Progressive interpolation using loop subdivision surface

Fuhua Cheng, Fengtao Fan, Shuhua Lai, Conglin Huang, Jiaxi Wang, Junhai yong

Progressive Interpolation Using Loop Subdivision Surfaces

Fuhua (Frank) Cheng[1], Fengtao Fan[1], Shuhua Lai[2], Conglin Huang[1], Jiaxi Wang[1], and Junhai Yong[3]

[1] Depart. of Computer Science, University of Kentucky, Lexington, KY 40506, USA
cheng@cs.uky.edu, {ffan2,Conglin.Huang,Jiaxi.Wang}@uky.edu
[2] Depart. of Mathematics and Computer Science, Virginia State University, Petersburg, VA 23806, USA
slai@vsu.eud
[3] School of Software, Tsinghua University, Beijing 100084, P.R. China
yongjh@tsinghua.edu.cn

Abstract. A new method for constructing interpolating Loop subdivision surfaces is presented. The new method is an extension of the progressive interpolation technique for B-splines. Given a triangular mesh M, the idea is to iteratively upgrade the vertices of M to generate a new control mesh $\bar{M}$ such that limit surface of $\bar{M}$ interpolates M. It can be shown that the iterative process is convergent for Loop subdivision surfaces. Hence, the method is well-defined. The new method has the advantages of both a local method and a global method, i.e., it can handle meshes of any size and any topology while generating smooth interpolating subdivision surfaces that faithfully resemble the shape of the given meshes.

Keywords: Geometric Modeling, Loop Subdivision Surface, Interpolation.

1 Introduction

Subdivision surfaces are becoming popular in many areas such as animation, geometric modeling and games because of their capability in representing any shape with only one surface. A subdivision surface is generated by repeatedly refining a control mesh to get a limit surface. Hence, a subdivision surface is determined by the way the control mesh is refined, i.e., the *subdivision scheme*. A subdivision scheme is called an *interpolating scheme* if the limit surface interpolates the given mesh. Otherwise, it is called an *approximating scheme*. Popular subdivision schemes such as Catmull-Clark [2], Doo-Sabin [1], and Loop schemes [8] are approximating schemes while the Butterfly [6], the improved Butterfly [9] and the Kobbelt schemes [18] are interpolating schemes.

An interpolating subdivision scheme generates new vertices by performing local affine combinations on nearby vertices. This approach is simple and easy to implement. It can handle meshes with a large number of vertices. However, since no vertex is ever moved once it is computed, any distortion in the early stage of

the subdivision will persist. This makes interpolating subdivision schemes very sensitive to irregularity in the given mesh. In addition, it is difficult for this approach to interpolate normals or derivatives.

On the other hand, even though subdivision surfaces generated by approximating subdivision schemes do not interpolate their control meshes, it is possible to use this approach to generate a subdivision surface to interpolate the vertices of a given mesh. One method, called *global optimization*, does the work by building a global linear system with some fairness constraints to avoid undesired undulations [3,4]. The solution to the global linear system is a control mesh whose limit surface interpolates the vertices of the given mesh. Because of its *global property*, this method generates smooth interpolating subdivision surfaces that resemble the shape of the given meshes well. But, for the same reason, it is difficult for this method to handle meshes with a large number of vertices.

To avoid the computational cost of solving a large system of linear equations, several other methods have been proposed. A two-phase subdivision method that works for meshes of any size was presented by Zheng and Cai for Catmull-Clark scheme [5]. A method proposed by Lai and Cheng [15] for Catmull-Clark subdivision scheme avoids the need of solving a system of linear equations by utilizing the concept of similarity in the construction process. Litke, Levin and Schöder avoid the need of solving a system of linear equation by quasi-interpolating the given mesh [7]. However, a method that has the advantages of both a local method and a global method is not available yet.

In this paper a new method for constructing a smooth Loop subdivision surface that interpolates the vertices of a given triangular mesh is presented. The new method is an extension of the progressive interpolation technique for B-splines [10,12,13]. The idea is to iteratively upgrade the locations of the given mesh vertices until a control mesh whose limit surface interpolates the given mesh is obtained. It can be proved that the iterative interpolation process is convergent for Loop subdivision surfaces. Hence, the method is well-defined for Loop subdivision surfaces. The limit of the iterative interpolation process has the form of a global method. But the control points of the limit surface can be computed using a local approach. Therefore, the new technique enjoys the advantages of both a local method and a global method, i.e., it can handle meshes of any size and any topology while generating smooth interpolating subdivision surfaces that faithfully resemble the shape of the given meshes.

