Interpolation over Arbitrary Topology Meshes Using a Two-Phase Subdivision Scheme

Jianmin Zheng and Yiyu Cai

Abstract—The construction of a smooth surface interpolating a mesh of arbitrary topological type is an important problem in many graphics applications. This paper presents a two-phase process, based on a topological modification of the control mesh and a subsequent Catmull-Clark subdivision, to construct a smooth surface that interpolates some or all of the vertices of a mesh with arbitrary topology. It is also possible to constrain the surface to have specified tangent planes at an arbitrary subset of the vertices to be interpolated. The method has the following features: 1) it is guaranteed to always work and the computation is numerically stable, 2) there is no need to solve a system of linear equations and the whole computation complexity is $O(K)$ where K is the number of the vertices, and 3) each vertex can be associated with a scalar shape handle for local shape control. These features make interpolation using Catmull-Clark surfaces simple and, thus, make the new method itself suitable for interactive free-form shape design.

Index Terms—Computer graphics; computational geometry and object modeling; curve, surface, solid, and object representations; computer-aided engineering; computer-aided design.

1 Introduction

Modeling complex smooth surfaces is an important task in industrial design, geometric modeling, computer graphics, animation, and visualization. The surfaces might have complex topological structure; for example, arbitrary genus. Though nonuniform rational B-splines (NURBS) have been an industrial standard and are readily available in existing commercial modeling systems, it is usually difficult to construct such surfaces using NURBS because NURBS suffer from the topological restrictions of the control meshes. A patchwork of trimmed NURBS may be used instead. However, considerable effort is required to maintain the continuity at the connections of the patchwork [3].

Recursive subdivision was introduced as an efficient technique to model arbitrary topological surfaces. Starting from an initial polyhedral mesh, subdivision recursively refines the mesh by adding new vertices, edges, and faces. As the number of this process goes to infinity, the refined meshes finally converge to a smooth limit surface. A typical and favoring scheme is the Catmull-Clark [2], which is the generalization of bicubic B-splines. Since the 1990s, research on subdivision surfaces has been gaining rapid development in both algorithmic and mathematical aspects [24], [25]. These developments also move applications of subdivision surfaces forward.

Subdivision algorithms allow users to arrange control points in a way that naturally matches the geometric characteristics of the model without concern for maintaining a regular mesh structure. This makes subdivision surfaces attractive for interactive free-form surface design. It is, therefore, of particular interest to investigate theoretically safe and practically fast interpolation algorithms using subdivision surfaces. The goal of this paper is to develop a simple, fast, and reliable algorithm based on the Catmull-Clark subdivision scheme for constructing smooth interpolating surfaces of arbitrary topology. We choose to use Catmull-Clark subdivision surfaces because they are the generalization of bicubic B-splines, making them easier to use in conjunctions with existing modeling systems. The Catmull-Clark subdivision algorithm can handle arbitrarily topological meshes, including regular rectangular meshes, triangular meshes, and other nontriangular meshes.

1.1 Prior Work

Interpolation using subdivision surfaces can be achieved in two ways. The first approach is to use interpolating subdivision schemes. Dyn et al. [5] pioneered a so-called Butterfly scheme that interpolates the vertices of the input mesh. Zorin et al. [26] modified the subdivision rules of the Butterfly scheme to yield a smoother surface. These schemes are based on triangular meshes. Kobbelt [10] proposed an interpolatory subdivision scheme for quadrilateral nets. Levin [11] developed a combined subdivision scheme that interpolates a net of curves rather than a set of vertices.

There is a large class of subdivision algorithms that typically are generalizations of spline-based schemes. For example, the Doo-Sabin scheme [4] generalizes the knot insertion for biquadratic B-splines, the Catmull-Clark [2] generalizes bicubic B-splines, and the Loop [13] is the generalization of quartic box splines. Just like their spline counterparts, these schemes usually do not interpolate the vertices of the meshes. The second approach of achieving interpolation, therefore, is to do some modifications to force the limit surface to go through the vertices. Hoppe et al. [9]

- *J. Zheng is with the School of Computer Engineering, Nanyang Technological University, 50 Nanyang Avenue, Singapore 639798. E-mail: asjmzheng@ntu.edu.sg.*
- *Y. Cai is with the School of Mechanical and Aerospace Engineering, Nanyang Technological University, 50 Nanyang Avenue, Singapore 639798. E-mail:myycai@ntu.edu.sg.*

Manuscript received 27 Feb. 2005; revised 26 Aug. 2005; accepted 26 Sept., 2005; published online 10 Mar. 2006.
Recommended for acceptance by H. Qin.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org, and reference IEEECS Log Number TVCG-0024-0205.

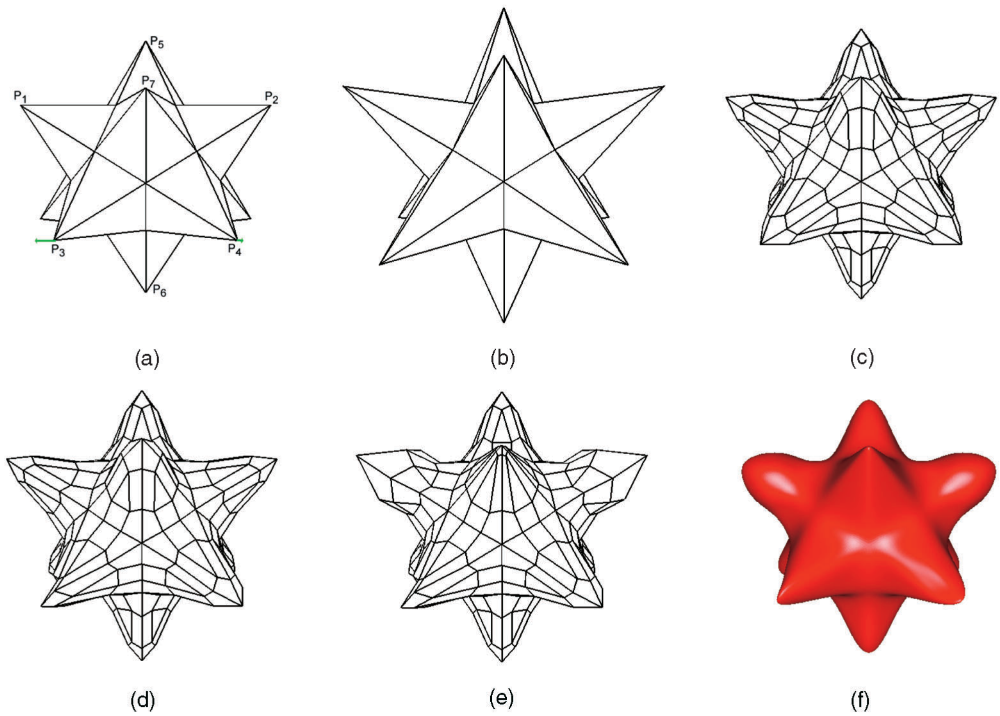

Fig. 1. Outline of the algorithm.

presented a modification of the Loop's algorithm. Nasri [14] presented a modification for the Doo-Sabin algorithm. Brunet [1] introduced a set of shape handles associated to the vertices for shape control in Nasri's approach. Halstead et al. [8] proposed an interpolation scheme using Catmull-Clark surfaces, which minimized a certain fairness measure. Both Nasri's and Halstead et al.'s methods had to construct a linear constraint on the control points of the initial mesh for each interpolating vertex and thus established a system of linear equations. The initial control mesh for the subdivision surface was obtained by solving the equations. However, it is unclear under what conditions the linear system is solvable [26]. As pointed out in [8], it is possible for the linear system to be singular or ill-conditioned. Besides, solving a large system of linear equations takes a considerable computational cost.

