Edge and Vertex Insertion for a Class of C^1 Subdivision Surfaces

Ayman Habib and Joe Warren *
Department of Computer Science
Rice University

August 25, 1997

Abstract

Modeling with subdivision surfaces has been the subject of investigation by many researchers. To model most real-world objects, these subdivision schemes should have the ability to represent both sharp edges and vertices. In this paper we propose a new method for edge and vertex insertion on four–directional C^1 quadratic box spline surfaces. Edges and vertices may be inserted into an existing smooth surface and then deformed to have tangent plane discontinuities. The approach is easy to implement and can be generalized to work on other classes of subdivision schemes.

keywords: Subdivision surfaces, Box splines, Edge insertion, Vertex insertion.

1 Introduction

Building mathematical models for physical objects is one of the fundamental problems in computer aided geometric design. Manipulation of these modeled objects depends greatly on the properties of the associated mathematical models. Several mathematical representations have been proposed and used. One of the emerging techniques that has been receiving extensive attention in recent years is the use of subdivision surfaces (Catmull and Clark, 1978; Doo and Sabin, 1978; Loop, 1987; Dyn and Levin, 1990; Hoppe et al., 1994; Reif, 1995; Warren, 1995). Subdivision surfaces are obtained as the limit of a recursive refinement process applied to polyhedra. The power of this technique comes

*Supported in part by NSF grant CCR–9500572. The first author was also supported by NSF grant CCR-9113239.

from its ability to model complicated shapes with much simpler polyhedra using a few subdivision rules. Subdivision surfaces avoid the need for trimming surface patches, that arises in boundary representations. During subdivision, each curved face of an object is represented by a portion of a polyhedron. The topology of the polyhedron automatically ensures correct connectivity of the final surface. Subdivision also automatically produces a hierarchy of polyhedra, that approximates the final limit surface. Multiresolution techniques, such as wavelets, can be defined using this hierarchy to obtain different level–of–details representations of these surfaces (DeRose et al., 1993).

The quality of a subdivision surface depends on the properties of the subdivision scheme that builds it. Several subdivision schemes have been proposed in the literature (Catmull and Clark, 1978; Doo and Sabin, 1978; Loop, 1987; Dyn and Levin, 1990; Hoppe et al., 1994). Box splines provide a natural framework for building subdivision surfaces. A well established theory of box splines, over regular parametric grids, has been developed (de Boor et al., 1993). In this paper we adopt the four–directional, C^1 quadratic box spline, as an example subdivision scheme, though the treatment can be extended to deal with other subdivision schemes as well.

The above mentioned box spline produces a C^1 piecewise quadratic surface only over polyhedra with special connectivity. In this paper we extend the subdivision scheme associated with this box spline to build C^1 surfaces over polyhedra of arbitrary topological type. The resulting subdivision scheme is similar to that of Peters and Reif (Peters and Reif,) with the exception that different rules are used at extraordinary vertices to allow insertion of sharp vertices. We also give an algorithm that allows introduction of edges on smooth surfaces. This work differs from that of Nasri (Nasri, 1997) in that the spaces of surfaces obtained after edge insertion contains the space of possible surfaces prior to edge insertion. This property allows a smooth C^1–limit surface to be expressed as a collection of many patches that meet smoothly. This last property also distinguishes our work from the work of (Hoppe et al., 1994) where a smooth surface must be perturbed to allow for its representation as multiple smooth patches.

The first section of this paper introduces the notation and some background material on subdivision in general and the box spline we are going to use as a case study. In the second section we address subdivision on the interior of a control polyhedron (interior subdivision) and derive subdivision rules that produce C^1 surfaces over arbitrary closed polyhedra. Nonuniform subdivision, which includes edge and vertex insertion algorithms is then presented in Sections 3 and 4, along with boundary management and some examples.

