Evaluation of Loop Subdivision Surfaces

Jos Stam

Alias|wavefront, Inc.
1218 Third Ave, 8th Floor,
Seattle, WA 98101, U.S.A.
jstam@aw.sgi.com

Abstract

This paper describes a technique to evaluate Loop subdivision surfaces at arbitrary parameter values. The method is a straightforward extension of our evaluation work for Catmull-Clark surfaces. The same ideas are applied here, with the differences being in the details only.

1 Introduction

Triangular meshes arise in many applications, such as solid modelling and finite element simulations. The ability to define a smooth surface from a given triangular mesh is therefore an important problem. For topologically regular meshes a smooth triangular surface can be defined using box splines [1]. In 1987 Loop generalized the recurrence relations for box splines to irregular meshes [3]. Using his subdivision rules any triangular mesh can be refined. In the limit of an infinite number of subdivisions a smooth surface is obtained. Away from *extraordinary vertices* (whose valence $N \neq 6$) the surface can be parametrized using triangular Bezier patches derived from the box splines [2]. Until recently it was believed that no parametrizations that lend themselves to efficient evaluation existed at the extraordinary points. This paper disproves this belief. We define parametrizations near extraordinary points and show how to evaluate them efficiently. The techniques are identical to those used in our previous work on evaluating Catmull-Clark subdivision surfaces [5]. The differences are in the details only: different parameter domain, different subdivision rules and consequently a different eigenanalysis. We assume that the reader is familiar with the content of [5].

The remainder of this short paper is organized as follows. The next section briefly reviews triangular Loop subdivision surfaces. Section 3 summarizes how we define and evaluate a parametrization for such surfaces. Section 4 discusses implementation details while Section 5 depicts some results obtained using our scheme. Finally, some conclusions and possible extensions of this work are given in Section 6. Material which is of a rather technical nature is explained in the appendices.

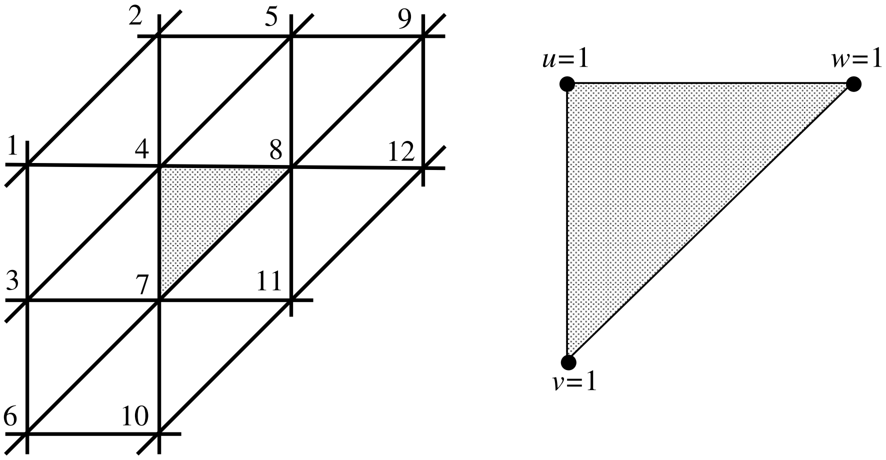


Figure 1: A single regular triangular patch defined by 12 control vertices.

2 Loop Subdivision Surfaces

Loop triangular splines generalize the box spline subdivision rules to meshes of arbitrary topology. On a regular part of the mesh each triangular patch can be defined by 12 control vertices as shown in Fig. 1. The basis functions corresponding to each of the control vertices are given in Appendix A. We obtained these basis functions by using a conversion from box splines to triangular Bezier patches developed by Lai [2]. This (regular) triangular patch can be denoted compactly as:

$$\mathbf{s}(v,w) = \mathbf{C}^T\,\mathbf{b}(v,w), \quad (v,w) \in \Omega,$$

where $\mathbf{C}$ is a 12×3 matrix containing the control vertices of the patch ordered as in Fig. 1 and $\mathbf{b}(v,w)$ is the vector of basis functions (see Appendix A). The surface is defined over the "unit triangle":

$$\Omega = \{\ (v,w) \mid v \in [0,1] \ \text{ and } \ w \in [0, 1-v]\ \}.$$

The parameter domain is a subset of the plane such that $v = 1$ corresponds to the point $(1,0)$ and $w = 1$ corresponds to the point $(0,1)$. We introduce the third parameter $u = 1 - v - w$ such that (u,v,w) forms a barycentric system of coordinates for the unit triangle. The value $u = 1$ corresponds to the origin $(0,0)$. The degree of the basis function is at most 4 in each parameter and our surface patch is therefore a quartic spline.

The situation around an extraordinary vertex of valence N is depicted in Fig. 2. The shaded triangle in this figure is defined by the $K = N + 6$ control vertices surrounding the patch. The extraordinary vertex corresponds to the parameter value $u = 1$. Since the valence of the extraordinary vertex in the middle of the figure is $N = 7$, there are $K = 13$ control vertices in this case. The figure also provides the labelling of the control vertices. We store the initial K control vertices in a $K \times 3$ matrix

$$\mathbf{C}_0^T = (\mathbf{c}_{0,1}, \cdots, \mathbf{c}_{0,K})\,.$$

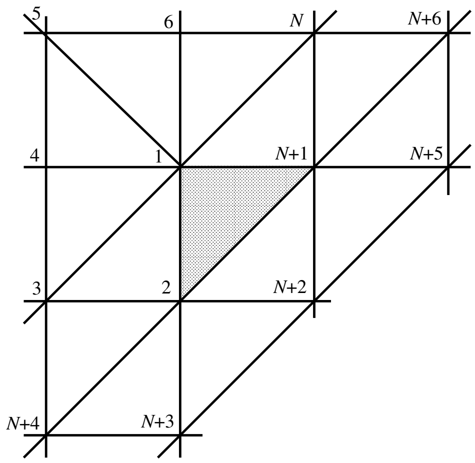


Figure 2: An irregular triangular patch defined by $K = N + 6 = 13$ control vertices. The vertex labelled "1" in the middle of the figure is extraordinary of valence 7.