1.2 Contributions

In this paper, we present an interpolation algorithm using a modification of the Catmull-Clark subdivision scheme, which provably always works and does not require solving global linear systems. The basic idea is to generate an initial mesh using a fixed number of iterations for the linear equations, then to refine the mesh using a new subdivision rule to topologically separate vertices so that the supports of interpolated vertices are mutually disjunct, and finally to displace the vertices in the support to force interpolation and to provide additional local shape control. Our work combines some ideas of Nasri [14], [15], Brunet [1], and Halstead et al. [8] with new insights. There are also similarities between our method and Peters' patch-based schemes [17], [18], [19]. While our method focuses on subdivision surfaces for local interpolation over irregular meshes, Peters' work gives alternative techniques that directly construct geometrically continuous biquadratic/bicubic patches satisfying interpolation constraints. His methods do not need to solve a global system of equations either.

The whole procedure of our approach is illustrated in Fig. 1. In this paper, we call the input mesh (Fig. 1a) the *interpolating polyhedron*. It specifies the vertices $(P_1, \cdots, P_7)$ and tangent planes (two short green lines indicate the normal vectors of the tangent planes at P_3 and P_4) that need to be interpolated. We construct an initial control mesh $\widehat{M}^0$ (Fig. 1b) by a method developed in Section 3.1. Refining this control mesh using a set of new rules introduced in Section 2 gives an intermediate mesh (Fig. 1c), which is then perturbed to yield a refined mesh $\widehat{M}^1$ (Fig. 1d) where the superscript denotes the number of the refinement. The perturbation is used to assure the interpolation property. Note that the local structures corresponding to P_3 and P_4 have obvious changes due to the tangent-plane constraints. We may also use shape handles introduced in Section 4 to adjust $\widehat{M}^1$ for local shape control, resulting in an updated mesh (Fig. 1e). Here, the shape handles for P_1 and P_2 equal 1.5 and the shape handle for P_7 is 0.5. The limit surface (Fig. 1f) is then generated by applying the Catmull-Clark algorithm to (the possibly updated) $\widehat{M}^1$. It interpolates the required vertices and tangent planes.

Our main contributions include:

- We present a modification of the Catmull-Clark scheme. It is a two-phase subdivision process that uses a set of new rules for the first subdivision iteration, followed by the Catmull-Clark rules that are applied to the limit. The geometric intuition of introducing this two-phase subdivision process is to separate the influence region of each vertex of the input mesh so that we can adjust the newly created vertices locally to make the interpolation problem always work. Also, this modification provides the capability of increasing the appearance similarity between the initial control mesh and the limit surface by adjusting some scale numbers (known as the "blend ratio"). The two-phase process can be used to easily model fillets and blends.

- We develop an interpolation algorithm under the framework of our two-phase subdivision process. The algorithm is proven to always work.
- The algorithm complexity is $O(K)$ where K is the number of vertices of the mesh. No solution of a global linear system is required.
- A set of shape handles are introduced with this two-phase subdivision scheme. They can be used to locally adjust the shape of the limit surface.

1.3 Overview

In the next section, we review Catmull-Clark subdivision surfaces and present a modification of the Catmull-Clark scheme. Section 3 describes an algorithm for the construction of a subdivision surface that interpolates part or whole vertices of the input mesh. Section 4 shows how to introduce a set of shape handles with this interpolating scheme for the shape adjustment of the limit surface. Section 5 draws a conclusion. We restrict our discussion to closed meshes. Extension to open meshes is straightforward.

2 A Two-Phase Subdivision Scheme

A closed mesh we consider is a polyhedronlike configuration of faces, edges, and vertices such that each vertex corresponds to a point in 3D space, each edge is a line segment bounded by two vertices, and each face is bounded by a loop of edges. We also require that each edge is shared by exactly two faces and, in each loop, adjacent edges share a vertex. Note that in this definition, a face is defined topologically and its exact geometry is actually not specified. One may consider a face as lying in the convex hull of its edges [20]. The Catmull-Clark subdivision algorithm takes such a mesh and generates a smooth surface as the limit of the process of recursive refinement. The process for each refinement iteration includes:

1. For each face, compute a new face point as the average of all of the old points of the face.
2. For each edge, compute a new edge point as the average of the two old endpoints of the edges and the two new face points of the faces originally sharing the edge.
3. For each vertex, compute a new vertex point as a linear combination of the points within the neighborhood of the vertex. Specifically,

$$\frac{n-2}{n}V + \frac{1}{n^2}\sum_{j=1}^{n} E_j + \frac{1}{n^2}\sum_{j=1}^{n} F_j,$$

where n is the valence of the old vertex; V is the old vertex point; E_j are the endpoints, other than V, of all edges incident on the old vertex; and F_j are the face points of all faces sharing the old vertex.
4. Create new edges by connecting each new face point to the new edge points of the edges surrounding the face, and connecting each new vertex point to the new edge points of the edges incident on the old vertex.
5. Create new faces that have a loop of new edges.

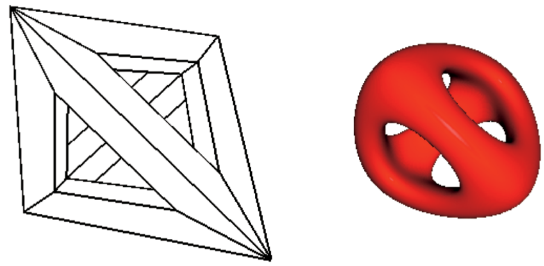

Fig. 2. A control mesh and the Catmull-Clark surface.

The above Steps 1–3 define the new geometry and Steps 4 and 5 define the connectivity. When this process step continues, it yields a sequence of refined meshes which eventually converges to a limit surface, known as the Catmull-Clark surface. An example of a Catmull-Clark surface with its initial control mesh is shown in Fig. 2.

The Catmull-Clark subdivision algorithm works on a mesh of arbitrary topological type. After the first refinement step, all faces in the refined mesh become quadrilateral, and the number of extraordinary vertices (i.e., vertices of valence other than 4) will remain constant in the subsequent subdivision steps. The limit surface gives rise to bicubic B-spline patches for all faces except those in the neighborhood of extraordinary points. Therefore, the limit surface is curvature-continuous except at the extraordinary vertices, where theoretical analysis has shown that the limit surface is tangent-plane-continuous. If there happens to be no extraordinary vertex (i.e., the initial mesh is regular), the limit surface is just a bicubic B-spline surface.