1.1 Basic Concepts of Subdivision Schemes

To establish our notation, we start with a curve subdivision scheme. The canonical example of a curve subdivision scheme is Chaikin's algorithm (Chaikin, 1974) which computes quadratic B–splines, with uniform knots, recursively. One step of the algorithm is shown in Figure 1.

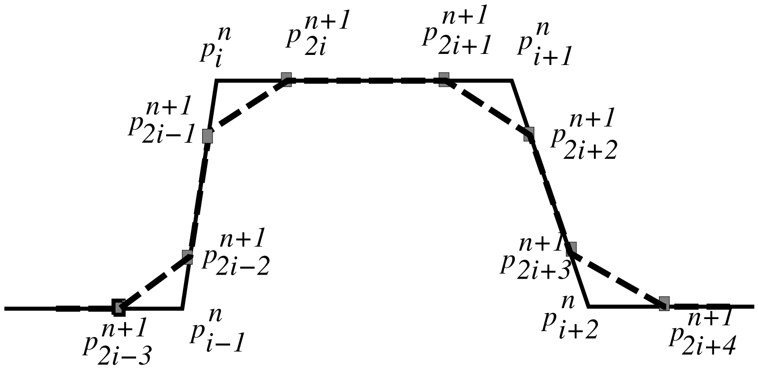

Figure 1: Chaikin's Algorithm. Solid (dashed) lines denote the control polygon before (after) subdivision.

Chaikin's algorithm starts with a control polygon. This polygon is then transformed into a new polygon with twice as many vertices by inserting new control points 1/4 and 3/4 the way between old vertices along each segment of the control polygon. The process is then repeated over and over.

Formally, let $\mathbf{p}^0 = \left[p_0^0, p_1^0, \ldots\right]$ denote the vector of initial control points (for now we shall ignore boundaries and assume infinite length vectors of control points). Also let $\mathbf{p}^i$ denote the control polygon obtained by applying the subdivision process i times to $\mathbf{p}^0$. The subdivision process successively replaces the vector $\mathbf{p}^n$ by $\mathbf{p}^{n+1}$. For Chaikin's algorithm the control points in $\mathbf{p}^{n+1}$ are computed from:

$$\left.\begin{array}{l} p_{2i}^{n+1} = \frac{3}{4}p_i^n + \frac{1}{4}p_{i+1}^n \\ p_{2i+1}^{n+1} = \frac{1}{4}p_i^n + \frac{3}{4}p_{i+1}^n \end{array}\right\} i \geq 0$$

Risenfield (Riesenfeld, 1975) proves that the curve obtained at the limit by repeating this subdivision process is the quadratic B–spline with control polygon $\mathbf{p}^0$ and uniform knots. Boldface $\mathbf{p}^i$ will generally denote the vector of control points obtained by subdividing $\mathbf{p}^0$, i times. Individual control points (the j^{th} point) in $\mathbf{p}^i$ will be denoted p_j^i. Figure 1 shows one round of subdivision using Chaikin's algorithm.

The subdivision rules for the four middle control points shown in Figure 1 can be written locally in matrix form as:

$$\begin{bmatrix} p_{2i-1}^{n+1} \\ p_{2i}^{n+1} \\ p_{2i+1}^{n+1} \\ p_{2i+2}^{n+1} \end{bmatrix} = \frac{1}{4} \begin{bmatrix} 1 & 3 & 0 & 0 \\ 0 & 3 & 1 & 0 \\ 0 & 1 & 3 & 0 \\ 0 & 0 & 3 & 1 \end{bmatrix} \cdot \begin{bmatrix} p_{i-1}^{n} \\ p_{i}^{n} \\ p_{i+1}^{n} \\ p_{i+2}^{n} \end{bmatrix} \tag{1}$$

The matrix in the right hand side of (1) is referred to as the local **subdivision matrix**. A subdivision scheme may have more than one local subdivision matrix depending on the geometry of the control polygon, in the neighborhood at which subdivision is applied, or on the number of rounds of subdivision.