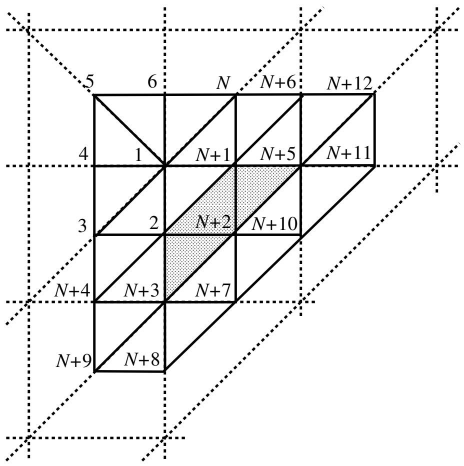


Figure 3: The mesh of Fig. 2 after one Loop subdivision step. Notice that three-quarters of the triangular patch can be evaluated.

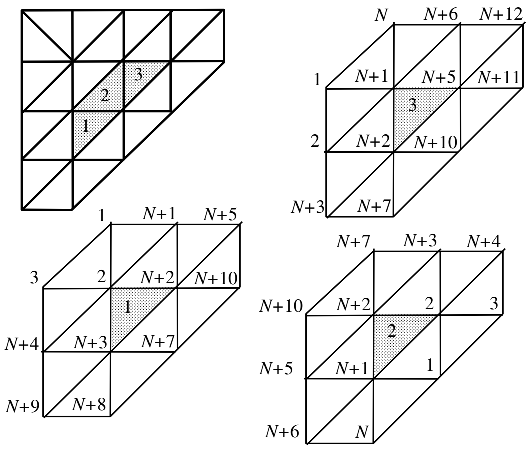


Figure 4: Three regular meshes corresponding to the three shaded patches. The labelling of the control vertices defines the picking matrices.

3 Method of Evaluation

3.1 Setup

Through subdivision we can generate a new set of $M = K + 6 = N + 12$ control vertices as shown in Fig. 3. Notice that we now have enough control vertices to evaluate three-quarters of the triangular patch. We denote the new set of control vertices by:

$$\begin{aligned} \mathbf{C}_1^T &= (\mathbf{c}_{1,1}, \cdots, \mathbf{c}_{1,K}) \quad \text{and} \\ \bar{\mathbf{C}}_1^T &= (\mathbf{c}_{1,1}, \cdots, \mathbf{c}_{1,K}, \mathbf{c}_{1,K+1}, \cdots, \mathbf{c}_{1,M}) . \end{aligned}$$

The subdivision step in terms of these matrices is entirely described by an $K \times K$ *extended subdivision matrix* $\mathbf{A}$:

$$\mathbf{C}_1 = \mathbf{A}\mathbf{C}_0,$$

where

$$\mathbf{A} = \begin{pmatrix} \mathbf{S} & \mathbf{0} \\ \mathbf{S}_{11} & \mathbf{S}_{12} \end{pmatrix}, \tag{1}$$

and the blocks are defined in Appendix B. The additional vertices needed to evaluate the surface are obtained from a bigger subdivision matrix $\bar{\mathbf{A}}$:

$$\bar{\mathbf{C}}_1 = \bar{\mathbf{A}}\mathbf{C}_0,$$

where

$$\bar{\mathbf{A}} = \begin{pmatrix} \mathbf{S} & \mathbf{0} \\ \mathbf{S}_{11} & \mathbf{S}_{12} \\ \mathbf{S}_{21} & \mathbf{S}_{22} \end{pmatrix},$$

and $\mathbf{S}_{21}$ and $\mathbf{S}_{22}$ are defined in Appendix B. Three subsets of 12 control vertices from $\bar{\mathbf{C}}_1$ define three regular triangular patches which can now be evaluated. If we repeat the subdivision step, we generate an infinite sequence of control vertices:

$$\bar{\mathbf{C}}_n = \bar{\mathbf{A}}\mathbf{C}_{n-1} = \bar{\mathbf{A}}\mathbf{A}^{n-1}\mathbf{C}_0, \quad n \geq 1.$$

For each $n \geq 1$ subsets of 12 vertices from $\bar{\mathbf{C}}_n$ form the control vertices of a regular triangular patch. Let us denote these three sets of control vertices by the following three 12×3 matrices $\mathbf{B}_{n,k}$, with $k = 1, 2, 3$. To compute these control vertices we introduce the $12 \times M$ "picking matrices" $\mathbf{P}_k$:

$$\mathbf{B}_{n,k} = \mathbf{P}_k\bar{\mathbf{C}}_n, \quad k = 1, 2, 3.$$

Each row of the picking matrix $\mathbf{P}_k$ is filled with zeros except for a one in the column corresponding to the index shown in Fig. 4. Each surface patch is then defined as follows:

$$\mathbf{s}_{n,k}(v, w) = \mathbf{B}_{n,k}^T\mathbf{b}(v, w) = \bar{\mathbf{C}}_n^T\mathbf{P}_k^T\mathbf{b}(v, w).$$