It is known that the control mesh of a B-spline surface roughly captures the shape of the surface. At times, however, designers expressed some dissatisfaction about the loose resemblance between the control mesh and the resulting surface. New schemes were proposed to improve the resemblance for cubic curves and surfaces [16], [23]. As the generalization of bicubic B-splines, Catmull-Clark surfaces behave similarly. Meanwhile, for subdivision surfaces, the top-level subdivision iterations play a key role on the overall shape of the limit surface. Based on these observations, we propose to use a simple two-phase subdivision process to increase the resemblance of Catmull-Clark surfaces to their control meshes. The basic idea is to use one set of new rules for the first subdivision iteration as the first phase, followed by the Catmull-Clark rules to the end as the second phase. The new rules are also based on the original Catmull-Clark subdivision rules. They keep the first three steps of the Catmull-Clark refinement. To avoid confusion, we call those new face, edge, and

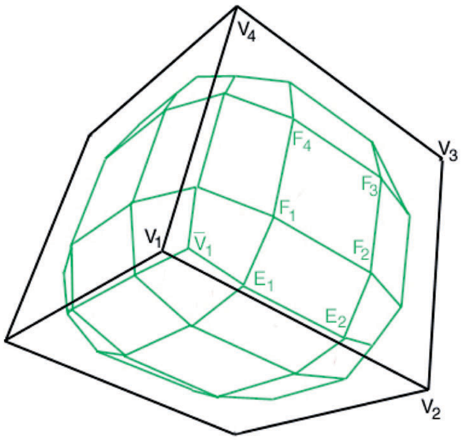

Fig. 3. The mesh after the first subdivision.

vertex points generated by the Catmull-Clark rules the intermediate face, edge, and vertex points. Then, the steps after Step 3 are as follows:

4. For each face with m surrounding vertices V_i, $i = 1, \cdots, m$, and an intermediate face point F, generate m new face points F_i:
$$F_i = \lambda V_i + (1 - \lambda)F,$$
where $\lambda \in [0, 1]$ is a certain constant. F_i can be considered as the image of V_i on the face.
5. For each edge with two endpoints V_1 and V_2, and an intermediate edge point E, compute two new edge points $E_i = \lambda V_i + (1 - \lambda)E$ for $i = 1, 2$. E_i can be considered as the image of V_i on the edge.
6. For each vertex V with an intermediate vertex point V', generate a new vertex point $\bar{V} = \lambda V + (1 - \lambda)V'$.
7. A new face of type F is created for each old face by connecting new face points $F_1, F_2, \cdots, F_m$.
8. Two four-sided new faces of type-E are created for each old edge by connecting the two new edge points E_1 and E_2 and connecting each new edge point E_i to the two new face points—the images of the corresponding old vertex V_i on the two adjacent faces sharing the edge.
9. n four-sided new faces of type-V are created for each old vertex of valence n by connecting the new vertex point to the images of the vertex on the surrounding edges (i.e., the corresponding new edge points).

An example of the mesh after the first subdivision is shown in Fig. 3.

Note that in the above process, the constant λ for each vertex could be different. But for simplicity, in this paper we assume there is only one λ for all vertices. A similar parameter, called *blend ratio*, was also introduced in Peters' method [18].

The modified algorithm retains many properties of the original Catmull-Clark algorithm. For example, the limit surface lies on the convex hull of the initial control mesh. This is because the new points are a convex combination of the old points. For $\lambda \in (0, 1)$, the modified rule will not alter the number of the non-four-valent vertices and the number of non-four-sided faces. Therefore, the number of the final extraordinary points will be the same. This means the continuity behavior will not be changed and depends only on the Catmull-Clark rules.

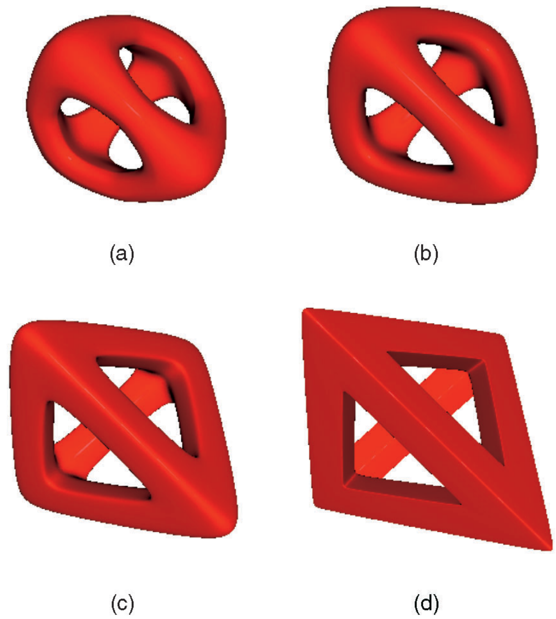

Fig. 4. Surfaces generated by the two-phase subdivision algorithm corresponding to different values of λ: (a) $\lambda = 0.1$, (b) $\lambda = 0.3$, (c) $\lambda = 0.6$, (d) $\lambda = 0.9$.

On the other hand, as $\lambda \to 1$, the similarity of the limit surface and the initial control mesh increases. Fig. 4 demonstrates the effects of different values of λ. In particular, when $\lambda = 1$, all type-V faces around a vertex degenerate to that vertex, all type-F faces are exactly the same as the corresponding old faces, and all type-E faces degenerate to the edges. The subsequent Catmull-Clark subdivision steps will not change the appearance of the mesh. Thus, the limit surface looks the same as the initial mesh. This effect cannot be achieved by the original Catmull-Clark algorithm. Another application of the modified algorithm is for fillet or blend operation on a polyhedral object [6], which is used quite often in CAD/CAM. We let the polyhedron be the initial mesh, and choose λ such that $1 - \lambda$ is a small positive number. Then, the limit surface will remain close to the original mesh but smooth out the sharp edges and vertices. [3] proposed another hybrid subdivision scheme to achieve so-called semishape creases.

3 The Surface Interpolation Method

We begin with considering *Interpolation Problem I*: Given an interpolating polyhedron $\widehat{P}$ with a set of vertices $P = \{P_i : i = 1, \cdots, K\}$ and a subset SP of P, construct a smooth surface interpolating the vertices in SP.

The basic idea of our approach is to construct another mesh $\widehat{M}^0$ with a set of vertices $M = \{M_i : i = 1, \cdots, K\}$, having the same topology as $\widehat{P}$, where the vertices M_i are to be determined. The sets P and M have a one-to-one correspondence. We perform the first-phase subdivision on $\widehat{M}^0$, yielding a refined mesh $\widehat{M}^1$. The second phase, using the Catmull-Clark algorithm, is then applied to $\widehat{M}^1$. For vertices M_i corresponding to the vertices in the compliment

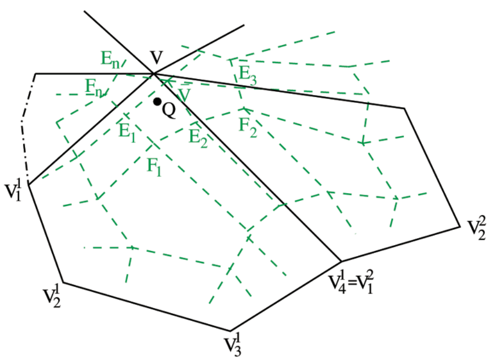

Fig. 5. The neigborhood around a vertex.

set of SP, we simply let them be the corresponding P_i. However, for those vertices M_i that correspond to the vertices P_i in SP, we require the limit points corresponding to M_i coincide with P_i. To assure this interpolation property, the spatial positions of the vertices in $\widehat{M}^1$ may need some rectification. In the following, we describe the details of our approach.

