{M}_0$. And then, labels are assigned to all vertices in $\mathbf{M}_0$ according to Equation (1). The labels may lead to some illegal faces in $\mathbf{M}_0$, so the greed algorithm is used to remove the illegal faces. The adaptive subdivision process begins once all the illegal faces are removed from $\mathbf{M}_0$. The adaptive subdivision process stops when all labels are zero. Our adaptive subdivision process guarantees that all faces in $\mathbf{M}_0$ are subdivided in no less than the required subdivision depths, and the smooth transition between faces with different subdivision depths does not have any crack.

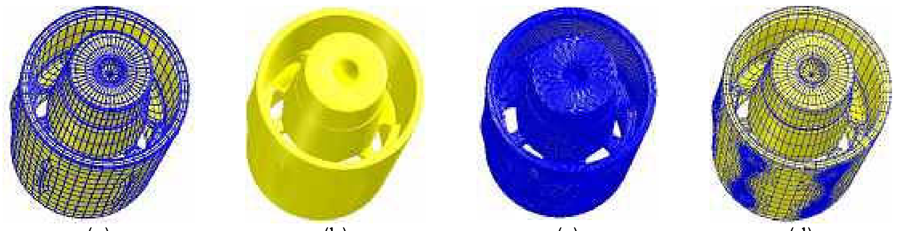

Fig. 6. Adaptive subdivision of a marker cap: (a) input mesh, (b) limit surface, (c) uniform subdivision, (d) adaptive subdivision.

Two examples shown in Figures 6 and 7 are used to compare the performance of adaptive subdivision with uniform Catmull-Clark subdivision. The first example is a marker cap. For an error tolerance of 0.1, the maximum subdivision depth of the mesh faces in the input control mesh is 3. Uniform Catmull-Clark subdivision (Figure 6(c)) in this case generates 273,398 vertices, 546,816 edges and 273,408 faces As shown in Figure 6(c), the edges are too crowded to be distinguished. The label-driven adaptive subdivision (Figure 6(d)) leads to 15,086 vertices, 30,192 edges and 15,096 faces only, an eighteen times improvement on the total number of vertices, edges and faces.

The second example is a rocker arm. For an error tolerance of 0.25, the maximum subdivision depth is 2. Uniform Catmull-Clark Subdivision (Figure 7(c)) leads to 22,656 vertices, 45,312 edges, and 22,656 faces; while the label-driven adaptive subdivision (Figure 7(d)) generates 2,706 vertices, 5,412 edges, and 2,706 faces only, i.e., only 3/25 of the total vertices, edges and faces required in the uniform case. When the error tolerance is 0.2, the maximum subdivision depth is 4. Uniform Catmull-Clark subdivision in this case leads to 362,496 vertices, 724,992 edges and 362,496 faces; while the label-driven adaptive subdivision generates 9,022 vertices, 18,044 edges and 9,022 faces only, a forty times improvement on the total number of vertices, edges and faces. As one can see in these figures, the unbalanced subdivision provides a way to build a smooth, crack-free transition between faces with different subdivision depths, therefore, avoids the need of performing the same level of subdivision on all the faces.

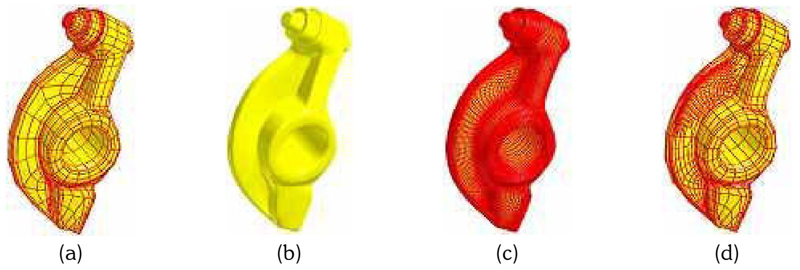

Fig. 7. Adaptive subdivision of a rocker arm: (a) input mesh, (b) limit surface, (c) uniform subdivision, (d) adaptive subdivision.

7. CONCLUSIONS

This paper presents an adaptive mesh refinement technique for cubic subdivision surfaces. The adaptive refinement process is driven by labels of mesh vertices instead of subdivision depths of the mesh faces. Smooth and crack-free transition between mesh faces with different subdivision depths is provided by an unbalanced subdivision process. The strengths of the new technique include: (1) significantly reduce the number of faces in the refined meshes in just a few (3-5) subdivision steps, (2) conformity (crack-free condition) of the resulting mesh is automatically guaranteed, (3) the refined meshes converge to the same limit surface, and (4) the input mesh can have arbitrary topology.

