

# Parametrization of General Catmull-Clark Subdivision Surfaces and its Applications

Shuhua Lai and Fuhua (Frank) Cheng

Graphics & Geometric Modeling Lab, Department of Computer Science

University of Kentucky, Lexington, Kentucky 40506-0046

**Abstract.** A new parametrization technique and its applications for general Catmull-Clark subdivision surfaces are presented. The new technique extends J. Stam's work by redefining all the eigen basis functions in the parametric representation for general Catmull-Clark subdivision surfaces and giving each of them an explicit form. Therefore, the new representation can be used not only for evaluation purpose, but for analysis purpose as well. The new approach is based on an  $\Omega$ -partition of the parameter space and a detoured subdivision path. This results in a block diagonal matrix with constant size diagonal blocks ( $7 \times 7$ ) for the corresponding subdivision process. Consequently, eigen decomposition of the matrix is always possible and is simpler and more efficient. Furthermore, since the number of eigen basis functions required in the new approach is only one half of the previous approach, the new parametrization is also more efficient for evaluation purpose. This is demonstrated by applications of the new techniques in texture mapping, special feature generation, surface trimming, boolean operations and adaptive rendering.

**CR Categories:** I.3.5 [Computer Graphics]: Computational Geometry and Object Modelling - curve, surface, solid and object representations;

**Keywords:** subdivision, Catmull-Clark surfaces, surface parametrization, surface evaluation, eigenanalysis, discrete Fourier transform

## 1 Introduction

Subdivision surfaces have become popular recently in graphical modelling and animation because of their capability in modeling/representing complex shape of arbitrary topology [7], their relatively high visual quality, and their stability and efficiency in numerical computation. Subdivision surfaces can model/represent complex shape of arbitrary topology because there is no limit on the shape and topology of the control mesh of a subdivision surface. With the parametrization technique for subdivision surfaces becoming available [15] and with the fact that non-uniform B-spline and NURBS surfaces are special cases of subdivision surfaces becoming known [13], we now know that subdivision surfaces cover both *parametric forms* and *discrete forms*. Parametric forms are good for design and representation, discrete forms are good for machining and tessellation (including FE mesh generation) [1]. Hence, we have a representation scheme that is good for all graphics and CAD/CAM applications.

Research work for subdivision surfaces has been done in several important areas, such as surface interpolation [9], surface evaluation [5, 14, 15, 16, 17], surface trimming [10], boolean operations [4], and mesh editing [18]. However, powerful evaluation and analysis techniques for subdivision surfaces have not been fully developed yet. Parametrization methods that have been developed so far are suitable for evaluation purpose only, not for analysis purpose, because these methods either do not have an explicit expression, or are too complicated for each part to be explicit. For instance, in [15], eigen functions are pre-computed numerically and stored in a file. So they can be used for evaluation purpose only. Note that exact evaluation at a point of a subdivision surfaces is possible only if there is an explicit parametrization of the surface. Hence, an explicit parametrization is not only critical for analysis purpose, but for evaluation and rendering purpose as well.

In this paper we will present an  $\Omega$ -partition based approach to solve several important problems of subdivision surfaces: (1) computation of new control vertices at a specified subdivision level, (2) explicit parametrization of an extra-ordinary patch, and (3) surface evaluation at arbitrary parameter space point with eigen functions computed on the fly. The new approach will be presented for general Catmull-Clark subdivision surfaces [6]. But it works for any subdivision surfaces whose subdivision schemes can be implemented as a matrix-vector multiplication.

The new approach is based on the observation that the subdivision process on the control vertices can be broken into subdivision processes on smaller, same frequency groups after a few linear transformations. Each such subdivision process on points of the same frequency is independent of the *valence* of the extra-ordinary vertex. The dimension of the corresponding subdivision matrix for each frequency group is  $7 \times 7$ . Therefore, the process of using a large

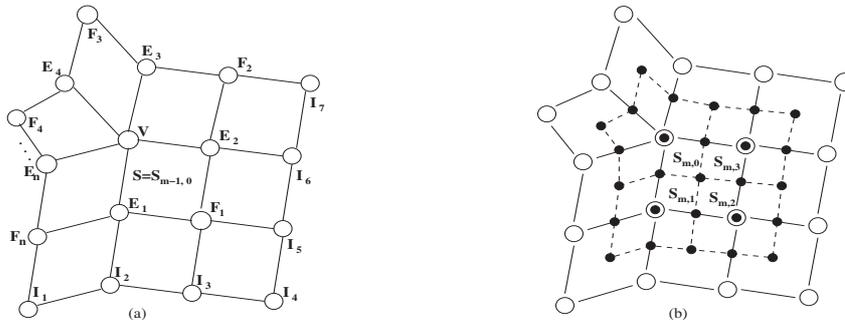


Figure 1: (a) Control vertices that influence an extra-ordinary patch. (b) New control vertices (solid dots) generated after a Catmull-Clark subdivision.

subdivision matrix to perform the subdivision process on the control vertices can be replaced with a set of  $7 \times 7$  matrices on the same frequency groups. This not only makes computation of the eigenstructures of the subdivision matrices always possible, but also simpler and more efficient. Inverses of the eigenvector matrices can also be explicitly computed. In addition to the parametric representation, we also derive a set of evaluation algorithms for applications in texture mapping, special feature generation, surface trimming, Boolean operations and adaptive rendering.

The remaining part of the paper is arranged as follows. Section 2 gives a brief review of the Catmull-Clark subdivision scheme and previous evaluation techniques. Section 3 shows an intuitive but expensive approach in parametrizing an extra-ordinary Catmull-Clark patch. Section 4 shows a more efficient approach in parametrizing a Catmull-Clark patch using an extended subdivision path. Section 5 shows how to compute the eigen structure of the subdivision matrix of the extended subdivision path. Section 6 shows the evaluation process of the new parametric representation at an arbitrary point of a Catmull-Clark patch. Section 7 gives some examples of analysis with our explicit representation around an extra-ordinary vertex. Section 8 shows application examples of the new scheme in texture mapping, special feature generation, surface trimming, boolean operations and adaptive rendering. The concluding remarks are given in Section 9.

## 1.1 Notations

The following notational conventions are adopted in this paper. Space objects such as points, lines and parametric functions are denoted by boldface upper case roman characters, e.g.,  $\mathbf{V}$ . Linearly transformed items or Fourier points are denoted by boldface lower case roman characters, e.g.,  $\mathbf{v}$ . All vectors are assumed to be columns. Vectors of ordinary items (resp. linearly transformed items or Fourier points) are denoted by upper (resp. lower) case italicized characters, e.g.,  $V$  (resp.  $g$ ). Matrices are denoted by uppercase roman characters, e.g.,  $M$ . The transpose of a vector  $V$  (resp. matrix  $M$ ) is denoted by  $V^T$  (resp.  $M^T$ ).

## 2 Background and Previous Work

### 2.1 Catmull-Clark Subdivision Surfaces

Given a control mesh, a *Catmull-Clark subdivision surface* (CCSS) is generated by iteratively refining the control mesh [6]. The limit surface is called a *subdivision surface* because the mesh refining process is a generalization of the uniform B-spline surface *subdivision technique*. 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. Vertex  $\mathbf{V}$  in Figure 1(a) is an extra-ordinary vertex of valence five. A mesh face with an extra-ordinary vertex is called an *extra-ordinary face*. The *valence* of an extra-ordinary face is the valence of its extra-ordinary vertex. In the following, for the sake of simplicity, a mesh face and the corresponding surface patch will be treated the same and denoted by the same notation.