We seek a parametrization $\mathbf{s}(v, w)$ for our triangular surface for all $(v, w) \in \Omega$. As shown in Fig. 5 we can partition the parameter domain into an infinite set of tiles Ω_k^n, with $n \geq 1$ and $k = 1, 2, 3$. These subdomains are defined for $n \geq 1$ more precisely as:

$$\begin{aligned} \Omega_1^n &= \left\{ (v, w) \mid v \in \left[2^{-n}, 2^{-n+1}\right] \text{ and } w \in \left[0, 2^{-n+1} - v\right] \right\} \\ \Omega_2^n &= \left\{ (v, w) \mid v \in \left[0, 2^{-n}\right] \text{ and } w \in [0, v] \right\} \\ \Omega_3^n &= \left\{ (v, w) \mid v \in \left[0, 2^{-n}\right] \text{ and } w \in \left[2^{-n}, 2^{-n+1} - v\right] \right\} . \end{aligned}$$

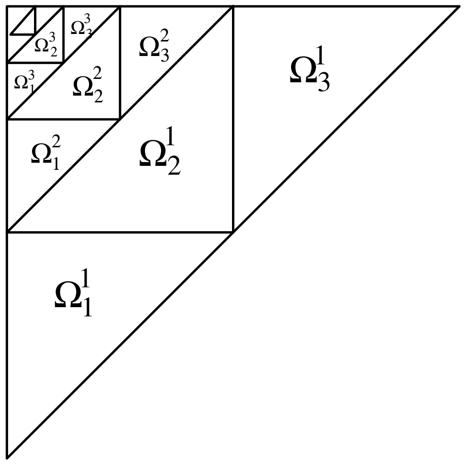


Figure 5: The parameter domain is partitioned into an infinite set of triangular tiles.

The surface patch is then defined by its restriction to each of these triangles:

$$\mathbf{s}(v,w)\Big|_{\Omega_k^n} = \mathbf{s}_{n,k}(\mathbf{t}_{n,k}(v,w)) = \mathbf{C}_0^T \left(\mathbf{P}_k \bar{\mathbf{A}} \mathbf{A}^{n-1}\right)^T \mathbf{b}(\mathbf{t}_{n,k}(v,w)), \tag{2}$$

where the transformation $\mathbf{t}_{n,k}$ maps the tile Ω_k^n onto the unit tile Ω (with the correct orientation of Fig. 1):

$$\begin{aligned} \mathbf{t}_{n,1}(v,w) &= (2^n v - 1, 2^n w), \\ \mathbf{t}_{n,2}(v,w) &= (1 - 2^n v, 1 - 2^n w) \quad \text{and} \\ \mathbf{t}_{n,3}(v,w) &= (2^n v, 2^n w - 1). \end{aligned}$$

Eq. 2 actually defines a parametrization for the surface patch. However, it is expensive to evaluate since it involves taking powers of a certain matrix to any number $n \geq 1$. To make the parametrization more efficient, we eigenanalyze.

3.2 Eigenstructure

When the valence $N > 3$, the extended subdivision matrix $\mathbf{A}$ is non-defective[1]. Consequently, $\mathbf{A}$ can be diagonalized:

$$\mathbf{A} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{-1}, \tag{3}$$

where $\mathbf{\Lambda}$ is the diagonal matrix which contains the eigenvalues and $\mathbf{V}$ contains the eigenvectors. These matrices have the following block structure:

$$\mathbf{\Lambda} = \begin{pmatrix} \mathbf{\Sigma} & \mathbf{0} \\ \mathbf{0} & \mathbf{\Delta} \end{pmatrix} \quad \text{and} \quad \mathbf{V} = \begin{pmatrix} \mathbf{U}_0 & \mathbf{0} \\ \mathbf{U}_1 & \mathbf{W}_1 \end{pmatrix}.$$

The diagonal blocks $\mathbf{\Sigma}$ and $\mathbf{\Delta}$ correspond to the eigenvalues of $\mathbf{S}$ and $\mathbf{S}_{12}$, respectively, and their corresponding eigenvectors are stored in $\mathbf{U}_0$ and $\mathbf{W}_1$, respectively. The matrix $\mathbf{U}_1$ is computed by extending the eigenvectors of $\mathbf{S}$, i.e., by solving the following linear systems:

$$\mathbf{U}_1\mathbf{\Sigma} - \mathbf{S}_{12}\mathbf{U}_1 = \mathbf{S}_{11}\mathbf{U}_0. \tag{4}$$

[1]The case $N = 3$ has a non-trivial Jordan block and is treated in Appendix C.

In Appendix B we compute the entire eigenstructure for Loop's scheme precisely. Let $\hat{\mathbf{C}}_0 = \mathbf{V}^{-1}\mathbf{C}_0$ be the projection of the initial control vertices onto the eigenspace of $\mathbf{A}$ and let $\Phi(v,w)$ be the K-dimensional vector of *eigenbasis* functions defined by:

$$\Phi(v,w)\Big|_{\Omega_k^n} = \mathbf{\Lambda}^{n-1}\left(\mathbf{P}_k\bar{\mathbf{A}}\mathbf{V}\right)^T \mathbf{b}(\mathbf{t}_{n,k}(v,w)) \quad n \geq 1 \ \text{ and } \ k = 1,2,3. \tag{5}$$

The eigenbasis functions for valences $N = 5$ and $N = 7$ are depicted in Fig. 6. Each function of the eigenbasis corresponds to one of the eigenvectors of the matrix $\mathbf{A}$. Each eigenbasis function is entirely defined by its restriction on the unit triangles Ω_1^1, Ω_2^1 and Ω_3^1. On each of these domains the eigenbasis is a quartic spline. The basis functions can be evaluated elsewhere since they satisfy the following scaling relation:

$$\Phi(v/2,w/2) = \mathbf{\Lambda}\Phi(v,w).$$

The triangular surface patch can now be written solely in terms of the eigenbasis:

$$\mathbf{s}(v,w) = \hat{\mathbf{C}}_0^T\Phi(v,w). \tag{6}$$

In the next section we show how to implement this equation.