Subdivision of the full control polygon can also be described by a recurrence of the form:

$$\mathbf{p}^{n+1} = S \cdot \mathbf{p}^{n} \tag{2}$$

The matrix S in (2) is a global subdivision matrix. The number of rows of S is double the number of columns (it has 2:1 slant) and it grows in size by 2 on each round of subdivision, thus it is eventually bi–infinite (for some surface schemes the number of control points quadruples and the size grows by four every round but nothing else changes).

When every control point affects only a small neighborhood around it, the associated basis functions have finite supports, and only a few basis functions affect the continuity of the curve at one point; thus all the relevant properties of the limit curve resulting from using a global subdivision matrix S are captured by only a finite portion of S (the local subdivision matrix shown in (1) for Chaikin's algorithm). From here on, unless otherwise specified, a subdivision matrix will be taken to mean the local subdivision matrix.

The values in the rows of S are referred to as the **subdivision mask**. The mask describes which control points in the vector $\mathbf{p}^i$ are used to compute the control point $\mathbf{p}_j^{i+1}$, and the corresponding weights. Usually only a few neighboring control points have non zero weights.

For surfaces, the subdivision process can still be represented by a recurrence of the form (2), but a more elaborate scheme for indexing control points is usually needed. Using the concepts and notation established in this section we introduce our case study box spline and its associated subdivision scheme in the next section.

1.2 The Case Study Box Spline

Box splines are compactly supported smooth piecewise polynomial functions. They are instances of the more general polyhedral splines. In two variables, a degree n box spline is the projection of an $n+2$–dimensional hyper–cube to the plane. The reader is referred to the excellent book by (de Boor et al., 1993) and the bibliography therein for a more elaborate presentation of box splines.

For our purposes, it suffices to recall the relevant properties of the box spline at hand. It has the set of four direction vectors,

$$\left[\begin{matrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & -1 \end{matrix} \right] \tag{3}$$

The induced grid and one basis function and its support in the parameter plane are shown in Figure 2. Each subdivision step refines the grid by halfing in each direction. The basis functions are piecewise quadratic with C^1 continuity across grid lines.

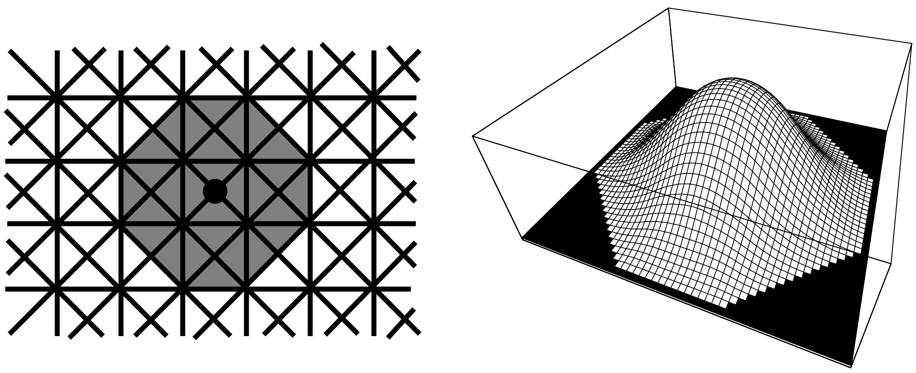

Figure 2: A four-directional grid, and a basis function. The dark region on the left indicates the support of the basis function centered at $\bullet$.

To specify the subdivision scheme for this box spline, we need to compute the associated subdivision mask. Generating functions can be used to compute the subdivision mask for arbitrary box splines e.g. (Warren, 1995). For this particular box spline, with direction vectors (3), the generating function for a basis function is

$$S(z) = \frac{1}{4}(1+z_1)(1+z_2)(1+z_1z_2)(1+z_1/z_2).$$

To compute the subdivision formula (the expression of one basis function on the coarse grid in terms of basis functions on the fine grid), expand $S(z)$ and write the coefficient of $z_1^i z_2^j$ at position (i,j) of the z_1, z_2-plane as shown in Figure 3. Thus the subdivision mask on the shaded face in Figure 3 is given by $\{1/4, 1/2, 1/4, 0\}$ and the associated subdivision matrix is:

$$S = \frac{1}{4} \begin{bmatrix} 2 & 1 & 0 & 1 \\ 1 & 2 & 1 & 0 \\ 0 & 1 & 2 & 1 \\ 1 & 0 & 1 & 2 \end{bmatrix}. \tag{4}$$