Given an extra-ordinary face  $\mathbf{S} = \mathbf{S}_{0,0}$ . If the valence of its extra-ordinary vertex is  $n$ , then the surface patch corresponding to this extra-ordinary face is influenced by  $2n+8$  control vertices. The control vertices shown in Figure 1(a) are the ones that influence the patch marked with an “ $\mathbf{S} = \mathbf{S}_{m-1,0}$ ”. In general, if  $\mathbf{S}_{m-1,0}$  is the extra-ordinary subpatch generated after  $m-1$  subdivision steps, then by performing a Catmull-Clark subdivision step on the control vertices of  $\mathbf{S}_{m-1,0}$ , one gets  $2n+17$  new control vertices. See Figure 1(b) for the new control vertices generated for the patch  $\mathbf{S}_{m-1,0}$  shown in (a). These  $2n+17$  new control vertices define four subpatches:  $\mathbf{S}_{m,b}$ ,  $b = 0, 1, 2, 3$  (Figure 1(b)).  $\mathbf{S}_{m,0}$  is again an extra-ordinary patch but  $\mathbf{S}_{m,1}$ ,  $\mathbf{S}_{m,2}$ , and  $\mathbf{S}_{m,3}$  are regular uniform bicubic B-spline patches. Iteratively repeat this process, one gets a sequence of regular bicubic B-spline patches ( $\mathbf{S}_{m,b}$ ),  $m \geq 1$ ,  $b = 1, 2, 3$ , a sequence of extra-ordinary patches ( $\mathbf{S}_{m,0}$ ),  $m \geq 0$ , and a sequence of extra-ordinary vertices. The extra-ordinary

patches converge to the limit point of the extra-ordinary vertices [9]. The regular bicubic B-spline patches ( $\mathbf{S}_{m,b}$ ),  $m \geq 1$ ,  $b = 1, 2, 3$ , and the limit point of the extra-ordinary vertices form a partition of  $\mathbf{S}$ .

## 2.2 Previous Parametrization/Evaluation Methods

An algorithm for the evaluation of a subdivision surface at an arbitrary point was first proposed by J. Stam in 1998 for Catmull-Clark subdivision surfaces [15] and then in 1999 for Loop subdivision surfaces [16]. Stam's approach shows that an extra-ordinary surface patch and its derivatives can be represented as a linear combination of the control points with weights defined by a set of  $2n + 8$  eigenbasis functions where  $n$  is the valence of the extra-ordinary patch. The representation satisfies simple scaling relations and can be easily evaluated in constant time. However, even though analytical expressions for the eigenbasis functions have been derived, some of them are too complicated to be reported in the paper [15]. Besides, some of the eigenbasis functions are redundant. We will show in this paper that only  $n + 6$  eigenbasis functions are actually needed and, consequently, the evaluation process can be made more efficient. J. Stam's approach is mainly developed for evaluation purpose. As we shall present, our parametrization results can be used not only for evaluation, but for analysis purpose as well.

D. Zorin and D. Kristjansson extend the work of J. Stam by considering subdivision rules for piecewise smooth surfaces with parameter-controlled boundaries [17]. The main contribution of their work is the usage of a different set of basis vectors for the evaluation process which, unlike eigenvectors, depend continuously on the coefficients of the subdivision rules. The advantage of this algorithm is that it is possible to define evaluation for parametric families of rules without considering excessive number of special cases, while improving numerical stability of calculation.

In addition to Stam's approach, two different parametrizations of Catmull-Clark subdivision surfaces have been proposed by Boier-Martin and Zorin [5]. The motivation of their work is to provide parametrization techniques that are differentiable everywhere. Although all the natural parameterizations of subdivision surfaces are not  $C^1$  around extraordinary vertices of valence higher than four [5], the resulting surfaces are still  $C^2$  almost everywhere. Moreover, despite of the fact that the partial derivatives diverge around an extraordinary vertex, in this paper, we will show that an standardized normal vector can be calculated explicitly everywhere. As we know, precisely calculated normal vector is indispensable for surface shading purposes.

Exact evaluation of piecewise smooth Catmull-Clark surfaces near sharp and semi-sharp features is considered in [14]. Constant-time performance is achieved by employing Jordan decomposition of the subdivision matrix. In this paper we will show that special features can be generated using ordinary Catmull-Clark rules with constant-time evaluation performance as well.

## 3 Parametrization of an Extra-Ordinary Patch

The regular bicubic B-spline patches  $\{\mathbf{S}_{m,b}\}$ ,  $m \geq 1$ ,  $b = 1, 2, 3$ , induce a partition on the unit square  $[0, 1] \times [0, 1]$ . The partition is defined by :  $\{\Omega_{m,b}\}$ ,  $m \geq 1$ ,  $b = 1, 2, 3$ , with

$$\begin{aligned}\Omega_{m,1} &= \left[\frac{1}{2^m}, \frac{1}{2^{m-1}}\right] \times \left[0, \frac{1}{2^m}\right], \\ \Omega_{m,2} &= \left[\frac{1}{2^m}, \frac{1}{2^{m-1}}\right] \times \left[\frac{1}{2^m}, \frac{1}{2^{m-1}}\right], \\ \Omega_{m,3} &= \left[0, \frac{1}{2^m}\right] \times \left[\frac{1}{2^m}, \frac{1}{2^{m-1}}\right]\end{aligned}$$

(see Figure 2 for an illustration of the partition [15]). For any  $(u, v) \in [0, 1] \times [0, 1]$  but  $(u, v) \neq (0, 0)$ , there is an  $\Omega_{m,b}$  that contains  $(u, v)$ . To find the value of  $\mathbf{S}$  at  $(u, v)$ , first map  $\Omega_{m,b}$  to the unit square. If  $(u, v)$  is mapped to  $(\bar{u}, \bar{v})$  by this mapping, then compute the value of  $\mathbf{S}_{m,b}$  at  $(\bar{u}, \bar{v})$ . The value of  $\mathbf{S}$  at  $(0, 0)$  is the limit of the extra-ordinary vertices. For convenience of subsequent reference, the above partition will be called an  $\Omega$ -partition of the unit square.

In the above process,  $m$  and  $b$  can be computed as follows:

$$\begin{aligned}m(u, v) &= \min\{\lceil \log_{\frac{1}{2}} u \rceil, \lceil \log_{\frac{1}{2}} v \rceil\}, \\ b(u, v) &= \begin{cases} 1, & \text{if } 2^m u \geq 1 \text{ and } 2^m v < 1 \\ 2, & \text{if } 2^m u \geq 1 \text{ and } 2^m v \geq 1 \\ 3, & \text{if } 2^m u < 1 \text{ and } 2^m v \geq 1. \end{cases}\end{aligned}$$

The mapping from  $\Omega_{m,b}$  to the unit square is defined as:  $(u, v) \rightarrow (\bar{u}, \bar{v}) = (\phi(u), \phi(v))$ , where

$$\phi(t) = \begin{cases} 2^m t, & \text{if } 2^m t \leq 1 \\ 2^m t - 1, & \text{if } 2^m t > 1. \end{cases} \quad (1)$$

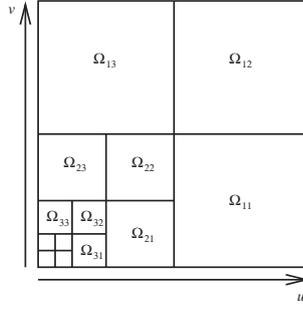


Figure 2:  $\Omega$ -partition of the unit square [15].

Since each  $\mathbf{S}_{m,b}$  is a standard B-spline surface, it can be expressed as  $\mathbf{S}(u, v) = W^T(\bar{u}, \bar{v})M\mathbf{G}_{m,b}$  where  $\mathbf{G}_{m,b}$  is the control point vector of  $\mathbf{S}_{m,b}$ ,  $W(u, v)$  is a vector containing the 16 power basis functions:

$$W^T(u, v) = [1, u, v, u^2, uv, v^2, u^3, u^2v, uv^2, v^3, u^3v, u^2v^2, uv^3, u^3v^2, u^2v^3, u^3v^3]$$

and  $M$  is the B-spline coefficient matrix. An important observation is,  $W^T(\bar{u}, \bar{v})$  can be expressed as the product of  $W^T(u, v)$  and two matrices:  $W^T(\bar{u}, \bar{v}) = W^T(u, v)K^mD_b$ , where  $K$  is a diagonal matrix

$$K = \text{Diag}(1, 2, 2, 4, 4, 4, 8, 8, 8, 8, 16, 16, 16, 32, 32, 64)$$

and  $D_b$  is an upper triangular matrix depending on  $b$  only.  $D_b$  can be obtained by replacing  $\bar{u}$ ,  $\bar{v}$  in  $W(\bar{u}, \bar{v})$  with  $\phi(u)$ ,  $\phi(v)$  defined in Eq. (1). Therefore, we have