4 Implementation

The eigenstructures of the subdivision matrices for a meaningful range of valences have to be computed once only. Let NMAX be the maximum valence, then each eigenstructure is stored internally in the following data structure:

```
typedef
 struct {
  double L[K];            /* eigenvalues */
  double iV[K][K];        /* inverse of the eigenvectors */
  double Phi[K][3][12];   /* Coefficients of the eigenbasis */
 } EIGENSTRUCT;
EIGENSTRUCT eigen[NMAX];,
```

where K=N+6. The coefficients of the eigenbasis functions are given in the basis of Appendix A. There are three sets of control vertices, one for each of the fundamental domains of the eigenbasis. These control vertices are simply equal to $\mathbf{P}_k\bar{\mathbf{A}}\mathbf{V}$. The eigenstructure was computed from the results of Appendix B and by solving the linear system defined by Eq. 4 numerically. Also, we numerically inverted the eigenvectors without encountering any numerical instabilities. We have included a data file called lpdata50.dat on the CDROM which contains these eigenstructures up to NMAX=50. Also included on the CDROM is a C program which reads in and prints out the data.

Using this eigenstructure the surface for any patch can be evaluated by first projecting the K control vertices defining the patch into the eigenspace of the subdivision matrix with the following routine.

```
ProjectPoints ( point *Cp, point *C, int N ) {
 for ( i=0 ; i<N+6 ; i++ ){
  Cp[i]=(0,0,0);
```

```
    for ( j=0 ; j<N+6 ; j++ ){
      Cp[i] += eigen[N].iV[i][j] * C[j];
    }
  }
}
```

This routine has to be called only once for a particular set of control vertices.

The evaluation at a parameter value `(v,w)` is performed by computing the product given in Eq. 6.

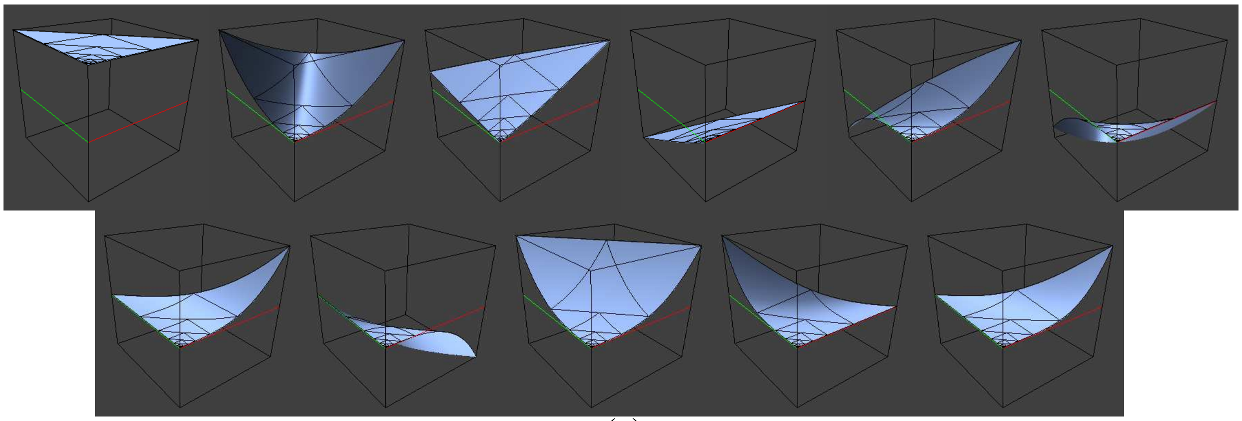
```
EvalSurf ( point P, double v, double w, point *Cp, int N ) {
  /* determine in which domain Ω_k^n the parameter lies */
  n = floor(1-log2(v+w));
  pow2 = pow(2,n-1);
  v *= pow2; w *= pow2;
  if ( v > 0.5 ) {
    k=0; v=2*v-1; w=2*w;
  }
  else if ( w > 0.5 ) {
    k=2; v=2*v; w=2*w-1;
  }
  else {
    k=1; v=1-2*v; w=1-2*w;
  }
  /* Now evaluate the surface */
  P = (0,0,0);
  for ( i=0 ; i<N+6 ; i++ ) {
    P += pow(eigen[N].L[i],n-1) *
           EvalBasis(eigen[N].Phi[i][k],v,w) * Cp[i];
  }
}
```

where the routine `EvalBasis` evaluates a regular triangular patch using the basis of Appendix A. To evaluate higher order derivatives, we replace `EvalBasis` with a function that evaluates a derivative of the basis. In this case, the end result also must be multiplied by two to the power `n*p`, where `p` is the order of differentiation. Therefore, the following line should be added at the end of `EvalSurf`:

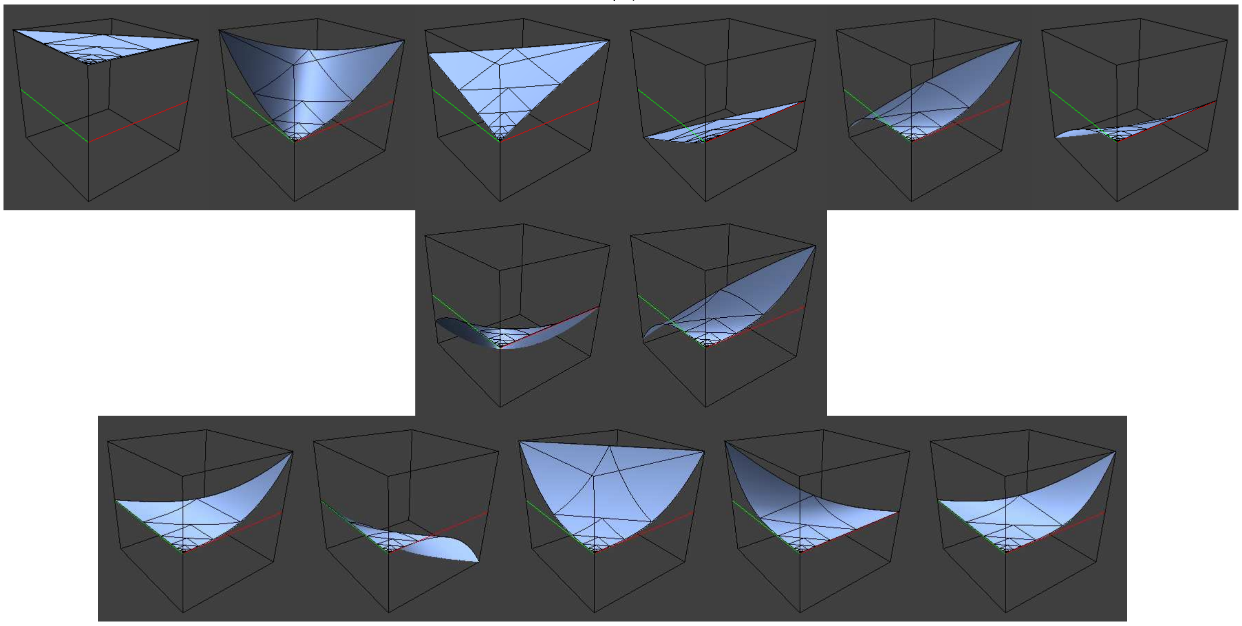
```
  P = k==1 ?  pow(-2,n*p)*P : pow(2,n*p)*P;
```

5 Results

We have implemented our evaluation technique and have used it to compute the eigenbases for different valences. Fig. 6 depicts the entire set of eigenbasis for valences 5 and 7. Notice that the last 6 eigenbasis functions are the same regardless of the valence, since they depend only on the eigenvectors of $\mathbf{S}_{21}$, which are the same for any valence. In fact, as for Catmull-Clark surfaces,

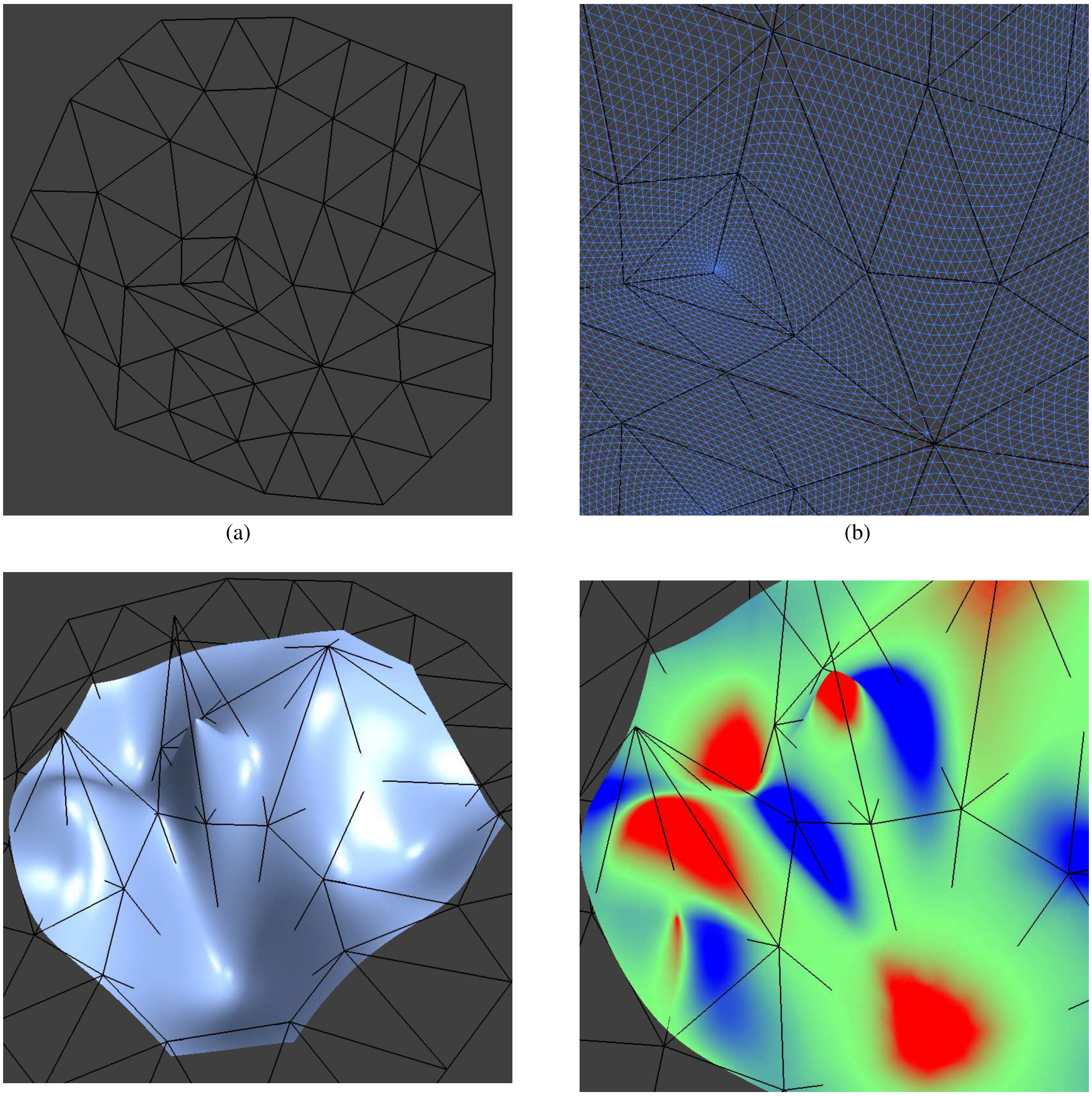


(a)