3.1 Interpolation Constraints

We consider one interpolating vertex in $\widehat{P}$ and its corresponding vertex in $\widehat{M}^0$ and the neighborhood. Without ambiguity, we denote the interpolating vertex by Q and the corresponding vertex $\widehat{M}^0$ by V. Refer to Fig. 5 for labels. Assume there are n faces meeting at V. We also denote the vertices of the neighborhood of V in mesh $\widehat{M}^0$ by $V_1^1, \cdots, V_{m_1-2}^1, V_{m_1-1}^1 (= V_1^2)$, $V_2^2, \cdots, V_{m_2-2}^2, V_{m_2-1}^2 (= V_1^3)$, $V_2^3, \cdots, V_{m_n-1}^n (= V_1^1)$, where m_i is the number of vertices of the ith face which has vertices $V, V_1^i, \cdots, V_{m_i-1}^i$. Note that for each V_1^i (or $V_{m_i-1}^i$), there is an edge between V and V_1^i (or $V_{m_i-1}^i$). For other V_j^i, there is no edge directly connecting to V. For convenience, we call vertex V and its neighborhood an *umbrella*, denoted by $V\text{–}V_1^1 V_2^1 \cdots V_{m_n-1}^n$. After the first phase subdivision, a smaller umbrella $\bar{V}\text{–}E_1 F_1 E_2 F_2 \cdots E_n F_n$ is generated, where $\bar{V}$ is the new vertex point, E_i are the new edge points, and F_i are the new face points. It is easy to check that these new points can be computed by

$$F_i = \left(\lambda + \frac{1-\lambda}{m_i}\right) V + (1-\lambda) \frac{1}{m_i} \sum_{j=1}^{m_i-1} V_j^i, \tag{1}$$

$$\begin{aligned} E_i =& \left[\frac{1+3\lambda}{4} + \frac{1-\lambda}{4}\left(\frac{1}{m_{i-1}} + \frac{1}{m_i}\right)\right] V + \frac{1-\lambda}{4} V_1^i \\ &+ \frac{1-\lambda}{4}\left(\frac{\sum_{j=1}^{m_{i-1}-1} V_j^{i-1}}{m_{i-1}} + \frac{\sum_{j=1}^{m_i-1} V_j^i}{m_i}\right), \end{aligned} \tag{2}$$

$$\begin{aligned} \bar{V} =& \left(\frac{n-2+2\lambda}{n} + \frac{1-\lambda}{n^2} \sum_{i=1}^{n} \frac{1}{m_i}\right) V + \frac{1-\lambda}{n^2} \sum_{i=1}^{n} V_1^i \\ &+ \frac{1-\lambda}{n^2} \sum_{i=1}^{n} \left(\frac{1}{m_i} \sum_{j=1}^{m_i-1} V_j^i\right). \end{aligned} \tag{3}$$

Note that the new umbrella has $2n+1$ vertices, forming n four-sided faces, and this topological structure will not change during the subsequent Catmull-Clark refinement. Using a discrete Fourier analysis, Halstead et al. showed that this umbrella converges to a limit point:

$$\frac{n^2 \bar{V} + 4\sum_{i=1}^{n} E_i + \sum_{i=1}^{n} F_i}{n(n+5)} \tag{4}$$

The interpolation condition can be satisfied by setting the above limit point to Q. Substituting (1), (2), and (3) into this condition leads to

$$\begin{aligned} n(n+5)Q =& \left(n^2 - n + 6n\lambda + 4(1-\lambda)\sum_{i=1}^{n} \frac{1}{m_i}\right) V \\ &+ 2(1-\lambda)\sum_{i=1}^{n} V_1^i + 4(1-\lambda)\sum_{i=1}^{n}\left(\frac{1}{m_i}\sum_{j=1}^{m_i-1} V_j^i\right). \end{aligned} \tag{5}$$

Grouping the equations for all interpolating vertices and the simple setting for noninterpolating vertices mentioned in the beginning of Section 3, we arrive at a system of linear equations with K equations and K unknowns. These are the constraints on the initial control mesh $\widehat{M}^0$.

Now, we give a sufficient condition under which the system of linear equations has a unique solution.

Theorem 1: *If λ is chosen to satisfy*

$$\lambda > \left(7n - n^2 - 8\sum_{i=1}^{n} \frac{1}{m_i}\right) \Big/ \left(12n - 8\sum_{i=1}^{n} \frac{1}{m_i}\right)$$

for all interpolating vertices, then the system of linear equations is diagonally dominant.

This can be easily proven by checking the coefficients of each equation. The coefficient of V is $n^2 - n + 6n\lambda + 4(1-\lambda)\sum_{i=1}^{n} \frac{1}{m_i}$, and the absolute values of the coefficients of all other unknowns sum to $6(1-\lambda)n - 4(1-\lambda)\sum_{i=1}^{n} \frac{1}{m_i}$. Therefore, when the condition in Theorem 1 holds, the coefficient of V is greater than the sum of the absolute values of all others and the linear system, thus, is diagonally dominant. Furthermore, note that n and all m_i are always greater than 2. The right side of the inequality in Theorem 1 is less than $(7n - n^2)/(12n - 8 \cdot n/3) = (21 - 3n)/28 \leq 3/7$. This gives a very conservative estimation for λ. If $\lambda \geq 3/7$, then for any configuration, the linear interpolation system has a unique solution. In practice, the valid lower bound for λ could be very small. However, our experience shows that the values from $[0.2, 0.5]$ for λ give a visually pleasing shape of the limit surface.

3.2 Computing the Initial Control Mesh

The initial control mesh $\widehat{M}^0$ can now be computed by directly solving the linear equations derived in Section 3.1. As an alternative, the Gauss-Seidel or the Jacobi iterative method is also often used, especially when the coefficient matrix is sparse, large, and diagonally dominant [7]. The convergence is guaranteed by the diagonal dominance.

Therefore, the iterative method is preferable in our case. We rewrite the iteration as

$$\begin{aligned} V = Cn(n+5)\,S - 2C(1-\lambda)\sum_{i=1}^{n} V_1^i \\ - 4C(1-\lambda)\sum_{i=1}^{n}\frac{1}{m_i}\sum_{j=1}^{m_i-1} V_j^i, \end{aligned} \tag{6}$$

where $C = 1\Big/\left(n^2 - n + 6n\lambda + 4(1-\lambda)\sum_{i=1}^{n}\frac{1}{m_i}\right)$. The initial setting for the iteration is naturally chosen to be the interpolating polyhedron. Once all updates for vertices are within a prescribed bound or the iterative number exceeds a predetermined one, the iteration stops and the current vertices define the initial control mesh.

Since we have perturbation steps in our algorithm, which are described in Section 3.3 and Section 3.4, we can choose a moderate number as the maximum iteration number. It does not matter if the updates are not within a prescribed tolerance after the maximum number of iterations. We experimented quite a few examples. Five iterations could give very small updates for all our examples. So, in our current implementation, we choose the maximum iteration number to be 5. This makes the algorithm's complexity be linear in number of the interpolating vertices. In addition, the iterative approach makes the programming task simple. The user can ignore the underlying mathematics. The right side of (6) is just a linear combination of vertices in the local neighborhood of V. This process is similar to the one in [12], retaining the flavor of digital geometric processing.