$$\mathbf{S}(u, v) = W^T(u, v)K^mD_bM\mathbf{G}_{m,b}.$$

The computation of the control vertices of  $\mathbf{S}_{m,b}$  involves two matrices,  $A$  and  $\bar{A}$  [15].  $\bar{A}$  is a  $(2n + 17) \times (2n + 8)$  matrix, representing the subdivision process shown in Figure 1(b).  $A$  is a  $(2n + 8) \times (2n + 8)$  submatrix of  $\bar{A}$ , representing the process of mapping the  $2n + 8$  control vertices of the given extra-ordinary patch to the  $2n + 8$  control vertices of its extra-ordinary subpatch. Let

$$G = [\mathbf{V}, \mathbf{E}_1, \dots, \mathbf{E}_n, \mathbf{F}_1, \dots, \mathbf{F}_n, \mathbf{I}_1, \dots, \mathbf{I}_7]$$

then  $G$  (See Fig. 1(a) for its labelling) is the column vector representing the control vertices of  $\mathbf{S}$ . By applying  $A$  to  $G$  ( $m - 1$ ) times we get the  $2n + 8$  control vertices of the extra-ordinary subpatch  $\mathbf{S}_{m-1,0}$ . Now by applying  $\bar{A}$  to the control vertices of  $\mathbf{S}_{m-1,0}$  (represented as  $G_{m-1}$ ), we get  $2n + 17$  new control points which include the  $2n + 8$  control vertices of  $\mathbf{S}_{m,0}$ . Let  $\bar{G}_m$  be the column vector representation of these  $2n + 17$  vertices, we have  $\bar{G}_m = \bar{A}G_{m-1} = \bar{A}A^{m-1}G$ . Then by multiplying  $\bar{G}_m$  with an appropriate ‘‘picking’’ matrix  $P_b$ , we get the control vertices of the subpatch  $\mathbf{S}_{m,b}$ :  $G_{m,b} = P_b\bar{G}_m = P_b\bar{A}A^{m-1}G$ . Hence we have

$$\mathbf{S}(u, v) = W^T(u, v)K^mD_bMP_b\bar{A}A^{m-1}G. \quad (2)$$

This is a parametrization of an extra-ordinary patch. However, this is a costly process to use because it involves  $m - 1$  multiplications of the  $(2n + 8) \times (2n + 8)$  matrix  $A$ . In the next section, we will present an efficient approach to calculate  $G_{m,b}$  for any  $b$  and  $m$ .

## 4 Calculate Control Vertices after $m$ Subdivisions

The goal here is to show that instead of using the direct path from  $G$  to  $G_{m-1}$  to compute  $G_{m-1} = A^{m-1}G$  in the above equation, one should use the indirect, longer path ( $G \rightarrow g \rightarrow g_{m-1} \rightarrow G_{m-1}$ ) in Figure 3 to do the job. The reason for doing so is: the corresponding matrix  $T$  is a block diagonal matrix with each diagonal block of dimension  $7 \times 7$  only. Therefore, the process of computing their eigen decompositions is not only always possible, but also much simpler and more efficient. Details of this new approach and definitions of related mappings are given below. We consider a general CCSS here. That is, the new *vertex point*  $\mathbf{V}'$  after one subdivision is computed as follows:

$$\mathbf{V}' = \alpha_n \mathbf{V} + \beta_n \sum_{i=1}^n \mathbf{E}_i + \gamma_n \sum_{i=1}^n \mathbf{F}_i \quad (3)$$

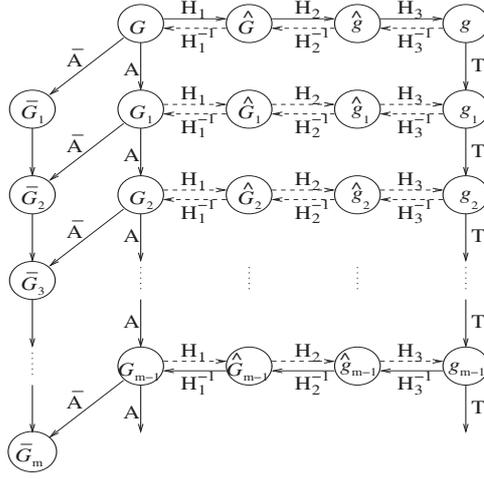


Figure 3: The extended subdivision diagram.

where  $\alpha_n, \beta_n$  and  $\gamma_n$  are positive numbers and their sum equals one. New *face points* and *edge points* are computed the same way as in [6].

First, to prepare  $G$  for the major transformation, we extend  $G$  into a vector of seven equal-length components, called  $\hat{G}$ :  $\hat{G} = (V^T, E^T, F^T, I_1^T, I_2^T, I_3^T, I_4^T)^T$ , where

$$\begin{aligned} V &= (\mathbf{V}, \mathbf{V}, \dots, \mathbf{V})^T, \\ E &= (\mathbf{E}_1, \mathbf{E}_2, \dots, \mathbf{E}_n)^T, \\ F &= (\mathbf{F}_1, \mathbf{F}_2, \dots, \mathbf{F}_n)^T, \\ I_k &= (\mathbf{I}_k, \mathbf{I}_{k+4}, \mathbf{0}, \dots, \mathbf{0})^T, \quad k = 1, 2, 3 \\ I_4 &= (\mathbf{I}_4, \mathbf{0}, \mathbf{0}, \dots, \mathbf{0})^T \end{aligned}$$

with all of them having the same length of  $n$ . We can get  $\hat{G}$  from  $G$  by a simple extension matrix  $H_1$ , i.e.,  $\hat{G} = H_1 G$ . Next, let  $\hat{g}$  be the result of applying an invertible linear transformation  $L$  to the components of  $\hat{G}$ :

$$\hat{g} = (LV^T, LE^T, LF^T, LI_1^T, LI_2^T, LI_3^T, LI_4^T)^T = (v^T, e^T, f^T, i_1^T, i_2^T, i_3^T, i_4^T)^T$$

Each component of  $\hat{g}$  has the same length  $n$ , but is indexed from 0 to  $n-1$ . We can get  $\hat{g}$  from  $\hat{G}$  by combining all  $L$ 's into a single matrix  $H_2$ , i.e.,  $\hat{g} = H_2 \hat{G}$ . It is easy to see that  $H_2$  is a block diagonal matrix. If we re-arrange the elements of  $\hat{g}$  into a set of same frequency groups:  $g = (h_0^T, h_1^T, \dots, h_{n-1}^T)^T$ , where  $h_\omega = (v_\omega, e_\omega, f_\omega, i_{1\omega}, i_{2\omega}, i_{3\omega}, i_{4\omega})^T$ , with  $0 \leq \omega \leq n-1$ . We can get  $g$  from  $\hat{g}$  through a  $7n \times 7n$  permutation matrix  $H_3$ , i.e.,  $g = H_3 \hat{g}$ . The above relationships hold for  $g_j, G_j, \hat{g}_j$  and  $\hat{G}_j, j \geq 1$ , as well (See Fig. 3). Since  $H_1, H_2$  and  $H_3$  are invertible, we can easily calculate  $g_j$  and  $G_j$  from each other.

For each  $j \geq 1$ , the subdivision process performed on  $G_{j-1}$  to get  $G_j$  can be reflected on  $g_{j-1}$  and  $g_j$  through  $H_1, H_2$  and  $H_3$ . The induced subdivision process [2] on  $g_{j-1}$  can be represented by a  $7n \times 7n$  matrix  $T$  as:  $g_j = T g_{j-1} = T^j g$ .  $T$  is a block diagonal matrix with each diagonal block  $T_\omega$  ( $\omega = 0, 1, 2, \dots, n-1$ ), being a  $7 \times 7$  matrix.  $T_\omega$  depends on the transformation  $L$ . However, the choice of  $L$  does not affect the result. In our derivation, for convenience, we choose  $L$  to be the Discrete Fourier Transform. The corresponding  $T_\omega$  can be found in [2]. Therefore, for each  $m \geq 1$ , we have (See Fig. 3):

$$A^{m-1} = H_1^{-1} H_2^{-1} H_3^{-1} T^{m-1} H_3 H_2 H_1 .$$

By combining the above expression with (2), we have

$$\mathbf{S}(u, v) = W^T K^m D_b M P_b \bar{A} H_1^{-1} H_2^{-1} H_3^{-1} T^{m-1} H_3 H_2 H_1 G \quad (4)$$

For a given  $(u, v)$ , every matrix in (4) is known to us if valance  $n$  is known. Hence it can be used to exactly and explicitly evaluate the position of  $\mathbf{S}(u, v)$ .