Figure 3 also shows how control points subdivide under this scheme. If the control points lie in $\mathbb{R}^d$, the subdivision rules (4) are applied in each coordinate independently.

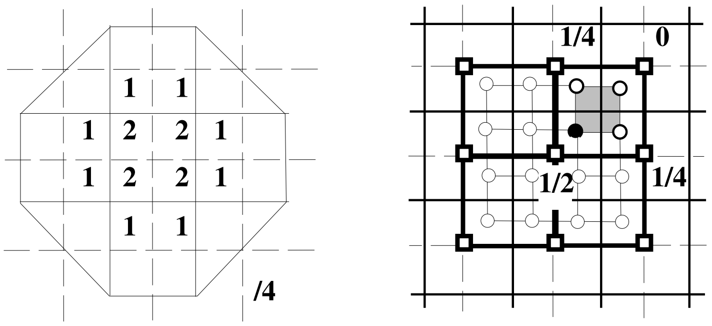

Figure 3: Subdivision scheme of the case–study box spline. The left figure shows the subdivision formula (the expression of a basis function on the coarse grid in terms of 12 basis functions on the fine grid). The right figure shows how control points subdivide; old control points are marked ($\square$) and new control points are marked $\circ$. The subdivision mask used to compute the control point $\bullet$ is shown.

To use the subdivision rules shown in Figure 3, we must be able to overlay each vertex of the control polyhedron at the center of one square in the parameter plane, with the connectivity shown (4–connectivity). This implies that every vertex of the polyhedron must have exactly four neighbors (*valence* 4) and that every face must be four–sided. These requirements restrict the set of polyhedra that the box spline can handle. In the next section we discuss these limitations and what more is needed to handle arbitrary polyhedra.

1.3 What is Needed to Handle Arbitrary Polyhedra?

The subdivision scheme associated with our box spline can only build surfaces of genus one without boundaries. To see this, we recall Euler's formula. For a 2-manifold, Euler's formula asserts that the number of vertices (v), edges (e), faces (f), and holes (h) are related to the genus (g) of the object by the formula:

$$v - e + f - h = 2(1 - g).$$

For a surface that can be tiled by a two dimensional grid where each vertex has valence four and all faces are four–sided, with no holes, we have $e = 2 * v$ and $v = f$ and thus g must be one.

For an arbitrary polyhedron, the regular box spline subdivision rules (4) can be used only on four–sided faces (Figure 3). For other faces, however, we need to develop special subdivision rules such that the limit surface has tangent plane continuity everywhere. We shall compute these special rules in the next section. We call this process *interior subdivision* since all vertices of the control polyhedron are assumed to be interior vertices. That is, there are no boundary vertices. This will enable us to model closed surfaces of arbitrary genus.

In reality, surfaces may also have boundaries. We call the process that builds surfaces with boundaries *nonuniform subdivision* because the topology of the control polyhedron close to a boundary is different from that in the interior and is not uniform anymore; thus special subdivision rules need to be applied there to get a smooth limit surface. In Section 3 we describe how to build these special subdivision rules.

The construction we give, in Section 3, handles boundaries such that boundary curves depend only on boundary control points. The benefit of this property is twofold: First, sharp edges can be introduced on a smooth surface by treating these edges as boundaries. Second, we obtain an alternative way to model surfaces of arbitrary genus, by breaking these surfaces into patches that join smoothly.