The remaining part of the paper is arranged as follows. The concept of progressive interpolation for Loop subdivision surfaces is presented in Section 2. The convergence of this iterative interpolation process is proven in Section 3. Implementation issues and test results are presented in Section 4. Concluding remarks are given in Section 5.

2 Progressive Interpolation Using Loop Surfaces

Given a 3D triangular mesh $M = M^0$. To interpolate the vertices of M^0 with a Loop subdivision surface, one needs to find a control mesh $\bar{M}$ whose Loop

surface passes through all the vertices of M^0. Instead of a direct approach, we use an iterative process to do the job.

First, we consider the Loop surface $\mathbf{S}^0$ of M^0. For each vertex $\mathbf{V}^0$ of M^0, we compute the distance between this vertex and its limit point $\mathbf{V}^0_\infty$ on $\mathbf{S}^0$, $\mathbf{D}^0 = \mathbf{V}^0 - \mathbf{V}^0_\infty$, and add this distance to $\mathbf{V}^0$ to get a new vertex $\mathbf{V}^1$, $\mathbf{V}^1 = \mathbf{V}^0 + \mathbf{D}^0$. The set of all the new vertices is called M^1. We then consider the Loop surface $\mathbf{S}^1$ of M^1 and repeat the same process.

In general, if $\mathbf{V}^k$ is the new location of $\mathbf{V}^0$ after k iterations of the above process and M^k is the set of all the new $\mathbf{V}^k$'s, then we consider the Loop surface $\mathbf{S}^k$ of M^k. We first compute the distance between $\mathbf{V}^0$ and the limit point $\mathbf{V}^k_\infty$ of $\mathbf{V}^k$ on $\mathbf{S}^k$

$$\mathbf{D}^k = \mathbf{V}^0 - \mathbf{V}^k_\infty. \tag{1}$$

We then add this distance to $\mathbf{V}^k$ to get $\mathbf{V}^{k+1}$ as follows:

$$\mathbf{V}^{k+1} = \mathbf{V}^k + \mathbf{D}^k. \tag{2}$$

The set of new vertices is called M^{k+1}.

This process generates a sequence of control meshes M^k and a sequence of corresponding Loop surfaces $\mathbf{S}^k$. $\mathbf{S}^k$ converges to an interpolating surface of M^0 if the distance between $\mathbf{S}^k$ and M^0 converges to zero. Therefore the key task here is to prove that $\mathbf{D}^k$ converges to zero when k tends to infinity. This will be done in the next section.

Note that for each iteration in the above process, the main cost is the computation of the limit point $\mathbf{V}^k_\infty$ of $\mathbf{V}^k$ on $\mathbf{S}^k$. For a Loop surface, the limit point of a control vertex $\mathbf{V}$ with valence n can be calculated as follows:

$$\mathbf{V}_\infty = \beta_n \mathbf{V} + (1 - \beta_n)\mathbf{Q} \tag{3}$$

where

$$\beta_n = \frac{3}{11 - 8 \times \left(\frac{3}{8} + (\frac{3}{8} + \frac{1}{4}\cos\frac{2\pi}{n})^2\right)}, \qquad \mathbf{Q} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{Q}_i.$$

$\mathbf{Q}_i$ are adjacent vertices of $\mathbf{V}$. This computation involves nearby vertices only. Hence the progressive interpolation process is a local method and, consequently, can handle meshes of any size.

Another point that should be pointed out is, even though this is an iterative process, one does not have to repeat each step strictly. By finding out when the distance between M^0 and $\mathbf{S}^k$ would be smaller than the given tolerance, one can go directly from M^0 to M^k, skipping the testing steps in between.

3 Convergence of the Iterative Interpolation Process

We need the following lemma for the proof.

Lemma 1. *Eigenvalues of the product of positive definite matrices are positive.*

The proof of Lemma 1 follows immediately from the fact that if P and Q are square matrices of the same dimension, then PQ and QP have the same eigenvalues (see, e.g., [16], p.14).