(b)

Figure 6: Complete set of eigenbasis function for a patch of valence (a) $N = 5$ and (b) $N = 7$.



(a) (b) (c) (d)

Figure 7: Results created using our evaluation scheme: (a) The base mesh which contains vertices with valences ranging from 3 to 9, (b) Isoparameter lines, (c) shaded surface and (d) Gaussian curvature plot.

these eigenbasis functions are equal to simple monomials (see [5]). The eigenbasis functions contain all the information necessary to analyze Loop subdivision surfaces.

To test our code we created the mesh shown in Figure 7.(a) which contains vertices of valence 3 to 9. Figure 7.(b) shows a closeup of the isoparameter lines generated by Loop subdivision and evaluated using our technique. In Figure 7.(c) we evaluated both the surface and the normal. Figure 7.(d) shows a Gaussian curvature plot, where red denotes positive curvature, green flat curvature and blue negative curvature.

6 Conclusions and Future Work

In this paper we have shown that our evaluation technique first developed for Catmull-Clark surfaces can be extended to the class of Loop subdivision surfaces. Our next step will be to present these results in a more general setting in which Catmull-Clark and Loop are regarded as special cases. The class of polynomial surfaces defined by Reif would be a good candidate [4].

Acknowledgments

Thanks to Michael Lounsbery for his assistance in computing the matrix $\mathbf{M}$ of Appendix A.

A Regular Triangular Spline Basis Functions

The triangular surface defined by the control vertices shown in Fig. 1 can be expressed in terms of 12 basis functions. Since Loop's scheme on the regular part of the mesh is a box spline, we can find the corresponding Bezier patch control vertices of the triangle. Lai has developed FORTRAN code which provides the conversion to the control vertices for the quartic triangular Bezier patches corresponding to the box spline [2]. We have used his code (with `L=2`, `M=2` and `N=2`) to get a 12×15 matrix $\mathbf{M}$ which converts from the Bezier control vertices of the patch to the 12 control vertices shown in Fig. 1. We get the 12 basis functions for our triangular patch by multiplying the 15 multivariate Bernstein polynomials by the matrix $\mathbf{M}$. Carrying out this multiplication leads to the following result (thanks to Maple's built in feature which converts to LaTeX):

$$
\begin{aligned}
\mathbf{b}^T(v,w) \;=\; & \frac{1}{12}\Big(\; u^4 + 2\,u^3 v,\;\; u^4 + 2\,u^3 w, \\
& u^4 + 2\,u^3 w + 6\,u^3 v + 6\,u^2 v w + 12\,u^2 v^2 + 6\,u v^2 w + 6\,u v^3 + 2\,v^3 w + v^4, \\
& 6\,u^4 + 24\,u^3 w + 24\,u^2 w^2 + 8\,u w^3 + w^4 + 24\,u^3 v + 60\,u^2 v w + 36\,u v w^2 + \\
& 6\,v w^3 + 24\,u^2 v^2 + 36\,u v^2 w + 12\,v^2 w^2 + 8\,u v^3 + 6\,v^3 w + v^4, \\
& u^4 + 6u^3 w + 12\,u^2 w^2 + 6\,u w^3 + w^4 + 2\,u^3 v + 6\,u^2 v w + 6\,u v w^2 + 2\,v w^3, \\
& 2\,u v^3 + v^4, u^4 + 6\,u^3 w + 12\,u^2 w^2 + 6\,u w^3 + w^4 + 8\,u^3 v + 36\,u^2 v w + \\
& 36\,u v w^2 + 8\,v w^3 + 24\,u^2 v^2 + 60\,u v^2 w + 24\,v^2 w^2 + 24\,u v^3 + 24\,v^3 w + 6\,v^4, \\
& u^4 + 8\,u^3 w + 24\,u^2 w^2 + 24\,u w^3 + 6\,w^4 + 6\,u^3 v + 36\,u^2 v w + 60\,u v w^2 + \\
& 24\,v w^3 + 12\,u^2 v^2 + 36\,u v^2 w + 24\,v^2 w^2 + 6\,u v^3 + 8\,v^3 w + v^4, \\
& 2\,u w^3 + w^4,\;\; 2\,v^3 w + v^4,
\end{aligned}
$$

$$2\,uw^3 + w^4 + 6\,uvw^2 + 6\,vw^3 + 6\,uv^2w + 12\,v^2w^2 + 2\,uv^3 + 6\,v^3w + v^4,$$
$$w^4 + 2\,vw^3\Big)\,,$$
where $u = 1 - v - w$.