## 5 Eigenanalysis of $T$

Equation (4) provides a formal parametrization of an extra-ordinary patch. This parametrization, however, is still costly to evaluate because it involves  $m-1$  multiplications of the matrix  $T$ . The evaluation process can be considerably

simplified if  $T$  is decomposed as  $T = X^{-1}\Lambda X$ , where  $\Lambda$  is a diagonal matrix of eigenvalues of  $T$  and  $X$  is an invertible matrix whose columns are the corresponding eigenvectors. Therefore, the evaluation of  $T^{m-1}$  becomes the evaluation of  $X^{-1}\Lambda^{m-1}X$  only.

Note that  $T$  is a block diagonal matrix. To find the eigen decomposition of  $T$ , we first find the eigen decomposition of each diagonal block  $T_\omega$  of  $T$ :  $T_\omega = X_\omega^{-1}\Lambda_\omega X_\omega$ , ( $\omega = 0, 1, \dots, n-1$ ). Since each diagonal block  $T_\omega$  is of size  $7 \times 7$ , its eigen decomposition can be calculated explicitly.  $X$ ,  $\Lambda$  and  $X^{-1}$  are then formed as block diagonal matrices with diagonal blocks being  $X_\omega$ ,  $\Lambda_\omega$  and  $X_\omega^{-1}$ , respectively. Consequently,  $\mathbf{S}(u, v)$  can be expressed as:

$$\mathbf{S}(u, v) = W^T K^m Z_b \Lambda^{m-1} Z G \quad (5)$$

where  $Z = XH_3H_2H_1$  and  $Z_b = D_bMP_b\bar{A}Z^{-1}$ . For any given  $n$ , these matrices are known explicitly.

There are totally  $n + 6$  different eigenvalues in  $\Lambda$ . These different eigenvalues of  $T$  are:

$$\begin{aligned} \lambda_0 &= (4\alpha_n - 1 + \sqrt{16\alpha_n^2 - 8\alpha_n + 8\beta_n - 3})/8 \\ \lambda_1 &= (4\alpha_n - 1 - \sqrt{16\alpha_n^2 - 8\alpha_n + 8\beta_n - 3})/8 \\ \lambda_{2\omega} &= (c_\omega + 5 + \sqrt{c_\omega^2 + 10c_\omega + 9})/16 \\ \lambda_{2\omega+1} &= (c_\omega + 5 - \sqrt{c_\omega^2 + 10c_\omega + 9})/16 \\ \lambda_{n+1} &= 1 \\ \lambda_{n+2} &= 1/8 \\ \lambda_{n+3} &= 1/16 \\ \lambda_{n+4} &= 1/32 \\ \lambda_{n+5} &= 1/64 \end{aligned}$$

where  $1 \leq \omega \leq n/2$ ,  $c_\omega = \cos(2\pi\omega/n)$ , and  $\alpha_n$  and  $\beta_n$  are defined in (3). It is easy to check that  $\lambda_0 > \lambda_1$  and  $\lambda_2 > \lambda_i$  for  $3 \leq i \leq n$ .

## 6 Evaluation of a CCSS Patch

In this section we show how can Eq. (5) be used in the efficient evaluation of a CCSS patch at a given  $(u, v)$ . Eq. (5) can be used for both extra-ordinary and regular patches because the derivation of Eq. (5) did not use the assumption that  $n \neq 4$ .

First note that  $\mathbf{S}(u, v)$  defined in Eq. (5) can be written as a linear combination of these different eigenvalues in  $\Lambda$  to the  $(m-1)$ st power:  $\mathbf{S}(u, v) = W^T K^m \sum \lambda_j^{m-1} (Z_b \Theta_j Z) G$ , where  $\Theta_j$  is a  $7n \times 7n$  matrix with all the entries being zero except the ones corresponding to  $\lambda_j$  in matrix  $\Lambda$ . Those entries of  $\Theta_j$  are 1. Let  $M_{b,j} = Z_b \Theta_j Z$ . We get

$$\mathbf{S}(u, v) = W^T K^m \sum_{j=0}^{n+5} \lambda_j^{m-1} M_{b,j} G. \quad (6)$$

The exact expressions of  $M_{b,j}$  are shown in Appendix A. Eq. (6) is the most important result of this paper. This equation can be used to evaluate a CCSS patch at any point (including  $(0, 0)$ ), and it can also be used to compute the derivative of a CCSS patch at any point (including  $(0, 0)$  as well). The patch can be regular or extra-ordinary.

Note that for any  $m \geq 0$ , we have  $W^T(u, v)K^m = W^T(2^m u, 2^m v)$ . Define

$$\Phi_{b,j}(u, v) = W^T(2^m u, 2^m v) \lambda_j^{m-1} M_{b,j} \quad \text{and} \quad \Phi_b(u, v) = \sum_{j=0}^{n+5} \Phi_{b,j}(u, v).$$

$\Phi_{b,j}(u, v)$  are called the  $j$ th *eigen basis function* of CCSSs. There are totally  $n + 6$  eigen basis functions and for any given  $(u, v)$ , every eigen basis function can be exactly and explicitly represented. It is easy to check that all the eigen basis functions satisfy the so called *scaling relation* [15, 17]:

$$\Phi_{b,j}(u/2, v/2) = \lambda_j \Phi_{b,j}(u, v)$$

With the above definition, Eq. (6) can be represented as

$$\mathbf{S}(u, v) = \Phi_b(u, v) G,$$

which is used for fast rendering in our implementation.

One can compute the derivatives of  $\mathbf{S}(u, v)$  to any order simply by differentiating  $W(u, v)$  in Eq. (6) accordingly. For example,

$$\frac{\partial}{\partial u} \mathbf{S}(u, v) = \left( \frac{\partial W}{\partial u} \right)^T K^m \sum_{j=0}^{n+5} \lambda_j^{m-1} M_{b,j} G. \quad (7)$$

## 7 Analysis with an Explicit Representation

### 7.1 Limit Point of an Extra-Ordinary Vertex

Eq. (6) not only can be used for evaluation purpose, but analytic derivation as well. For example, one gets the limit point of an extra-ordinary vertex simply by setting  $u = v = 0$  and  $m \rightarrow \infty$  in (6):

$$\mathbf{S}(0,0) = [1, 0, \dots, 0] \cdot M_{b,n+1} \cdot G = \frac{5\mathbf{V} + (12\beta_n + 8\gamma_n)\bar{\mathbf{E}} + (2\beta_n + 8\gamma_n)\bar{\mathbf{F}}}{5 + 14\beta_n + 16\gamma_n}$$

where  $\bar{\mathbf{E}} = (\sum_{i=1}^n \mathbf{E}_i)/n$  and  $\bar{\mathbf{F}} = (\sum_{i=1}^n \mathbf{F}_i)/n$ . This result generalizes Eq. (13) of [9].