To prove the convergence of the iterative interpolation process for Loop subdivision surfaces, note that at the $(k+1)$st step, the difference $\mathbf{D}^{k+1}$ can be written as:

$$\mathbf{D}^{k+1} = \mathbf{V}^0 - \mathbf{V}_\infty^{k+1} = \mathbf{V}^0 - \left(\beta_n \mathbf{V}^{k+1} + (1-\beta_n)\mathbf{Q}^{k+1}\right)$$

where $\mathbf{Q}^{k+1}$ is the average of the n adjacent vertices of $\mathbf{V}^{k+1}$

$$\mathbf{Q}^{k+1} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{Q}_i^{k+1}$$

By applying (2) to $\mathbf{V}^{k+1}$ and each $\mathbf{Q}_i^{k+1}$,

$$\mathbf{Q}_i^{k+1} = \mathbf{Q}_i^k \ + \ \mathbf{D}_{\mathbf{Q}_i^k}^k$$

we get

$$\begin{aligned}\mathbf{D}^{k+1} &= \mathbf{V}^0 - \left(\beta_n \mathbf{V}^k + (1-\beta_n)\mathbf{Q}^k\right) - \left(\beta_n \mathbf{D}^k + \tfrac{1-\beta_n}{n}\textstyle\sum_{i=1}^{n} \mathbf{D}_{\mathbf{Q}_i}^k\right) \\ &= \mathbf{D}^k - \left(\beta_n \mathbf{D}^k + \tfrac{1-\beta_n}{n}\textstyle\sum_{i=1}^{n} \mathbf{D}_{\mathbf{Q}_i}^k\right)\end{aligned}$$

where $\mathbf{Q}^k$ is the average of the n adjacent vertices of $\mathbf{V}^k$. In matrix form, we have

$$\left[\mathbf{D}_1^{k+1}, \mathbf{D}_2^{k+1}, \ldots, \mathbf{D}_m^{k+1}\right]^T = (I-B)\begin{bmatrix}\mathbf{D}_1^k \\ \mathbf{D}_2^k \\ \vdots \\ \mathbf{D}_m^k\end{bmatrix} = (I-B)^{k+1}\begin{bmatrix}\mathbf{D}_1^0 \\ \mathbf{D}_2^0 \\ \vdots \\ \mathbf{D}_m^0\end{bmatrix}$$

where m is the number of vertices in the given matrix, I is an identity matrix and B is a matrix of the following form:

$$\begin{pmatrix} \beta_{n_1} & \cdots & \frac{1-\beta_{n_1}}{n_1} & \cdots \\ \vdots & \ddots & & \\ \frac{1-\beta_{n_i}}{n_i} & \cdots & \beta_{n_i} & \cdots \\ \vdots & & & \beta_{n_m} \end{pmatrix}$$

The matrix B has the following properties:

(1) $b_{ij} \geq 0$, and $\sum_{j=1}^{n} b_{ij} = 1$ (hence, $\|B\|_\infty = 1$);
(2) There are $n_i + 1$ positive elements in the i-th row, and the positive elements in each row are equal except the element on the diagonal line.
(3) If $b_{ij} = 0$, then $b_{ji} = 0$;

Properties (1) and (2) follow immediately from the formula of $\mathbf{D}^{k+1}$ in eq. (3). Property (3) is true because if a vertex $\mathbf{V}_i$ is an adjacent vertex to $\mathbf{V}_j$ then $\mathbf{V}_j$ is obviously an adjacent vertex to $\mathbf{V}_i$. Due to these properties, we can write the matrix B as

$$B = DS$$

where D is a diagonal matrix and S is a symmetric matrix:

$$\mathbf{D} = \begin{pmatrix} \frac{1-\beta_{n_1}}{n_1} & 0 & \dots & 0 \\ 0 & \frac{1-\beta_{n_2}}{n_2} & \dots & 0 \\ \vdots & & \ddots & \\ 0 & & & \frac{1-\beta_{n_m}}{n_m} \end{pmatrix}, \quad \mathbf{S} = \begin{pmatrix} \frac{n_1\beta_{n_1}}{1-\beta_{n_1}} & \dots & 1 & \dots \\ \vdots & \ddots & & \\ 1 & \dots & \frac{n_i\beta_{n_i}}{1-\beta_{n_i}} & \dots \\ \vdots & & & \frac{n_m\beta_{n_m}}{1-\beta_{n_m}} \end{pmatrix}.$$

D is obviously positive definite. We will show that the matrix S is also positive definite, a key point in the convergence proof.

Theorem 1. *The matrix S is positive definite.*

Proof: To prove S is positive definite, we have to show the quadric form

$$f(x_1, x_2, \dots, x_m) = \mathbf{X}^T S \mathbf{X}$$

is positive for any non-zero $\mathbf{X} = (x_1, x_2, \dots, x_m)^T$.