3.3 Perturbation for Position interpolation

In general, the initial control mesh $\widehat{M}^0$ obtained by the iterative approach is just an approximate solution. If the Catmull-Clark rules are applied immediately to mesh $\widehat{M}^1$ obtained from $\widehat{M}^0$ by the first-phase subdivision, the limit surface may not interpolate the specified vertices. It is, therefore, necessary to make some rectification to mesh $\widehat{M}^1$.

Note that after the first-phase subdivision, all type-V faces in $\widehat{M}^1$ corresponding to different interpolating vertices are separated. That is, the respective umbrellas in $\widehat{M}^1$ of the interpolating vertices are mutually disjunct. Therefore, we only need to study one interpolating vertex and its neighborhood. Consider an interpolating vertex Q and its corresponding umbrella $V{-}E_1F_1E_2F_2\cdots E_nF_n$. Let $W = [V, E_1, \cdots, E_n, F_1, \cdots, F_n]^T$ be the column vector of vertices of this umbrella and we also let W_i denote the ith element in this column vector. We further denote $m = 2n+1$ and $L_0 = [\alpha_1, \cdots, \alpha_m]$, where $\alpha_1 = n^2/n(n+5)$, $\alpha_2 = \cdots = \alpha_{n+1} = 4/n(n+5)$, and $\alpha_{n+2} = \cdots = \alpha_{2n+1} = 1/n(n+5)$. Then, the limit point corresponding to vertex V is $V^\infty = L_0 W = \sum_{i=1}^{m}\alpha_i W_i$. When $V^\infty \neq Q$, we give each vertex W_i a perturbation vector ε_i such that the perturbed umbrella will converge to Q: $Q = \sum_{i=1}^{m}\alpha_i(W_i + \varepsilon_i)$. The perturbations are determined by minimizing $\sum_{i=1}^{m}\varepsilon_i \cdot \varepsilon_i$. It is easy to obtain:

$$\varepsilon_i = -\frac{\alpha_i}{\sum_{j=1}^{m}\alpha_j^2}(V^\infty - Q). \tag{7}$$

3.4 Perturbation for Tangent-Plane Interpolation

In free-form shape desgin and modeling, the ability to specify tangent plane at the interpolated points is an attractive property for local shape control. Now, we state *Interpolation Problem II*: In addition to *Interpolation Problem I*, a set of unit normal vector N_j is given to be associated with a subset SSP of SP, and a surface is required not only to pass through the required vertices in SP but also to have tangent planes with given normal vectors at vertices in SSP.

According to the formulae provided by [8], the normal vector to the surface at the limit point is given by $N^\infty = (L_1 W) \times (L_2 W)$, where $L_1 = [\beta_1, \cdots, \beta_m]$ and $L_2 = [\gamma_1, \cdots, \gamma_m]$ are two row vectors. The elements of these two vectors are

$$\beta_1 = \gamma_1 = 0;$$

for $j = 2, \cdots, n+1$,

$$\beta_j = A_n \cos\frac{2\pi(j-1)}{n}, \quad \gamma_j = A_n \cos\frac{2\pi(j-2)}{n}$$

with

$$A_n = 1 + \cos\frac{2\pi}{n} + \cos\frac{\pi}{n}\sqrt{2\left(9 + \cos\frac{2\pi}{n}\right)};$$

and for $j = n+2, \cdots, 2n+1$,

$$\begin{aligned} \beta_j &= \cos\frac{2\pi(j-1)}{n} + \cos\frac{2\pi j}{n}, \\ \gamma_j &= \cos\frac{2\pi(j-2)}{n} + \cos\frac{2\pi(j-1)}{n}. \end{aligned}$$

Our approach is to find an initial control mesh $\widehat{M}^0$ by the iterative approach of Section 3.2, then to refine it to yield $\widehat{M}^1$ by the first-phase subdivision, and finally to perturb the vertices in $\widehat{M}^1$ so as to force the limit surface to interpolate both the positions and the tangent planes. Therefore, our task is to find the perturbation $\varepsilon = [\varepsilon_1, \cdots, \varepsilon_m]^T$ with the position constraint $L_0(W+\varepsilon) = Q$ and the tangent-plane constraint $[L_1(W+\varepsilon)] \times [L_2(W+\varepsilon)] \parallel N$. If we also want to minimize $\sum_{i=1}^{m}\varepsilon_i \cdot \varepsilon_i$, we come up with a constrained minimization problem:

$$\textit{minimize} \qquad \sum_{i=1}^{m}\varepsilon_i \cdot \varepsilon_i \tag{8}$$

$$s.t. \qquad V^\infty + \sum_{i=1}^{m}\alpha_i\varepsilon_i = Q \tag{9}$$

$$(L_1 W)\cdot N + \sum_{i=1}^{m}\beta_i(\epsilon_i \cdot N) = 0 \tag{10}$$

$$(L_2 W)\cdot N + \sum_{i=1}^{m}\gamma_i(\epsilon_i \cdot N) = 0. \tag{11}$$

To solve the above problem, we introduce Lagrange's multipliers Λ, μ_1, and μ_2, where Λ is a vector, and μ_1 and μ_2 are two scalar numbers. Letting

Fig. 6. (a) A triangular mesh and (b) the interpolation surface.

$$l_1 = (L_1 W)\cdot N = \sum_{i=1}^{m} \beta_i (W_i \cdot N), l_2 = (L_2 W)\cdot N$$
$$= \sum_{i=1}^{m} \gamma_i (W_i \cdot N),$$

and including the constraints into the objective function, we obtain a single unconstrained objective function

$$F = \sum_{i=1}^{m} \varepsilon_i \cdot \varepsilon_i + (V^\infty - Q)\cdot \Lambda + \sum_{i=1}^{m} \alpha_i (\varepsilon_i \cdot \Lambda) + \mu_1 l_1$$
$$+ \mu_1 \sum_{i=1}^{m} \beta_i (\varepsilon_i \cdot N) + \mu_2 l_2 + \mu_2 \sum_{i=1}^{m} \gamma_i (\varepsilon_i \cdot N).$$

Taking the partial derivatives of F with respect to ε_k and setting the derivatives to zero lead to

$$\frac{\partial F}{\partial \varepsilon_k} = 2\varepsilon_k + \alpha_k \Lambda + \mu_1 \beta_k N + \mu_2 \gamma_k N = 0, \quad k = 1, \cdots, m, \tag{12}$$

which gives

$$\varepsilon_k = -\frac{1}{2}(\alpha_k \Lambda + \mu_1 \beta_k N + \mu_2 \gamma_k N). \tag{13}$$

After some calculations, it can be verified that

$$\sum_{i=1}^{m} \alpha_i \beta_i = \sum_{i=1}^{m} \alpha_i \gamma_i = 0, \quad \sum_{i=1}^{m} \beta_i \gamma_i = C_n \cos\frac{2\pi}{n},$$
$$\sum_{i=1}^{m} \beta_i^2 = \sum_{i=1}^{m} \gamma_i^2 = C_n$$

where

$$C_n = \frac{n}{2}\left(A_n^2 + 2 + 2\cos\frac{2\pi}{n}\right).$$

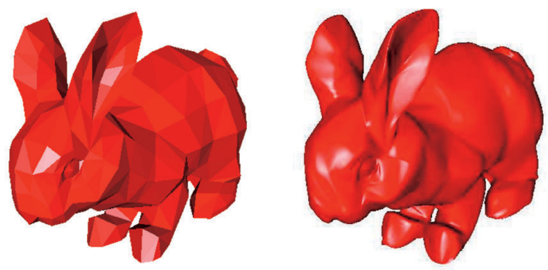

Fig. 7. (a) A triangular mesh and (b) the interpolation surface.