### 7.2 Partial Derivatives Around an Extra-Ordinary Vertex

It is known the first partial derivatives of  $\mathbf{S}(u, v)$  at  $(0, 0)$  diverge in a natural parametrization [5]. However, knowing the directions of them is sufficient in many applications. As pointed out by [2], when  $\lambda_0 \geq \lambda_2$ , a general Catmull-Clark subdivision surface is not  $C^1$  continuous. Suppose  $\lambda_0 < \lambda_2$ , dividing both sides of Eq. (7) by  $2^m \lambda_2^{m-1}$ , and by setting  $u = v = 0$  and  $m \rightarrow \infty$ , we get

$$\begin{aligned} D_u &= [0, 1, 0, 0, \dots, 0] \cdot M_{b,2} \cdot G \\ D_v &= [0, 0, 1, 0, \dots, 0] \cdot M_{b,2} \cdot G \end{aligned}$$

where  $D_u$  and  $D_v$  are the direction vectors of  $\frac{\partial \mathbf{S}(0,0)}{\partial u}$  and  $\frac{\partial \mathbf{S}(0,0)}{\partial v}$ , respectively. The normal vector at  $(0, 0)$  is the cross product of them. Similarly, when  $\lambda_0 < \lambda_2$ , it is easy to calculate the second partial derivatives at  $(0, 0)$ . These derivatives are listed as follows.

$$\begin{aligned} D_{uu} &= [0, 0, 0, 2, 0, \dots, 0] \cdot M_{b,2} \cdot G \\ D_{uv} &= [0, 0, 0, 0, 1, 0, \dots, 0] \cdot M_{b,2} \cdot G \\ D_{vv} &= [0, 0, 0, 0, 0, 2, 0, \dots, 0] \cdot M_{b,2} \cdot G \end{aligned}$$

where  $D_{uu}$ ,  $D_{uv}$  and  $D_{vv}$  are the direction vectors of  $\frac{\partial^2 \mathbf{S}(0,0)}{\partial u^2}$ ,  $\frac{\partial^2 \mathbf{S}(0,0)}{\partial u \partial v}$  and  $\frac{\partial^2 \mathbf{S}(0,0)}{\partial v^2}$ , respectively. Since  $M_{b,2}$  is explicitly and exactly known, all these vectors can be calculated once  $G$  is given.

### 7.3 Proof of $C^1$ -Continuity

With the explicit expressions of partial derivatives of  $\mathbf{S}(u, v)$  at  $(0, 0)$ , some properties of CCSS at an extra-ordinary point can be proved easily. For instance, one can prove that when  $\lambda_0 < \lambda_2$ , a CCSS is  $C^1$  continuous everywhere.

$C^1$ -Continuity of CCSS has been proven by many people with different approaches [2, 9, 11, 12]. Here a simple proof using our parametrization results is given below.

Expand  $D_u$  and  $D_v$ , we have

$$\begin{aligned} D_u &= \sum_{i=1}^n \bar{e}_i \cdot \mathbf{E}_i + \sum_{i=1}^n \bar{f}_i \cdot \mathbf{F}_i \\ D_v &= \sum_{i=1}^n \hat{e}_i \cdot \mathbf{E}_i + \sum_{i=1}^n \hat{f}_i \cdot \mathbf{F}_i \end{aligned}$$

where

$$\begin{aligned} \bar{e}_i &= \sum_{t=1}^5 x_{t1} c_{(i-t+2)}, & \hat{e}_i &= \sum_{t=1}^5 x_{t2} c_{(i-t+2)} \\ \bar{f}_i &= \sum_{t=1}^5 x_{t3} c_{(i-t+2)}, & \hat{f}_i &= \sum_{t=1}^5 x_{t4} c_{(i-t+2)} \end{aligned}$$

where  $c_\omega = \cos(2\pi\omega/n)$ . All scalars  $x_{ij}$ 's in the above definitions depend on valance  $n$  only and can be derived from  $M_{b,2}$  explicitly. To prove  $C^1$ -continuity at an extra-ordinary point, one needs to show that computation of the normal vector is independent of  $k$  (the ID of a face adjacent to an extra-ordinary point [2], which determines the order of the control points of a patch):

$$\left( \sum_{i=1}^n \bar{e}_i \mathbf{E}_{i+k} + \sum_{i=1}^n \bar{f}_i \mathbf{F}_{i+k} \right) \times \left( \sum_{i=1}^n \hat{e}_i \mathbf{E}_{i+k} + \sum_{i=1}^n \hat{f}_i \mathbf{F}_{i+k} \right).$$

To prove this, it is sufficient to show that  $\sum \bar{e}_i \mathbf{E}_{i+k} \times \sum \hat{e}_i \mathbf{E}_{i+k}$  is independent of  $k$ . The other parts can be proved similarly. Note

$$\sum_{i=1}^n \bar{e}_{i-k} \mathbf{E}_i \times \sum_{j=1}^n \hat{e}_{j-k} \mathbf{E}_j = \sum_{i \leq j} (\bar{e}_{i-k} \hat{e}_{j-k} - \bar{e}_{j-k} \hat{e}_{i-k}) \mathbf{E}_i \times \mathbf{E}_j$$

To prove the above expression is independent of  $k$ , we only need to prove  $(\bar{e}_{i-k} \hat{e}_{j-k} - \bar{e}_{j-k} \hat{e}_{i-k})$  is independent of  $k$ :

$$\begin{aligned} & \bar{e}_{i-k} \hat{e}_{j-k} - \bar{e}_{j-k} \hat{e}_{i-k} \\ &= \sum_{1 \leq s, t \leq 5} x_{s1} x_{t2} (c_{(i-k-s+2)} c_{(j-k-t+2)} - c_{(j-k-s+2)} c_{(i-k-t+2)}) \\ &= \sum_{1 \leq s, t \leq 5} x_{s1} x_{t2} (c_{(i-j-s+t)} - c_{(j-i-s+t)})/2 \end{aligned}$$

which is independent of  $k$ . Hence all the patches sharing a common extra-ordinary point have the same normal vector at the extra-ordinary point. Therefore, a CCSS is  $C^1$ -continuous at an extra-ordinary point.

When  $\lambda_0 \geq \lambda_2$ , it can be proved similarly that the resulting surface is not  $C^1$  continuous [2]. In fact, Eq. (6) and Eq. (7) can be used for many other analytic purposes as well. For example, the  $C^2$  discontinuity at an extra-ordinary point can be proved using these two formulas [3].

Although most of these properties of CCSS around an extra-ordinary vertex are well known, an explicit parametrization of CCSS nevertheless makes the analyzing process much more simpler and intuitive. Moreover, our results possibly can be used for studying other unknown properties of CCSS as well. For instance, we are investigating the integrability of a CCSS using the results in this paper.

## 8 Applications

### 8.1 Fast, Exact and Explicit Rendering

Eq. (6) not only gives us an explicit method to evaluate  $\mathbf{S}(u, v)$ , but also a faster and convenient way to render  $\mathbf{S}(u, v)$ . Note that  $M_{b,j}$  depend on the valence of the extra-ordinary vertex only. They can be explicitly and analytically computed for every different valence. For a given valence, we only need to perform such calculation once, no matter how many patches in the mesh are with such a valence. Once the step sizes for  $u$  and  $v$  are given, we can calculate all  $\Phi_b(u_i, v_k)$  beforehand and store them in a look-up table. Therefore, the evaluation of  $\mathbf{S}(u, v)$  at each point  $(u_i, v_k)$  basically is just a multiplication of  $\Phi_b(u_i, v_k)$  and  $G$  only. An algorithm of the fast rendering process is shown below. All the examples shown in this paper are rendered using this algorithm. One can see that it is essentially the same as the rendering process of a regular patch. An important difference between this approach and the previous approach [15] is that the previous approach was developed for special  $\alpha_n$  and  $\beta_n$  only. Therefore, it cannot handle general eigen basis functions while we can calculate all the eigen basis functions explicitly with only a small overhead. The horse shown in Fig. 5(b) is rendered using this algorithm with all the positions and normals exactly computed, not approximated. Hence, the quality of the image is better than those generated through the subdivision process.

CCSS-Rendering(Mesh,  $ustep$ ,  $vstep, \beta_n, \gamma_n$ )

1. For each valance  $n$  involved in input Mesh
2. For  $u = 0 : 1 : ustep$  and For  $v = 0 : 1 : vstep$
3. Calculate  $\Phi_b(u, v)$
4. For each patch whose valance is  $n$  in input Mesh
5. Find its  $2n + 8$  control points  $G$
6. For  $u = 0 : 1 : ustep$  and For  $v = 0 : 1 : vstep$
7. calculate each  $\mathbf{S}(u, v)$  and its normal

### 8.2 Generating Special Features

Eq. (6) can be used to render subdivision surfaces with special features. As we know, special features can be generated by properly arranging the control mesh. For instance, tripling a line in the control mesh generates a ridge or edge-like feature; tripling a control point generates a dart-like feature. One can get subdivision surfaces with complicated features and, consequently, complicated shape through this process. However, no matter how complicated the topology of the control mesh, as long as it is a two-manifold (to satisfy the definition of a CCSS), (6) will always generate the correct result. An example of a CCSS with sharp edges, corners and several genera is shown in Fig. 5(h). The control mesh of the surface is shown in Fig. 5(g). Since the features are generated from parametrization of the control mesh directly, the result is better than those generated by Boolean operations.