B Eigenstructure of the Subdivision Matrix

The subdivision matrix $\mathbf{A}$ is composed of three blocks. The upper left block contains the "extraordinary rules" of Loop's scheme. It is equal to

$$\mathbf{S} = \begin{pmatrix} a_N & b_N & b_N & b_N & b_N & \cdots & b_N & b_N & b_N \\ c & c & d & 0 & 0 & \cdots & 0 & 0 & d \\ c & d & c & d & 0 & \cdots & 0 & 0 & 0 \\ & & \vdots & & & \ddots & & \vdots & \\ c & d & 0 & 0 & 0 & \cdots & 0 & d & c \end{pmatrix},$$

where

$$a_N = 1 - \alpha(N), \quad b_N = \alpha(N)/N, \quad c = 3/8 \ \text{ and } \ d = 1/8.$$

We have used the shorthand notation

$$\alpha(N) = \frac{5}{8} - \frac{(3 + 2\cos(2\pi/N))^2}{64}.$$

If we Fourier transform the matrix we get:

$$\hat{\mathbf{S}} = \begin{pmatrix} a_N & Nb_N & 0 & 0 & \cdots & 0 \\ c & c+2d & 0 & 0 & \cdots & 0 \\ 0 & 0 & f(1) & 0 & \cdots & 0 \\ 0 & 0 & 0 & f(2) & \cdots & 0 \\ & & \cdots & & \ddots & \cdots \\ 0 & 0 & 0 & 0 & \cdots & f(N-1) \end{pmatrix},$$

where

$$f(k) = \frac{3}{8} + \frac{2}{8}\cos(2\pi k/N)\,.$$

The eigenvalues and the eigenvectors of the transformed matrix are trivial to compute because of the almost-diagonal structure. They are

$$\mu_1 = 1,\ \mu_2 = \frac{5}{8} - \alpha(N),\ \mu_3 = f(1), \cdots, \mu_{N+1} = f(N-1).$$

Notice that we have $\mu_3^2 = \mu_2$. This is not surprising since Loop constructed his scheme from this relation [3]. The eigenvalues μ_3 to μ_{N-1} are of multiplicity two, since $f(k) = f(N-k)$, except of course for the case when N is even, then $\mu_{2+N/2}$ is only of multiplicity one. The corresponding eigenvectors are (when stored column wise):

$$\hat{\mathbf{U}}_0 = \begin{pmatrix} 1 & -\frac{8}{3}\alpha_N & 0 & 0 & \cdots & 0 \\ 1 & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 1 & \cdots & 0 \\ & & \vdots & & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 1 \end{pmatrix}.$$

By Fourier transforming these vectors back, we can compute the eigenvectors of the matrix $\mathbf{S}$. The result is

$$\mathbf{U}_0 = \begin{pmatrix} 1 & -\frac{8}{3}\alpha_N & 0 & 0 & \cdots & 0 \\ 1 & 1 & 1 & 1 & \cdots & 1 \\ 1 & 1 & E(1) & E(2) & \cdots & E(N-1) \\ 1 & 1 & E(2) & E(4) & \cdots & E(2(N-1)) \\ & & \vdots & & \ddots & \vdots \\ 1 & 1 & E(N-1) & E(2(N-1)) & \cdots & E((N-1)(N-1)) \end{pmatrix},$$

where $E(k) = \exp(2\pi ik/N)$. These are complex valued vectors. To get real-valued vectors we just combine the two columns of each eigenvalue to obtain two corresponding real eigenvalues. For example, the two real eigenvectors for the eigenvalue μ_{3+k}, $k = 0, \cdots, N-1$ are:

$$\begin{aligned} \mathbf{v}_k^T &= (0, C(0), C(k), C(2k), \cdots, C((N-1)k)) \quad \text{and} \\ \mathbf{w}_k^T &= (0, S(0), S(k), S(2k), \cdots, S((N-1)k)), \end{aligned}$$

where

$$C(k) = \cos(2\pi k/N) \quad \text{and} \quad S(k) = \sin(2\pi k/N).$$

The corresponding matrix of diagonal vectors is equal to

$$\mathbf{\Sigma} = \operatorname{diag}\left(1, \mu_2, \mu_3, \mu_3, \cdots, \mu_{(N-1)/2}, \mu_{(N-1)/2}\right),$$

when N is odd, and is equal to

$$\mathbf{\Sigma} = \operatorname{diag}\left(1, \mu_2, \mu_3, \mu_3, \cdots, \mu_{N/2-1}, \mu_{N/2-1}, \frac{1}{8}\right),$$

when N is even. This completes the eigenanalysis of the matrix $\mathbf{S}$. Let us now turn to the remainder of the matrix $\mathbf{A}$.

The remaining blocks of the matrix $\mathbf{A}$ are now given.

$$\mathbf{S}_{12} = \frac{1}{16}\begin{pmatrix} 2 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 2 \end{pmatrix} \quad \text{and} \quad \mathbf{S}_{11} = \frac{1}{16}\begin{pmatrix} 2 & 6 & 0 & 0 & \cdots & 0 & 0 & 6 \\ 1 & 10 & 1 & 0 & \cdots & 0 & 0 & 1 \\ 2 & 6 & 6 & 0 & \cdots & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & \cdots & 0 & 1 & 10 \\ 2 & 0 & 0 & 0 & \cdots & 0 & 6 & 6 \end{pmatrix}.$$

The matrix $\mathbf{S}_{12}$ has the following eigenvalues:

$$\nu_1 = \nu_2 = \nu_3 = \frac{1}{8}, \quad \text{and} \quad \nu_4 = \nu_5 = \frac{1}{16},$$

i.e.,

$$\mathbf{\Delta} = \operatorname{diag}\left(\frac{1}{8}, \frac{1}{8}, \frac{1}{8}, \frac{1}{16}, \frac{1}{16}\right).$$

And the corresponding eigenvectors are:

$$\mathbf{W}_1 = \begin{pmatrix} 0 & -1 & 1 & 0 & 0 \\ 1 & -1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix}.$$