Substituting (13) into (9) gives

$$\Lambda = -\frac{2}{\sum_{i=1}^{m} \alpha_i^2}(Q - V^\infty). \tag{14}$$

Substituting (13) into (10) and (11) yields

$$2l_1 - \mu_1 \sum_{i=1}^{m} \beta_i^2 - \mu_2 \sum_{i=1}^{m} \beta_i \gamma_i = 0,$$
$$2l_2 - \mu_1 \sum_{i=1}^{m} \beta_i \gamma_i - \mu_2 \sum_{i=1}^{m} \gamma_i^2 = 0,$$

from which we can obtain

$$\mu_1 = \frac{2l_1 - 2\cos\frac{2\pi}{n} l_2}{C_n \sin^2\frac{2\pi}{n}},$$
$$\mu_2 = \frac{2l_2 - 2\cos\frac{2\pi}{n} l_1}{C_n \sin^2\frac{2\pi}{n}}.$$

We finally obtain the perturbation vectors

$$\varepsilon_k = \frac{\alpha_k}{\sum_{i=1}^{m} \alpha_i^2}(Q - V^\infty) - \frac{(l_1 - \cos\frac{2\pi}{n}\, l_2)\beta_k + (l_2 - \cos\frac{2\pi}{n} l_1)\gamma_k}{C_n \sin^2\frac{2\pi}{n}} N. \tag{15}$$

On the right side of the above equation, the first term is used to correct the position and the second term is for the normal vector.

3.5 Results

So far, we have described how to compute the initial control mesh $\widehat{M}^0$, how to perform the first phase subdivision, and how to adjust mesh $\widehat{M}^1$ so that the limit surface will interpolate the input mesh. We have implemented the whole algorithm using C++ under MS Windows. Fig. 1 demonstrates the process of the algorithm where λ is chosen to be $1/3$. Below, we present more results of applying the algorithm to a few models that were taken from the repository of "The Princeton Shape Benchmark" [21]. The test of the application was performed on a 1.6 GHz Intel Pentium 4 with 512 MB of RAM.

Figs. 6, 7, 8, and 9 show both the input meshes and the interpolation surfaces. The two models in Fig. 8 were obtained by applying Catmull-Clark subdivision to the original triangular meshes once and then moving each vertex to its limit position on the Catmull-Clark surfaces. In Fig. 9, the normal vector at each vertex of the mesh is also known. The surface shown on the right interpolates both

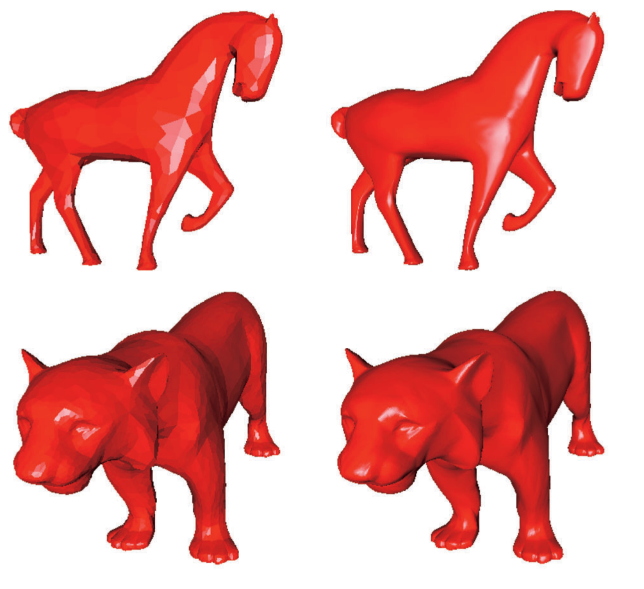

Fig. 8. (a) Two nontriangular meshes and (b) the interpolation surfaces.

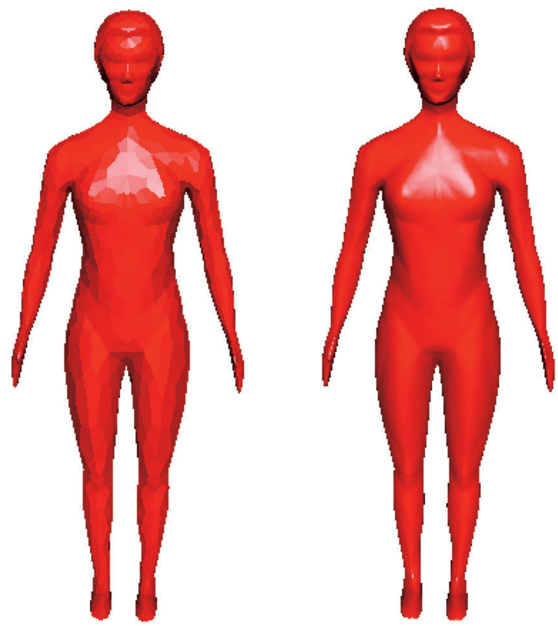

Fig. 9. (a) A mesh and (b) the surface that interpolates both positions and normals.

the vertices and the normal vectors. The statistics of the testing results are given in Table 1, which includes the valences the models have, the number of iterations required for solving the linear equations, the CPU time the algorithm took to compute the initial control mesh, and the maximum update for the vertices at the last iteration.

From the above examples, we have seen that our algorithm can quickly generate the initial control mesh. Once the initial control mesh is obtained, the interpolation surface is completely determined. To compute the interpolation surface, we may perform the following three processes: the first phase of subdivision, perturbation and the second phase of subdivision that is just the Catmull-Clark subdivision. An alternative approach is to apply only the first phase of subdivision and perturbation to the initial control mesh, yielding the updated $\widehat{M}^1$. This updated $\widehat{M}^1$ is actually the Catmull-Clark mesh. This means we can use Stam's method [22] to exactly evaluate any point on the final interpolation surface.

4 Further Shape Handles

To increase the capability of adjusting the shape of a Doo-Sabin subdivision surface, Brunet [1] defined a set of scalar shape handles associated to the initial vertices. These shape handles can be interactively modified to locally control the shape of the limit surface but do not affect the interpolatory properties. In this section, we show that Brunet's idea can be easily extended to our two-phase subdivision scheme.