### 8.3 Texture Mapping

Precise texture mapping on a CCSS is possible only if a proper parametric representation is available for each extra-ordinary patch. However, to implement texture mapping on a CCSS, one needs to divide the interior faces of the control mesh into regions such that each region is of a rectangular structure first. Such a division will be called a *regular division*. Figure 4 shows a division of the interior faces of a CCSS into seven rectangular regions. Once a regular division of the interior faces of the control mesh is available, one simply performs texture mapping on each of these regions using standard approach. Examples of texture mapping on four subdivision surface represented objects: two rocker arms, a space station, a cow, and a leopard are shown in Fig. 5(c), 5(f), 5(l), 5(j) and 5(k), respectively. The regular division usually is not unique. Different divisions of the interior faces of the control mesh would lead to different texture outputs.

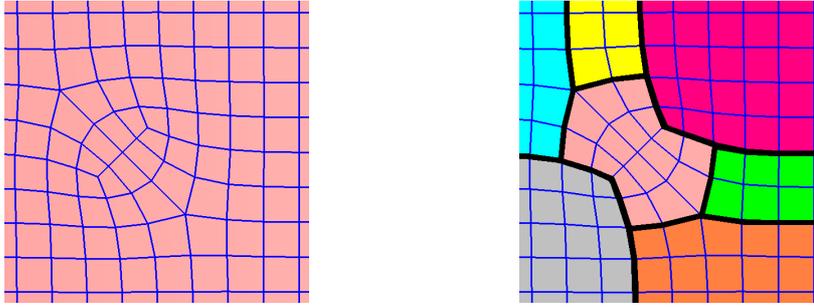


Figure 4: Regular division of the control mesh of a CCSS.

## 8.4 Surface Trimming

Surface trimming is another important application in computer graphics and CAD/CAM. The trimming loops are defined in the parameter space of the surface and iso-parametric lines in the parameter space are clipped against the trimming loops to have the trimmed regions removed. Hence, a global or local parametrization is necessary for precise and efficient rendering of a trimmed CCSS. In Fig. 5(a), a trimmed CCSS surface is shown. The trimmed regions are defined by the boundaries of the word 'ACMTOG05'. The CCSS surface has four extra-ordinary vertices in the trimmed region, but partitioning of the control mesh is not required here because the surface is rendered on the basis of individual patches.

## 8.5 Adaptive Rendering

Adaptive rendering is a technique for fast rendering of complicated objects. The rendering process of a patch depends on its flatness. A flat patch will not be tessellated as densely as other patches. Adaptive rendering is not a problem with (6) because (6) is capable of generating any point of the surface required in the tessellation process. One thing we must keep in mind is that, in order to avoid crack, we must generate the same number of points on the shared boundary of adjacent faces. But we can generate any number of points, even zero, inside a patch. An example of adaptive rendering is shown in Fig. 5(i) where a ventilation control component is represented by a single CCSS. The flatness of a patch is determined by the maximum norm of the second order forward differences of its control points.

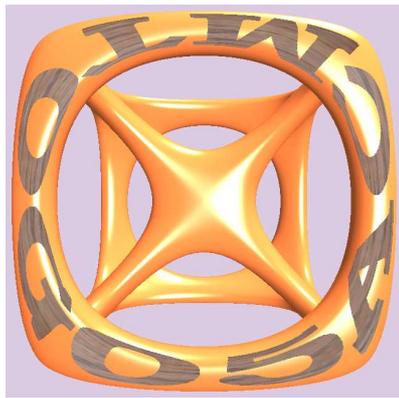
## 8.6 Boolean Operations

In solid modelling, an object is formed by performing Boolean operations on simpler objects or primitives. A CSG tree is used in recording the construction history of the object and is also used in the ray-casting process of the object. Surface-surface intersection (including the in-on-out test) and ray-surface intersection are the core operations in performing the Boolean operations and the ray-casting process. Each operation requires a parametrization of the surface to do the work. This is especially important for the in-on-out test. None of these is a problem with (6). Examples of performing Boolean operations on two and three cows are presented in Figure 5(d) and 5(e), respectively. A *difference* operation is first performed to remove some portions from each of these cows and a *union* operation is then performed to join them together. Performing Boolean operations on subdivision surfaces has been studied by Biermann, Kristjansson, and Zorin [4]. The emphasis of their work is different though - they focus on construction of the approximating multiresolution surface for the result, instead of precise computation of the surface-surface intersection curves.

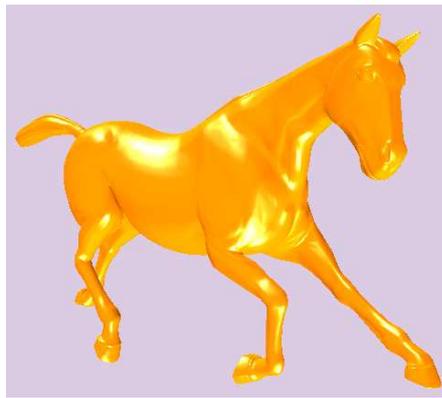
## 9 Summary

New parametrization and evaluation techniques for extra-ordinary patches of CCSSs are presented in this paper. The parametrization is obtained by performing subdivision on a group of same-frequency point sets after a few linear transformations, not on the control vertices themselves directly. This results in a block diagonal matrix with constant size diagonal blocks ( $7 \times 7$ ) for the corresponding subdivision process. Consequently, eigen decomposition of the subdivision matrix is always possible and is simpler and more efficient. Besides, the new approach works for the general CCSSs, not just a special case. The evaluation process using this parametrization works for both extra-ordinary and regular CCSS patches.

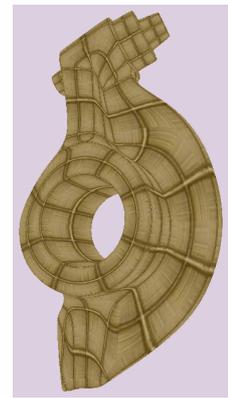
Two points have to be made here. First, the exponent  $m$  in (6) can not be cancelled out. This is because when  $\lambda_j$  is not a multiple of  $1/2$ ,  $m - 1$  in  $K^{m-1}$  and  $\lambda_j^{m-1}M_{b,j}$  does not cancel out. Hence, when  $n \neq 4$ , there does not exist a matrix  $M$  such that  $S(u, v) = W^T M G$ . Second, even though the computation is presented for Catmull-Clark subdivision surfaces, the new approach actually works for any subdivision surfaces whose subdivision schemes can