The main contribution of this paper is the edge and vertex insertion algorithms for the box spline surface and the ideas used to develop them. Some edge insertion algorithms are readily available in the literature, but only for special tensor product configurations (Catmull and Clark, 1978; Doo and Sabin, 1978). Hoppe et al. (Hoppe et al., 1994) insert edges and vertices on subdivision surfaces; however they address only subdivision over triangles. Furthermore the spaces of polyhedra resulting from their technique are not nested; thus their smooth surfaces cannot be reproduced as multiple patches that join smoothly, a desirable property for many applications that our scheme enjoys.

2 Interior Subdivision

In this section we address the problem of building closed subdivision surfaces, that are C^1–continuous everywhere, over arbitrary closed polyhedra. We call this process interior subdivision because all the vertices of the control polyhedron are interior vertices. The next subsection explains how arbitrary polyhedra subdivide under our box spline subdivision, and specifies what we need to do to handle arbitrary closed polyhedra. Section 2.2 gives the actual construction. Details of the proof of tangent plane continuity of the limit surfaces has been moved to the Appendix to increase the readability and continuity of the paper.

2.1 Primal vs. Dual Subdivision Schemes

A subdivision surface is obtained as the limit of a refinement process performed on a grid. We can distinguish two different types of subdivision schemes, depending on how the underlying grid is refined, primal and dual. In a primal scheme, every square of the grid is divided into four squares, usually by halfing in each direction to get the finer grid. The old grid is a subgrid of the new one. Dual schemes also divide every square in the coarse grid into four, but then the polyhedral dual is taken to be the new grid. One simple example is tensor–product B–spline subdivision. Odd degree B–splines have primal subdivision schemes, since the basis functions of an odd degree B–spline are centered at the knots, while even degree B–splines have dual schemes since the basis functions of an even degree B–spline are centered between knot values. The extension of the tensor–product biquadratic B–spline subdivision by (Doo and Sabin, 1978) is also a dual subdivision scheme. The scheme corresponding to the box spline at hand is a dual subdivision scheme as well.

Under a dual scheme, control points subdivide as shown in Figure 4. Inspecting Figure 4 we notice that one round of subdivision of a dual scheme creates a new face for each face, edge or vertex of the old control polyhedron. From here on, we shall call these new faces *face–faces*, *edge–faces* and *vertex–faces* respectively.

In Figure 4, face–faces resulting from old r–sided faces are r–sided. Vertex–faces resulting from old vertices of valence m are also m–sided, while edge–faces are always four sided.

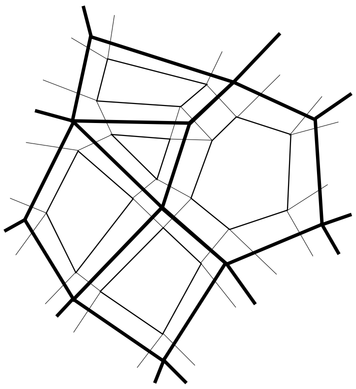

Figure 4: Dual subdivision for arbitrary valence polyhedra. Thick lines mark the edges of the original control polyhedron while thin lines mark the edges of the new polyhedron. After subdivision all control points have valence 4.

A key observation is that after one round of subdivision, although faces may have arbitrary number of sides, all new vertices are of valence four as shown in Figure 4. Thus, after the first round of subdivision, all new vertex–faces as well as edge–faces have exactly 4 sides. Only faces coming from old faces that are not four–sided produce non four–sided faces. Since we know how to handle four–sided faces (box spline subdivision formula (4), Section 1.2), we need only analyze non four–sided faces. We call these *extraordinary faces*. Subdivision rules for these extraordinary faces will be derived in the next Section.