For each vertex in the initial control mesh (or equivalently, in the interpolating polyhedron), we define a scalar number called the shape handle. These scalar shape handles are used to modify the spatial position of the vertices obtained after the first step of the subdivision plus possible perturbations described in Section 3. Assume the umbrella in $\widehat{M}^1$ corresponding to an initial vertex in $\widehat{M}^0$ is V–$E_1F_1\cdots E_nF_n$. We have known that its respective limit point V^∞ is $L_0[V, E_1, \cdots, E_n, F_1, \cdots, F_n]^T$. The shape handle S defines a geometric transformation for all vertices in this umbrella:

$$\begin{aligned}
V' &= V^\infty + S\,(V - V^\infty)\\
E'_j &= V^\infty + S\,(E_j - V^\infty), \quad j = 1, \cdots, n\\
F'_j &= V^\infty + S\,(F_j - V^\infty), \quad j = 1, \cdots, n.
\end{aligned}$$

The updated vertices will then be used in further steps of the Catmull-Clark subdivision. The shape handles have the following properties:

- The geometric transformation generally does not change the topological connectivity of the mesh. What are actually changed are only the spatial positions of the vertices.
- The geometric transformation will not affect the interpolatory properties. This follows from the identities

$$\begin{aligned}
&L_0[V', E'_1, \cdots, E'_n, F'_1, \cdots, F'_n]^T\\
&= L_0[V^\infty, \cdots, V^\infty]^T\\
&+ S\,(L_0[V, E_1, \cdots, E_n, F_1, \cdots, F_n]^T\\
&- L_0[V^\infty, \cdots, V^\infty]^T) = V^\infty
\end{aligned}$$

and

$$\begin{aligned}
&L_j[V', E'_1, \cdots, E'_n, F'_1, \cdots, F'_n]^T\\
&= S\,L_j[V, E_1, \cdots, E_n, F_1, \cdots, F_n]^T, \quad j = 1, 2.
\end{aligned}$$

- S has a scaling effect on the size of the type-V faces. When $S > 1$, the size of the type-V faces is increased. When $0 < S < 1$, the size is decreased.
- An analysis similar to that in [1] may be carried out to show that great values of S produce a flat spot at V^∞. The shape handles can be considered as tensionlike parameters that modify the distance from the surface to the tangent plane in the neighborhood of the initial vertices.

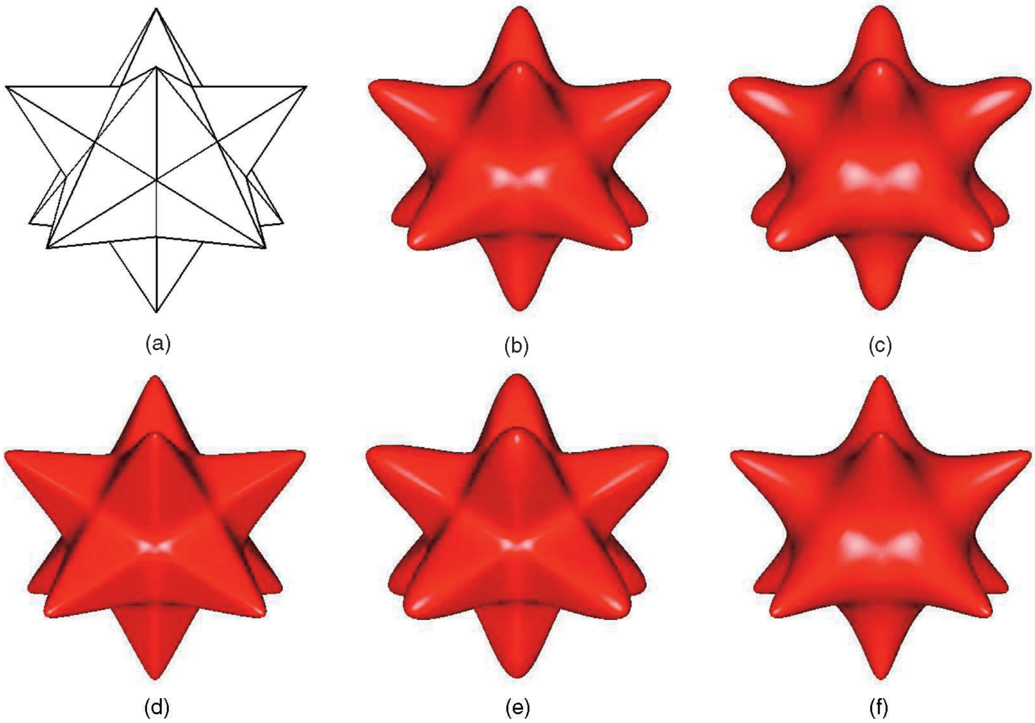

Fig. 10. Shape adjustment by shape handles (S_1 and S_2 stand for the shape handles for convex and novconvex vertices, respectively). (a) An input mesh. (b) $S_1 = S_2 = 1$. (c) $S_1 = S_2 = 1.3$. (d) $S_1 = S_2 = 0.6$. (e) $S_1 = 1.3$ and $S_2 = 0.6$. (f) $S_1 = 0.6$ and $S_2 = 1.3$.

In summary, the shape handles can be used either as design parameters, or to increase the smoothness of the surface. An example is given in Fig. 10, where the five limit surfaces (with different values of the shape handles) all interpolate the vertices of an input mesh shown in Fig. 10a. Here, we denote by S_1 the shape handles associated to the convex vertices and by S_2 the shape handles associated to the nonconvex vertices. Note that the shape handles just change the local shape and will not change the interpolation property.

5 Conclusions

We have described an algorithm for constructing interpolation surface of arbitrary topology. The algorithm is based on a two-phase subdivision process which is a modification of the Catmull-Clark scheme. Given an interpolating polyhedron, an iterative approach with a fixed number of iterations is adopted to compute an initial control mesh $\widehat{M}^0$. Then, the first-phase subdivision is applied to $\widehat{M}^0$ to create a refined mesh $\widehat{M}^1$. Perturbation schemes are developed to modify the spatial positions of $\widehat{M}^1$. We are also allowed to adjust some scalar shape handles to locally control the shape of the limit surface. Finally, the Catmull-Clark algorithm is used to generate the limit surface, which interpolates specified vertices and tangent planes. This approach is proven to always have solutions. There is no need to solve a system of linear equations. Therefore, it is simple and fast. These features make the method feasible to be used in interactive shape design.

Compared with other interpolatory subdivision schemes such as the Butterfly algorithm, our method needs an extra step that computes the initial control mesh. The experiments have shown that our method can quickly find the initial control mesh. Once the initial control mesh has been obtained, our method then works in a similar fashion as the Butterfly scheme. That is, some subdivision rules are used to refine the mesh. The Butterfly algorithm is for arbitrary topological triangular meshes. If an input mesh is not a triangular mesh, some preprocess is needed to convert it into a triangular mesh. Our method is for arbitrary

TABLE 1
The Statistics of the Experimental Results

	type of the mesh	number of vertices	valences	iteration number	time (ms)	max update (of model size)
Fig. 6	triangular	567	3,⋯,12	4	8.7	0.04%
Fig. 7	triangular	453	3,⋯,11	4	6.2	0.02%
Fig. 8 (horse)	quadrilateral	1388	3,⋯,11	3	16	0.05%
Fig. 8 (tiger)	quadrilateral	5726	3,⋯,11	3	69	0.02%
Fig. 9	quadrilateral	2084	3,⋯,9	3	24	0.03%

topological meshes that could be triangular or nontriangular. In particular, since our method is based on the Catmull-Clark scheme, in general it works even better for quadrilaterals. Therefore, our method is more suitable for applications that require surfaces that locally have two preferred directions and modeling systems that support tensor-product NURBS surfaces.