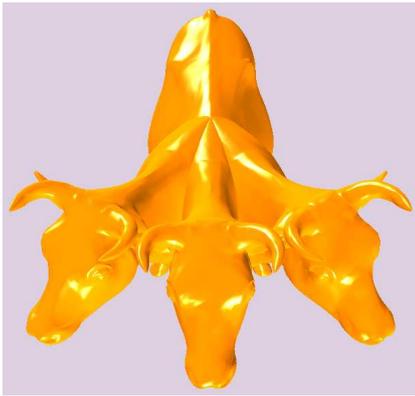
(a) Trimmed surface



(b) Exactly evaluated surface



(c) Textured surface



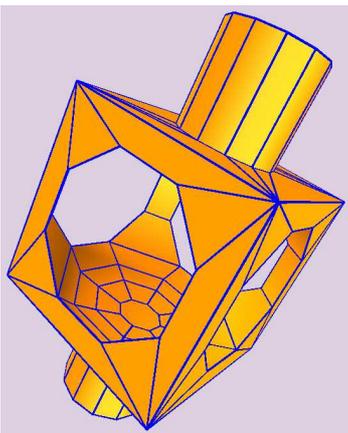
(d) Boolean operations



(e) Boolean operations



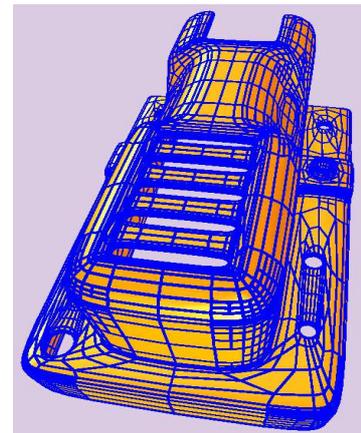
(f) Textured surface



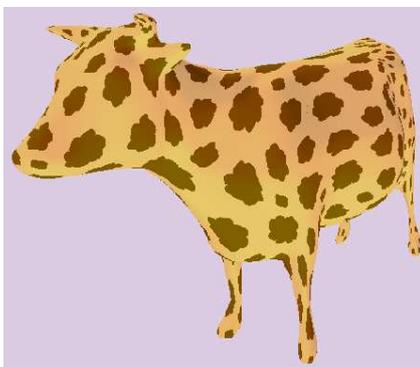
(g) Mesh with tripled edges



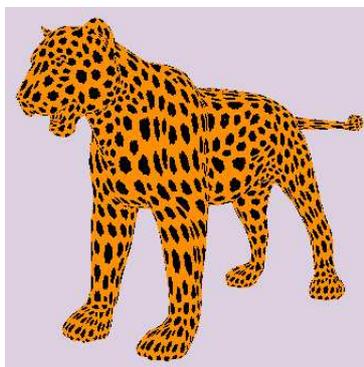
(h) Surface with special features



(i) Adaptive rendering



(j) Textured surface



(k) Textured surface



(l) Textured surface

Figure 5: Applications of parametric CCSS.

be implemented as a matrix-vector multiplication.

**Acknowledgement.** Data set for Fig. 5(d), 5(e) and 5(j), and data sets for Fig. 5(b) and 5(k) are downloaded from the following web sites

- <http://graphics.cs.uiuc.edu/~garland/research/quadrics.html>.
- <http://graphics.csail.mit.edu/~sumner/research/deftransfer/data.html>

respectively.

## A Eigen Basis Matrices $M_{b,j}$

We introduce two notations first:

$$M_{b,j}[k] \quad \text{and} \quad M_{b,j} = [Y_1, Y_2, \dots, Y_{2n+8}].$$

The first notation,  $M_{b,j}[k]$ , represents the  $k$ th row of the matrix  $M_{b,j}$ . The second notation means that  $M_{b,j}$  is a matrix of  $2n + 8$  columns and the  $i$ th column of  $M_{b,j}$  is  $Y_i$ . Recall that  $c_\omega = \cos(2\pi\omega/n)$ . Also, we define

$$f(k, \omega, x_1, x_2, x_3, x_4, x_5) = \sum_{i=1}^5 x_i c_{(k-i+2)\omega}.$$

The matrices  $M_{b,j}$ ,  $b = 1, 2, 3$ ,  $j = 0, 1, \dots, n + 5$ , are shown below in eight groups.

1.  $M_{b,0}$  and  $M_{b,1}$  (corresponding to eigenvalues  $\lambda_0$  and  $\lambda_1$ ): let

$$\begin{aligned} h &= (r-1)(8r-1)(32r-1)(64r-1)(r-s); \\ Y_b^T &= \frac{1}{h} Q_b \cdot [1, r, r^2, r^3, r^4, r^5, r^6]; \\ \hat{Y}_b &= -\frac{8s}{n} Y_b; \quad \bar{Y}_b = \frac{8s-1}{n} Y_b \end{aligned}$$

where  $r = \lambda_0$ ,  $s = \lambda_1$ , and  $Q_b$  (see Supplemental Material) is a constant matrix of dimension  $16 \times 7$ . Then

$$M_{b,0} = [Y_b, \hat{Y}_b, \dots, \hat{Y}_b, \bar{Y}_b, \dots, \bar{Y}_b, \mathbf{0}, \mathbf{0}, \mathbf{0}, \mathbf{0}, \mathbf{0}, \mathbf{0}, \mathbf{0}]$$

where  $\mathbf{0}$  is a zero vector of dimension 16.  $M_{b,1}$  is obtained by switching  $r$  and  $s$  in the above items.

2.  $M_{b,2\omega}$  and  $M_{b,2\omega+1}$  (corresponding to eigenvalues  $\lambda_{2\omega}$  and  $\lambda_{2\omega+1}$ ),  $1 \leq \omega < n/2$ : let

$$\begin{aligned} h &= (64r-1)(32r-1)(16r-1)(r-s)/2; \\ V_k^T &= [c_{(k+1)\omega}, c_{k\omega}, c_{(k-1)\omega}, c_{(k-2)\omega}, c_{(k-3)\omega}]; \\ B_k^T &= [V_k^T, rV_k^T, r^2V_k^T, r^3V_k^T, r^4V_k^T, r^5V_k^T]; \\ \hat{Y}_k &= \frac{1}{nh} Q_{b1} B_k; \quad \bar{Y}_k = \frac{4s-1}{nh(1+c_\omega)} Q_{b2} B_k \end{aligned}$$

where  $1 \leq k \leq n$ ,  $r = \lambda_{2\omega}$ ,  $s = \lambda_{2\omega+1}$  and  $Q_{b1}$  and  $Q_{b2}$  (see Supplemental Material) are constant matrices of dimension  $16 \times 30$ . Then

$$M_{b,2\omega} = [\mathbf{0}, \hat{Y}_1, \dots, \hat{Y}_n, \bar{Y}_1, \dots, \bar{Y}_n, \mathbf{0}, \mathbf{0}, \mathbf{0}, \mathbf{0}, \mathbf{0}, \mathbf{0}, \mathbf{0}].$$

$M_{b,2\omega+1}$  is obtained by switching  $r$  and  $s$  in the above items. Note that if  $n$  is odd, the last two matrices in this group are  $M_{b,n-1}$  and  $M_{b,n}$ . Otherwise, the last two matrices are  $M_{b,n-2}$  and  $M_{b,n-1}$ .

3.  $M_{b,n}$  when  $n$  is even (corresponding to eigenvalue  $\lambda_n$ ): set

$$\begin{aligned} \hat{Y}^T &= [1, -1, 1, -1, \dots, (-1)^{n+1}]/(4n), \\ \bar{Y}^T &= [0, 0, 0, 0, \dots, 0] \end{aligned}$$

where  $\hat{Y}$  and  $\bar{Y}$  are of length  $n$ . Then for  $b = 1, 2$  and  $3$ , we have

$$\begin{aligned} M_{b,n}[4] &= [0, \hat{Y}^T, \bar{Y}^T, 0, 0, 0, 0, 0, 0]; \\ M_{b,n}[5] &= [0, \bar{Y}^T, \hat{Y}^T, 0, 0, 0, 0, 0, 0]; \\ M_{b,n}[6] &= [0, -\hat{Y}^T, \bar{Y}^T, 0, 0, 0, 0, 0, 0] \end{aligned}$$

and all the other rows are zero.

4.  $M_{b,n+1}$  (corresponding to eigenvalue=1): set

$$t = 5 + 14\beta_n + 16\gamma_n; \quad \hat{h}_k = 4(3\beta_n + 2\gamma_n)/(nt); \quad \bar{h}_k = 2(\beta_n + 4\gamma_n)/(nt),$$

$1 \leq k \leq n$ . Then for  $b = 1, 2$  and  $3$ , we have

$$M_{b,n+1}[1] = [5/t, \hat{h}_1, \dots, \hat{h}_n, \bar{h}_1, \dots, \bar{h}_n, 0, 0, 0, 0, 0, 0, 0]$$

and all the other rows are zero.