2.2 Rules for Extraordinary Faces

According to the argument in the previous section, we can reformulate our subdivision process as a face subdivision process. The problem now becomes: for an n–sided polyhedral face, construct a set of subdivision rules that build a tangent plane continuous limit surface on this face. These new rules must also blend smoothly with subdivision rules applied to neighboring faces to ensure that no discontinuity is introduced between faces. In the following section we recall the conditions stated by (Reif, 1995) for tangent plane continuity and regularity of the limit surface of a subdivision process. In Section 2.2.2 the actual derivation of the subdivision rules for extraordinary faces is given. A detailed proof of tangent plane continuity of the limit surfaces built using the proposed subdivision rules is given in the Appendix.

2.2.1 Smoothness Conditions

It is well known that the tangent plane continuity of the limit surface of a subdivision scheme depends on the eigenstructure of the subdivision matrix, e.g. (Reif, 1995; Warren, 1995). We shall use the analysis given by (Reif, 1995) to study the smoothness of the limit surfaces of our subdivision scheme. First, we recall the set of conditions Reif gives for tangent plane continuity and regularity of the limit surfaces of a subdivision process:

Let M be an $r \times r$ subdivision matrix. Also let $\lambda_1, \lambda_2, \ldots, \lambda_r$ be the eigenvalues of M in descending moduli with corresponding eigenvectors $v_1, v_2, \ldots, v_r$ respectively. Then the following set of conditions are sufficient for M to produce tangent plane continuous and regular limit surfaces:

1. $1 = \lambda_1 > \lambda_2 = \lambda_3 > |\lambda_i| \quad i \geq 4$ where λ_2 has geometric and algebraic multiplicity 2.
2. The *characteristic map* of the eigenvectors v_2, v_3 of M and the basis functions is regular and injective (we shall explain this condition more, along with the proof of tangent plane continuity, and regularity in the Appendix.)

Henceforth, we refer to the above two conditions as the *C–conditions* for short. For the rest of this paper we use the notation S_n to denote the $n \times n$ matrix containing the subdivision coefficients (masks) for an n–sided face. Our objective is to compute the values of the entries of S_n for arbitrary n. Our strategy will be to use the first C–condition to derive subdivision rules with correct eigenstructure, then verify that the regularity condition is satisfied as well.

2.2.2 The Construction

We would like our subdivision rules to specialize to the regular box spline rules for four–sided faces. To this end we begin by studying the subdivision rules of four–sided faces. Faces with 4-sides subdivide according to the mask (4) as shown in Figure 3. We know that these rules applied to a polyhedron with only four-sided faces generate a limit surface that is piecewise quadratic, tangent plane continuous and regular everywhere. Studying the eigenstructure of S_4, the (4×4) matrix containing the subdivision rules (4), we notice:

1. The rows of S_4 are shifts of one another, that is S_4 is a *circulant* matrix.
2. The eigenvalues of S_4 in descending moduli are $1, 1/2, 1/2, 0$.
3. The corresponding matrix of eigenvectors,

$$V = \begin{bmatrix} 1 & 1 & 0 & -1 \\ 1 & 0 & 1 & 1 \\ 1 & -1 & 0 & -1 \\ 1 & 0 & -1 & 1 \end{bmatrix}.$$

For the general case of an n–sided face, the matrix S_n must also be circulant. This property follows from index invariance, which means that the same subdivision rules are used irrespective of vertex indexing. For a circulant matrix S_n with first row $a_0, a_1, \ldots, a_{n-1}$, a well known result from linear algebra (Davis, 1979) asserts that the eigenvalues of S_n are the values of the polynomial

$$\sum_{k=0}^{n-1} a_k x^k \qquad \text{evaluated at} \qquad x = \omega = e^{(\frac{2\pi j}{n})i} \quad j = 0, \ldots, n-1. \tag{5}$$