The limitation of this work is that the proposed adaptive subdivision technique currently work on subdivision surfaces of the Catmull-Clark style only. A future work is to extend the new technique to cover quadratic Doo-Sabin subdivision surfaces and subdivision surfaces with triangular mesh faces.

ACKNOWLEDGEMENTS

Work of the first author is supported by NSF of China (60403047) and FANEDD (200342). Work of the second author is supported by NSF (DMI-0422126).

8. REFERENCES

[1] Alliez, P. and Desbrun, M. Progressive Compression for Lossless Transmission of Triangle Meshes. In *Proceedings of SIGGRAPH* 2001, 195-202.

[2] Biermann, H., Kristjansson, D., and Zorin, D. 2001. Approximate Boolean Operations on Free-Form Solids. In *Proceedings of SIGGRAPH* 2001, 185-194.

[3] Catmull, E., and Clark, J. 1978. Recursively Generated B-spline Surfaces on Arbitrary Topological Meshes. *Computer-Aided Design* 10, 6, 350-355.

[4] Cheng, F., Jaromczyk, J.W., Lin, J.-R., Chang, S.-S., and Lu J.-Y. 1989. A Parallel Mesh Generation Algorithm Based on the Vertex Label Assignment Scheme. *International Journal for Numerical Methods in Engineering* 28, 1429-1448.

[5] Cheng, F. 1992. Estimating Subdivision Depths for Rational Curves and Surfaces. *ACM Trans. on Graphics* 11, 2 140-151.

[6] Cheng, F., Chen, G., Yong, J.-H., 2004. Subdivision Depth Computation for Catmull-Clark Subdivision Surfaces, preprint.

[7] DeRose, T., Kass, M., Truong, T. 1998. Subdivision Surfaces in Character Animation. In *Proceedings of SIGGRAPH* 1998, 85-94.

[8] Doo, D., and Sabin, M. 1978. Behavior of Recursive Division Surfaces near Extraordinary Points. *Computer-Aided Design* 10, 6, 356-360.

[9] Garland, M., and Heckbert, P. 1997. Surface Simplification Using Quadric Error Metrics. In *Proceedings of SIGGRAPH* 1997, 209-216.

[10] Garland, M. 1999. *Quadric-Based Polygonal Surface Simplification*. PhD thesis, Carnegie Mellon University.

[11] Hoppe, H. 1996. Progressive Meshes . In *Proceedings of SIGGRAPH* 1996, 99-108.

[12] Khodakovsky, A., Schroder, P., and Sweldens, W. 2000. Progressive Geometry Compression. In *Proceedings of SIGGRAPH* 2000, 271-278.

[13] Kobbelt, L. 1996. Interpolatory Subdivision on Open Quadrilateral Nets with Arbitrary Topology. *Computer Graphics Forum* 15, 3, 409-420.

[14] Kobbelt, L. 2000. $\sqrt{3}$ Subdivision. In *Proceedings of SIGGRAPH* 2000, 103-112.

[15] Lee, A., Moreton, H., and Hoppe, H. Displaced Subdivision Surfaces. In *Proceedings of SIGGRAPH* 2000, 85-94.

[16] Levin, A. 1999. Interpolating Nets of Curves by Smooth Subdivision Surfaces. In *Proceedings of SIGGRAPH* 1999, 57-64.

[17] Lindstrom, P. 2000. Out-of-Core Simplification of Large Polygonal Models. In *Proceedings of SIGGRAPH* 2000, 259-262.

[18] Litke, N., Levin, A., and Schroder, P. 2001. Trimming for Subdivision Surfaces. *Computer Aided Geometric Design* 18, 5, 463-481.

[19] Peters, J. Patching Catmull-Clark Meshes. In *Proceedings of SIGGRAPH* 2000, 255-258.

[20] Sederberg, T.W., Zheng, J., Sabin M., Sewell, D., 1998. Non-Uniform Recursive Subdivision Surfaces. In *Proceedings of SIGGRAPH* 1998, 387-394.

[21] Sederberg, T.W., Zheng, J., Bakenov, A., Nasri, A., 2003. T-Splines and T-NURCCs. In *Proceedings of SIGGRAPH* 2003, 477-484.

[22] Stam, J. 1998. Exact Evaluation of Catmull-Clark Subdivision Surfaces at Arbitrary Parameter Values. In *Proceedings of SIGGRAPH* 1998, 395-404.

[23] Velho, L., and Zorin, D. 2001. 4-8 subdivision. *Computer Aided Geometric Design* 18, 5, 397-427.

[24] Zorin, D., Schroder, P., and Sweldens, W. Interactive Multiresolution Mesh Editing. In *Proceedings of SIGGRAPH* 1997, 259-268.