5.  $M_{b,n+2}$  (corresponding to eigenvalue=1/8): set

$$\hat{y}_k = \begin{cases} \frac{-11(n-1)}{144n}, & k = 1 \\ \frac{1}{144n}, & 1 < k \leq n \end{cases} \quad \bar{y}_k = \begin{cases} \frac{5(n-2)}{288n}, & k = 1 \text{ or } k = n \\ \frac{1}{144n}, & 1 < k < n \end{cases}$$

$$t = 48(8\gamma_n - 1); \quad \hat{h} = (11 - 24\beta_n + 8\gamma_n)/(3tn); \quad \bar{h} = (-5 + 24\beta_n - 32\gamma_n)/(3tn).$$

Then for  $b = 1, 2$  and  $3$  we have

$$M_{b,n+2}[7] = [-1/t, \hat{h} + \hat{y}_1, \hat{h} + \hat{y}_2, \dots, \hat{h} + \hat{y}_n, \bar{h} + \bar{y}_1, \bar{h} + \bar{y}_2, \dots, \bar{h} + \bar{y}_n, \frac{1}{288}, \frac{1}{72}, \frac{1}{288}, 0, 0, 0, 0];$$

$$M_{b,n+2}[10] = [-1/t, \hat{h} + \hat{y}_n, \hat{h} + \hat{y}_1, \dots, \hat{h} + \hat{y}_{n-1}, \bar{h} + \bar{y}_n, \bar{h} + \bar{y}_1, \dots, \bar{h} + \bar{y}_{n-1}, 0, 0, 0, 0, \frac{1}{288}, \frac{1}{72}, \frac{1}{288}]$$

and all the other rows are zero.

6.  $M_{b,n+3}$  (corresponding to eigenvalue=1/16): set

$$\hat{y}_k = \sum_{\omega=0}^n \frac{7 \cdot f(k, \omega, 0, 1, 0, -1, 0)}{64n(2c_\omega - 7)}; \quad \bar{y}_k = \sum_{\omega=0}^n \frac{f(k, \omega, -5, -23, 23, 5, 0)}{192n(2c_\omega - 7)},$$

where  $1 \leq k \leq n$ . Then for  $b = 1, 2$  and  $3$  we have

$$M_{b,n+3}[11] = [0, \hat{y}_1, \hat{y}_2, \dots, \hat{y}_n, \bar{y}_1, \bar{y}_2, \dots, \bar{y}_n, -\frac{1}{192}, 0, \frac{1}{192}, 0, 0, 0, 0];$$

$$M_{b,n+3}[13] = [0, -\hat{y}_n, -\hat{y}_1, \dots, -\hat{y}_{n-1}, -\bar{y}_n, -\bar{y}_1, \dots, -\bar{y}_{n-1}, 0, 0, 0, 0, \frac{1}{192}, 0, -\frac{1}{192}]$$

and all the other rows are zero.

7.  $M_{b,n+4}$  (corresponding to eigenvalue=1/32): set

$$\hat{y}_k = \sum_{\omega=0}^n \frac{-f(k, \omega, 0, 41, 180, 41, 0)}{384n(4c_\omega - 45)}, \quad \bar{y}_k = \sum_{\omega=0}^n \frac{f(k, \omega, 10, 100, 100, 10, 0)}{384n(4c_\omega - 45)},$$

$$t = 32(-41 + 96\beta_n - 32\gamma_n); \quad \hat{h} = (-131 + 192\beta_n + 160\gamma_n)/(6tn); \quad \bar{h} = (55 - 96\beta_n - 80\gamma_n)/(3tn),$$

$1 \leq k \leq n$ . Then for  $b = 1, 2$  and  $3$ , we have

$$M_{b,n+4}[14] = [7/(2t), \hat{h} + \hat{y}_1, \hat{h} + \hat{y}_2, \dots, \hat{h} + \hat{y}_n, \bar{h} + \bar{y}_1, \bar{h} + \bar{y}_2, \dots, \bar{h} + \bar{y}_n, \frac{1}{384}, -\frac{1}{192}, \frac{1}{384}, 0, 0, 0, 0];$$

$$M_{b,n+4}[15] = [7/(2t), \hat{h} + \hat{y}_n, \hat{h} + \hat{y}_1, \dots, \hat{h} + \hat{y}_{n-1}, \bar{h} + \bar{y}_n, \bar{h} + \bar{y}_1, \dots, \bar{h} + \bar{y}_{n-1}, 0, 0, 0, 0, \frac{1}{384}, -\frac{1}{192}, \frac{1}{384}]$$

and all the other rows are zero.

8.  $M_{b,n+5}$  (corresponding to eigenvalue=1/64): set

$$\hat{y}_k = \sum_{\omega=0}^n \frac{f(k, \omega, 0, 27, 461, 461, 27)}{768n(8c_\omega - 217)}; \quad \bar{y}_k = \sum_{\omega=0}^n \frac{-f(k, \omega, 10, 229, 888, 229, 10)}{1152n(8c_\omega - 217)},$$

$$t = -209 + 448\beta_n - 64\gamma_n; \quad \hat{h} = (61 - 92\beta_n - 64\gamma_n)/(48tn); \quad \bar{h} = (-683 + 1216\beta_n + 752\gamma_n)/(576tn),$$

$1 \leq k \leq n$ . Then for  $b = 1, 2$  and  $3$ , we have

$$M_{b,n+5}[16] = [-45/(256t), \hat{h} + \hat{y}_1, \dots, \hat{h} + \hat{y}_n, \bar{h} + \bar{y}_1, \dots, \bar{h} + \bar{y}_n, \frac{-1}{2304}, \frac{1}{768}, \frac{-1}{768}, \frac{1}{2304}, \frac{-1}{768}, \frac{1}{768}, \frac{-1}{2304}]$$

and all the other rows are zero.

## References

- [1] Austin SP, Jerard RB, Drysdale RL, Comparison of discretization algorithms for NURBS surfaces with application to numerically controlled machining, *Computer Aided Design* 1997, 29(1): 71-83.
- [2] Ball AA, Storry DJT, Conditions for tangent plane continuity over recursively generated B-spline surfaces, *ACM Transactions on Graphics*, 1988, 7(2): 83-102.
- [3] Ball AA, Storry DJT, An investigation of curvature variations over recursively generated B-spline surfaces, *ACM Transactions on Graphics*, 1990, 9(4):424-437.
- [4] Biermann H, Kristjansson D, Zorin D, Approximate Boolean operations on free-form solids, *Proceedings of SIGGRAPH*, 2001: 185-194.
- [5] Boier-Martin I, Zorin D, Differentiable Parameterization of Catmull-Clark Subdivision Surfaces, *Eurographics Symposium on Geometry Processing* (2004).
- [6] Catmull E, Clark J. Recursively generated B-spline surfaces on arbitrary topological meshes, *Computer-Aided Design*, 1978, 10(6):350-355.
- [7] DeRose T, Kass M, Truong T, Subdivision Surfaces in Character Animation, *Proceedings of SIGGRAPH*, 1998: 85-94.
- [8] Doo D, Sabin M, Behavior of recursive division surfaces near extraordinary points, *Computer-Aided Design*, 1978, 10(6):356-360.
- [9] Halstead M, Kass M, DeRose T, Efficient, fair interpolation using Catmull-Clark surfaces, *Proceedings of SIGGRAPH*, 1993:35-44.
- [10] Litke N, Levin A, Schröder P, Trimming for Subdivision Surfaces, *Computer Aided Geometric Design* 2001, 18(5):463-481.
- [11] Reif U, A unified approach to subdivision algorithms near extraordinary vertices, *Computer Aided Geometric Design*, 1995, 12(2): 153-174.
- [12] Jörg Peters, Ulrich Reif, Analysis of Algorithms Generalizing B-Spline Subdivision, *SIAM Journal of Numerical Analysis*, 1998, Vol. 35, No. 2, pp. 728-748, April 1998.
- [13] Sederberg TW, Zheng J, Sewell D, Sabin M, Non-uniform recursive subdivision surfaces, *Proceedings of SIGGRAPH*, 1998:19-24.
- [14] Smith J, Epps D, Sequin C, Exact Evaluation of Piecewise Smooth Catmull-Clark Surfaces Using Jordan Blocks, <http://www.cs.berkeley.edu/~jordans/pubs/> June, 2004.
- [15] Stam J, Exact Evaluation of Catmull-Clark Subdivision Surfaces at Arbitrary Parameter Values, *Proceedings of SIGGRAPH* 1998:395-404.
- [16] Stam J, Evaluation of Loop Subdivision Surfaces, *SIGGRAPH'99 Course Notes*, 1999.
- [17] Zorin D, Kristjansson D, Evaluation of Piecewise Smooth Subdivision Surfaces, *The Visual Computer*, 2002, 18(5/6):299-315.
- [18] Zorin, D., Schröder, P., and Sweldens, W. Interactive Multiresolution Mesh Editing. In *Proceedings of SIGGRAPH 1997*, 259-268.