Note that if vertices $\mathbf{V}_i$ and $\mathbf{V}_j$ are the endpoints of an edge e_{ij} in the mesh, then $s_{ij} = s_{ji} = 1$ in the matrix S. Hence, it is easy to see that

$$f(x_1, x_2, \dots, x_n) = \sum_{e_{ij}} 2x_i x_j + \sum_{i=1}^{m} \frac{n_i\beta_{n_i}}{1-\beta_{n_i}} x_i^2$$

where e_{ij} in the first term ranges through all edges of the given mesh. On the other hand, if we use f_{ijr} to represent a face with vertices $\mathbf{V}_i$, $\mathbf{V}_j$ and $\mathbf{V}_r$ in the mesh, then since an edge in a closed triangular mesh is shared by exactly two faces, the following relationship holds:

$$\sum_{f_{ijr}} (x_i + x_j + x_r)^2 = \sum_{e_{ij}} 4x_i x_j + \sum_{i=1}^{m} n_i x_i^2$$

where f_{ijr} on the left hand side ranges through all faces of the given mesh. The last term in the above equation follows from the fact that a vertex with valence n is shared by n faces of the mesh. Hence, $f(x_1, x_2, \dots, x_n)$ can be espressed as

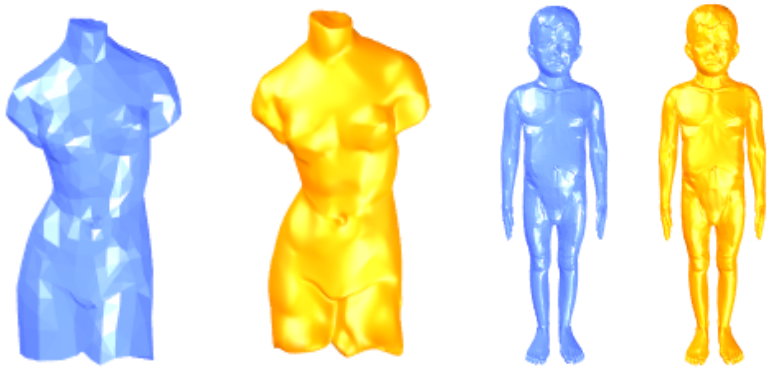
$$f(x_1, x_2, \dots, x_n) = \sum_{f_{ijr}} \frac{1}{2} (x_i + x_j + x_r)^2 + \sum_{i=1}^{m} \left(\frac{n_i\beta_{n_i}}{1-\beta_{n_i}} - \frac{n_i}{2} \right) x_i^2$$

From eq. (3), it is easy to see that $\frac{n\beta_n}{1-\beta_n} \geq \frac{3}{5}n$ for $n \geq 3$. Hence, $f(x_1, x_2, \dots, x_m)$ is positive for any none zero $\mathbf{X}$ and, consequently, S is positive definite.

Based on the above lemma and theorem, it is easy to conclude that the iterative interpolation process for Loop subdivision is convergent.

(a) Given mesh (b) I-surface (c) Given mesh

(d) I-surface (e) Given mesh (f) I-surface



(g) Given mesh (h) I-surface (i) G-mesh (j) I-surface

(k) Given mesh (l) I-surface (m) Given mesh (n) I-surface

Fig. 1. Examples of progressive interpolation using Loop subdivision surfaces (G-mesh ≡ given mesh; I-surface ≡ interpolating Loop surface)

Theorem 2. *The iterative interpolation process for Loop subdivision surface is convergent.*

Proof: The iterative process is convergent if and only if absolute value of the eigenvalues of the matrix $P = I - B$ are all less than 1, or all eigenvalues $\lambda_i,\ 1 \leq i \leq m$, of B are $0 < \lambda_i \leq 1$.

Since $\|B\|_\infty = 1$, we have $\lambda_i \leq 1$. On the other hand, since B is the product of two positive definite matrices D and S, following Lemma 1, all its eigenvalues must be positive. Hence, the iterative process is convergent.

4 Results

The progressive interpolation process is implemented for Loop subdivision surfaces on a Windows platform using OpenGL as the supporting graphics system. Quite a few cases have been tested. Some of them (a hog, a rabbit, a tiger, a statue, a boy, a turtle and a bird) are presented in Figure 1. All the data sets are normalized, so that the bounding box of each data set is a unit cube. For each case, the given mesh and the constructed interpolating Loop surface are shown. The sizes of the data meshes, numbers of iterations performed, maximum and average errors of these cases are collected in Table 1.