It is worth pointing out that the interpolating schemes are easy to produce excessive undulations. Therefore, interpolation is often combined with a scheme that minimizes a fairness norm to remove the undulation behavior. For example, Halstead et al. [8] constructed the interpolation surface that minimized a combination of thin plate and membrane energies. This resulted in solving a global linear system. This paper does not address the problem of global optimal fairness. We focus on developing a simple and safe algorithm. Our algorithm can be used to quickly produce an interpolating shape. It is possible that the resulting shape exhibits some undulations. If the quality of the shape is not satisfactory, further processing with more computational cost is needed. For instance, we have a lot of degrees of freedom and shape parameters in our two-phase scheme, so it could be possible to create a fair shape by properly setting them to minimize the fairness norm. This is a topic we are investigating.

Finally, in our two-phase scheme, we have actually two sets of shape parameters. One is the λ in the first phase subdivision, and the other is the shape handle. How to efficiently adjust them to achieve various desired effects in practical applications (such as in CAD/CAM) is an interesting topic for future work.

ACKNOWLEDGMENTS

This work is funded by a research grant (SUG 8/04) of Nanyang Technological University. The authors would also like to thank Singapore's Agency for Science, Technology, and Research for its support of this project.

REFERENCES

[1] P. Brunet, "Including Shape Handles in Recursive Subdivision Surfaces," *Computer Aided Geometric Design,* vol. 5, no. 1, pp. 41-50, June 1988.

[2] E. Catmull and J. Clark, "Recursively Generated B-Spline Surfaces on Arbitrary Topological Meshes," *Computer-Aided Design,* vol. 10, pp. 350-355, Sept. 1978.

[3] T. DeRose, M. Kass, and T. Truong, "Subdivision Surfaces in Character Animation," *Proc. SIGGRAPH '98},* pp. 85-94, 1998.

[4] D. Doo and M. Sabin, "Behaviour of Recursive Division Surfaces Near Extraordinary Points," *Computer-Aided Design,* vol. 10, pp. 356-360, Sept. 1978.

[5] N. Dyn, D. Levin, and J.A. Gregory, "A Butterfly Subdivision Scheme for Surface Interpolation with Tension Control," *ACM Trans. Graphics,* vol. 9, no. 2, pp. 160-169, Apr. 1990.

[6] P. Fjällström, "Smoothing of Polyhedral Models," *Proc. Second Ann. Symp. Computational Geometry,* pp. 226-235, 1986.

[7] G. Golub and C. van Loan, *Matrix Computations,* second ed. Baltimore, MD: The John Hopkins University Press, 1989.

[8] M. Halstead, M. Kass, and T. DeRose, "Efficient, Fair Interpolation Using Catmull-Clark Surfaces," *Proc. Computer Graphics (SIGGRAPH '93),* vol. 27, pp. 35-44, Aug. 1993.

[9] H. Hoppe, T. DeRose, T. Duchamp, M. Halstead, H. Jin, J. McDonald, J. Schweitzer, and W. Stuetzle, "Piecewise Smooth Surface Reconstruction," *Proc. Conf. Computer Graphics (SIGGRAPH '94),* pp. 295-302. 1994.

[10] L. Kobbelt, "Interpolatory Subdivision on Open Quadrilateral Nets with Arbitrary Topology," *Computer Graphics Forum,* vol. 15, no. 3, pp. 409-420, 485, Sept. 1996.

[11] A. Levin, "Interpolating Nets of Curves by Smooth Subdivision Surfaces," *Proc. Conf. Computer Graphics (SIGGRAPH '99),* pp. 57-64, Aug. 8-13, 1999.

[12] N. Litke, A. Levin, and P. Schröder, "Fitting Subdivision Surfaces," *Proc. Visualization 2001,* pp. 319-324. 2001.

[13] C. Loop, "Smooth Subdivision Surfaces Based on Triangles," master's thesis, Univ. of Utah, Dept. Mathematics, 1987.

[14] A. Nasri, "Polyhedral Subdivision Methods for Free-Form Surfaces," *ACM Trans. Graphics,* vol. 6, no. 1, pp. 29-73, Jan. 1987.

[15] A. Nasri, "Surface Interpolation on Irregular Networks with Normal Conditions," *Computer Aided Geometric Design,* vol. 8, no. 1, pp. 89-96, Feb. 1991.

[16] N.M. Ntoko, "A Generalisation of the Timmer Blending Functions for Bezier-Type Cubic Curves," *Proc. Eurographics '85,* C.E. Vandoni, ed., pp. 311-314, 1985.

[17] J. Peters, "Smooth Free-Form Surfaces Over Irregular Meshes Generalizing Quadratic Splines," *Computer Aided Geometric Design,* vol. 10, pp. 347-361, 1993.

[18] J. Peters, "Constructing C^1 Surfaces of Arbitrary Topology Using Biquadratic Bicubic Splines," *Designing Fair Curves and Surfaces,* N. Sapidis, ed., pp. 1-19. SIAM Publications, 1994.

[19] J. Peters, "C^1 Surface Spline," *SIAM J. Numerical Analysis,* vol. 32, 1995.

[20] M. Sabin, "Cubic Recursive Division with Bounded Curvature," *Curves and Surfaces,* P.J. Laurent, A.L. Méhauté, and L.L. Schumaker, eds., pp. 411-414. Boston: Academic Press, 1991.

[21] P. Shilane, P. Min, M. Kazhdan, and T. Funkhouser, "The Princeton Shape Benchmark," *Shape Modeling Int'l,* 2004.

[22] J. Stam, "Exact Evaluation of Catmull-Clark Subdivision Surfaces at Arbitrary Parameter Values," *Proc. SIGGRAPH '98,* pp. 395-404, 1998.

[23] H.G. Timmer, "Alternative Representation for Parametric Cubic Curves and Surfaces," *Computer-Aided Design,* vol. 12, pp. 25-28, 1980.

[24] J. Warren, J.D. Warren, and H. Weimer, *Subdivision Methods for Geometric Design: A Constructive Approach.* Morgan Kaufmann Publishers Inc., 2001.

[25] D. Zorin and P. Schröder, Subdivision for Modeling and Animation, *Course Notes SIGGRAPH '00,* 2000.

[26] D. Zorin, P. Schröder, and W. Sweldens, "Interpolating Subdivision for Meshes with Arbitrary Topology," *Computer Graphics,* Ann. Conf. Series, vol. 30, pp. 189-192, 1996.

Jianmin Zheng received the BS and PhD degrees from Zhejiang University, China. He is an assistant professor in the School of Computer Engineering at Nanyang Technological University, Singapore. Previously, he was a faculty member at Zhejiang University and a research faculty at Brigham Young University in Provo, Utah. His research interests include computer-aided geometric design, CAD/CAM, computer graphics, animation, digital imaging, and visualization.

Yiyu Cai received the BSc degree from Nanjing University, China, in 1983, the MSc degree from Zhejian University, China, in 1990, and the PhD degree from the Department of Mechanical and Production Engineering at the National University of Singapore in 1996. He is an associate professor with the School of Mechanical and Aerospace Engineering at Nanyang Technological University, Singapore. His research interests include CAD/CAM, geometric modeling, virtual simulation, and medical imaging.