We point out that the following problem might occur when trying to solve Eq. 4. When N is even, the column corresponding to the last eigenvector of $\mathbf{S}$ gives rise to a degenerate linear system, since the eigenvalue is $1/8$. Fortunately, the system can be solved manually, and in this case the last column of $\mathbf{U}_1$ is given by:

$$\mathbf{u}_{1,N+1}^T = (0, 8, 0, -8, 0)\,.$$

The remaining two blocks of the matrix $\bar{\mathbf{A}}$ are

$$\mathbf{S}_{21} = \frac{1}{8}\begin{pmatrix} 0 & 3 & 0 & 0 & \cdots & 0 & 0 & 1 \\ 0 & 3 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 3 & 1 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & \cdots & 0 & 0 & 3 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 3 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 1 & 3 \end{pmatrix} \text{ and } \mathbf{S}_{22} = \frac{1}{8}\begin{pmatrix} 3 & 1 & 0 & 0 & 0 \\ 1 & 3 & 1 & 0 & 0 \\ 0 & 1 & 3 & 0 & 0 \\ 3 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 3 & 1 \\ 0 & 0 & 0 & 1 & 3 \end{pmatrix}.$$

C Valence $N = 3$

When the valence of the extraordinary point is equal to three, the analysis of Section 3 breaks down since, in that case, the extended subdivision matrix has a non-trivial Jordan block. This means that the eigenvectors do not form a basis and the subdivision matrix cannot be diagonalized. Fortunately, this case can be dealt with quite easily since the matrices involved are only of size 9×9. Most of the computations reported in this appendix were computed using Maple's `jordan` command. In this case the Jordan decomposition of the subdivision matrix is

$$\mathbf{A} = \mathbf{V}\mathbf{J}\mathbf{V}^{-1},$$

where

$$\mathbf{J} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1/4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1/4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1/8 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1/8 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1/8 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1/16 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1/16 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1/16 \end{pmatrix},$$

$$
\mathbf{V} = \begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 33 \\
1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & -22 \\
1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & -22 \\
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & -22 \\
1 & 3 & 3 & 1 & -1 & 0 & 0 & 0 & 198 \\
1 & 0 & 4 & 1 & 0 & 0 & 0 & \frac{165}{16} & 473 \\
1 & -3 & 0 & 0 & 1 & 0 & 0 & 0 & 198 \\
1 & 4 & 0 & 0 & 0 & 1 & 1 & \frac{165}{16} & 438 \\
1 & 0 & -3 & -1 & 1 & 1 & 0 & 0 & 198
\end{pmatrix}
$$

and

$$
\mathbf{V}^{-1} = \begin{pmatrix}
2/5 & 1/5 & 1/5 & 1/5 & 0 & 0 & 0 & 0 & 0 \\
0 & -1/3 & -1/3 & 2/3 & 0 & 0 & 0 & 0 & 0 \\
0 & 2/3 & -1/3 & -1/3 & 0 & 0 & 0 & 0 & 0 \\
-8 & 0 & 3 & 3 & 1 & 0 & 1 & 0 & 0 \\
-4 & 0 & 0 & 3 & 0 & 0 & 1 & 0 & 0 \\
-8 & 3 & 3 & 0 & 1 & 0 & 0 & 0 & 1 \\
\frac{7}{11} & \frac{26}{33} & -\frac{7}{33} & -\frac{40}{33} & 0 & -1 & 1 & 1 & -1 \\
-\frac{16}{165} & 0 & \frac{16}{165} & \frac{16}{165} & -\frac{16}{165} & \frac{16}{165} & -\frac{16}{165} & 0 & 0 \\
\frac{1}{55} & -\frac{1}{165} & -\frac{1}{165} & -\frac{1}{165} & 0 & 0 & 0 & 0 & 0
\end{pmatrix}.
$$

Using these matrices the surface can now be evaluated as in the other cases. Only the evaluation routine has to be modified to account for the additional Jordan block. The modification relies on the fact that powers of the Jordan block have a simple analytical expression:

$$
\begin{pmatrix} 1/16 & 1 \\ 0 & 1/16 \end{pmatrix}^n = \begin{pmatrix} 1/16^n & n/16^{n-1} \\ 0 & 1/16^n \end{pmatrix}.
$$

With this in mind the last lines of routine `EvalSurf` should read:

```
/* Now evaluate the surface */
P = (0,0,0);
for ( i=0 ; i<N+6 ; i++ ) {
  P += pow(eigen[N].L[i],n-1) *
          EvalBasis(eigen[N].Phi[i][k],v,w) * Cp[i];
  if ( i==N+4 && N==3 )
    P += (n-1) * pow(eigen[N].L[i],n-2) *
          EvalBasis(eigen[N].Phi[i][k],v,w) * Cp[i+1];
}
}
```

References

[1] C. de Boor, K. Hollig, and S. D. Riemenschneider. *Box Splines*. Springer Verlag, Berlin, 1993.

[2] M. J. Lai. Fortran Subroutines For B-Nets Of Box Splines On Three- and Four-Directional Meshes. *Numerical Algorithms*, 2:33–38, 1992.

[3] C. T. Loop. *Smooth Subdivision Surfaces Based on Triangles*. M.S. Thesis, Department of Mathematics, University of Utah, August 1987.

[4] U. Reif. A Unified Approach To Subdivision Algorithms Near Extraordinary Vertices. *Computer Aided Geometric Design*, 12:153–174, 1995.

[5] J. Stam. Exact Evaluation of Catmull-Clark Subdivision Surfaces at Arbitrary Parameter Values. In *Computer Graphics Proceedings, Annual Conference Series, 1998*, pages 395–404, July 1998.