Table 1. Loop surface based progressive interpolation: test results

Model	# of vertices	# of iterations	Max Error	Ave Error
Hog	606	10	0.000870799	0.000175255
Rabbit	453	13	0.000932857	0.000111197
Tiger	956	9	0.000720453	0.00014148
Statue	711	11	0.000890806	0.000109163
Boy	17342	6	0.000913795	0.000095615
Turtle	445	10	0.000955765	0.0001726
Bird	1129	9	0.000766811	0.000088345

From these results, it is easy to see that the progressive interpolation process is very efficient and can handle large meshes with ease. This is so because of the expotential convergence rate of the iterative process. Another point that can be made here is, although no fairness control factor is added in the progressive iterative interpolation, the results show that it can produce visually pleasing surface easily.

5 Concluding Remarks

A progressive interpolation technique for Loop subdivision surfaces is presented and its convergence is proved. The limit of the iterative interpolation process has the form of a global method. Therefore, the new method enjoys the strength of a global method. On the other hand, since control points of the interpolating

surface can be computed using a local approach, the new method also enjoys the strength of a local method. Consequently, we have a subdivision surface based interpolation technique that has the advantages of both a local method and a global method. Our next job is to investigate progressive interpolation for Catmull-Clark and Doo-Sabin subdivision surfaces.

Acknowledgments. This work is supported by NSF (DMI-0422126). The last author is supported by NSFC(60625202, 60533070). Triangular meshes used in this paper are downloaded from the Princeton Shape Benchmark [17].

References

1. Doo, D., Sabin, M.: Behaviour of recursive division surfaces near extraordinary points. Computer-Aided Design 10, 356–360 (1978)
2. Catmull, E., Clark, J.: Recursively generated B-spline surfaces on arbitrary topological meshes. Computer-Aided Design 10, 350–355 (1978)
3. Halstead, M., Kass, M., DeRose, T.: Efficient, fair interpolation using Catmull-Clark surfaces. In: Proc.SIGGRAPH, pp. 47–61 (1993)
4. Nasri, A.H.: Surface interpolation on irregular networks with normal conditions. Computer Aided Geometric Design 8, 89–96 (1991)
5. Zheng, J., Cai, Y.Y.: Interpolation over arbitrary topology meshes using a two-phase subdivision scheme. IEEE Trans. Visualization and Computer Graphics 12, 301–310 (2006)
6. Dyn, N., Levin, D., Gregory, J.A.: A Butterfly Subdivision Scheme for Surface Interpolation with Tension Control. ACM Trans. Graphics 9, 160–169 (1990)
7. Litke, N., Levin, A., Schröder, P.: Fitting Subdivision Surfaces. Proc. Visualization 319-324 (2001)
8. Loop, C.: Smooth Subdivision Surfaces Based on Triangles. Master'thesis. Dept. of Math., Univ. of Utah (1987)
9. Zorin, D., Schroder, P., Sweldens, W.: Interpolating Subdivision for Meshes with Arbitrary Topology. Computer Graphics, Ann. Conf. Series 30, 189–192 (1996)
10. de Boor, C.: How does Agee's method work? In: Proc. 1979 Army Numerical Analysis and Computers Conference ARO Report, Army Research Office, vol. 79, pp. 299–302 (1979)
11. Lin, H., Bao, H., Wang, G.: Totally positive bases and progressive iteration approximation. Computer & Mathematics with Applications 50, 575–585 (2005)
12. Lin, H., Wang, G., Dong, C.: Constructing iterative non-uniform B-spline curve and surface to fit data points. Science in China (Series F) 47, 315–331 (2004)
13. Qi, D., Tian, Z., Zhang, Y., Zheng, J.B.: The method of numeric polish in curve fitting. Acta Mathematica Sinica, (in Chinese) 18, 173–184 (1975)
14. Delgado, J., Peña, J.M.: Progressive iterative approximation and bases with the fastest convergence rates. Computer Aided Geometric Design 24, 10–18 (2007)
15. Lai, S., Cheng, F.: Similarity based Interpolation using Catmull-Clark Subdivision Surfaces. The Visual Computer 22, 865–873 (2006)
16. Magnus, I.R., Neudecker, H.: Matrix Differential Calculus with Applications in Statistics and Econometrics. John Wiley & Sons, New York (1988)
17. Shilane, P., Min, P., Kazhdan, M., Funkhouser, T.: The Princeton Shape Benchmark. Shape Modeling Int'l (2004)
18. Kobbelt, L.: Interpolatory Subdivision on Open Quadrilateral Nets with Arbitrary Topology. Comput. Graph. Forum 5, 409–420 (1996)