Now to compute the entries of S_n it suffices to specify its eigenvalues. These eigenvalues along with the (natural) assumption of index invariance specify the entries of S_n uniquely because if we pick the eigenvalues $\mu_0, \mu_1, \ldots, \mu_{n-1}$, and utilize (5), the problem of computing the entries of S_n reduces to solving the following system of linear equations in subdivision coefficients $a_i, i = 0, \ldots, n-1$:

$$\begin{bmatrix} 1 & 1 & 1 & \ldots & 1 \\ 1 & \omega & \omega^2 & \ldots & \omega^{n-1} \\ 1 & \omega^2 & \omega^4 & \ldots & \omega^{2(n-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{n-1} & \omega^{2(n-1)} & \ldots & \omega^{(n-1)^2} \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_{n-1} \end{bmatrix} = \begin{bmatrix} \mu_0 \\ \mu_1 \\ \mu_2 \\ \vdots \\ \mu_{n-1} \end{bmatrix}. \tag{6}$$

The matrix of coefficients on the left hand side of (6) is a well known matrix in linear algebra and Fourier analysis because it appears in Fast Fourier Transform (FFT) computation. From here on we shall refer to this matrix as an FFT–matrix.

By the first C–condition, the three leading eigenvalues must have the values $1, \lambda, \lambda$, where $0 < ||\lambda|| < 1$. For faces with more than four sides, there are many choices for the remaining eigenvalues, though their magnitudes must always be less than λ. In all cases we need to make sure that the second C–condition is satisfied as well.

One choice that guarantees that the mask reduces to the regular box spline mask when the number of sides is 4, is to pick the second and third largest eigenvalues to equal $1/2$ and the remaining eigenvalues to be zero. Since the eigenvalues in the first C–condition are ordered only by magnitude we must select three μ_i's to have the values $1, 1/2, 1/2$. We notice that the eigenvalues $1, 1/2, 1/2$ of S_4 correspond to $j = 0, 1, 3$ in (5). To generalize to S_n we pick

$$\mu_0 = 1 \quad \mu_1 = 1/2 \quad \mu_{n-1} = 1/2 \quad \text{and} \quad \mu_i = 0 \text{ for } i = 2, \ldots, n-2.$$

This selection generalizes the $n = 4$ case and also simplifies the computations in the general case.

Now utilizing the properties of the FFT matrix, the system of equations (6) solves to:

$$\begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_{n-1} \end{bmatrix} = \frac{1}{n} \begin{bmatrix} 1 & 1 & 1 & \ldots & 1 \\ 1 & \omega^{-1} & \omega^{-2} & \ldots & \omega^{-(n-1)} \\ 1 & \omega^{-2} & \omega^{-4} & \ldots & \omega^{-2(n-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{-(n-1)} & \omega^{-2(n-1)} & \ldots & \omega^{-(n-1)^2} \end{bmatrix} \begin{bmatrix} 1 \\ 1/2 \\ 0 \\ \vdots \\ 1/2 \end{bmatrix} = \frac{1}{n} \begin{bmatrix} 2 \\ 1 + \cos(\theta) \\ 1 + \cos(2\theta) \\ \vdots \\ 1 + \cos((n-1)\theta) \end{bmatrix} \tag{7}$$

where $\theta = \dfrac{2\pi}{n}$.

This solution enjoys the following properties:

1. The mask we obtain is symmetric since
$$\cos(r\theta) = \cos(r\frac{2\pi}{n}) = \cos((n-r)\frac{2\pi}{n}) = \cos((n-r)\theta).$$
2. All the coefficients, a_i, $i = 0, \ldots, n-1$, are positive since $|\cos(\theta)| \leq 1$.
3. By construction, the new rules specialize to the box spline rules for $n = 4$.
4. The computation of the eigenvectors and thus the regularity analysis is simpler than for other selections.

In the Appendix we prove the tangent plane continuity of the limit surface resulting from using the subdivision mask (7). The proof is done using Reif's analysis (Reif, 1995), by building the Characteristic Map of the subdivision scheme and proving that it is regular and injective.

2.3 Examples

Figure 5 shows two examples of C^1 surfaces built using the subdivision rules computed in Section 2.2. Figure 5(a) shows an object with genus zero obtained by subdividing an octahedron. Notice the small triangles at the center of each original triangular face of the octahedron. Figure 5(b) shows a double torus.

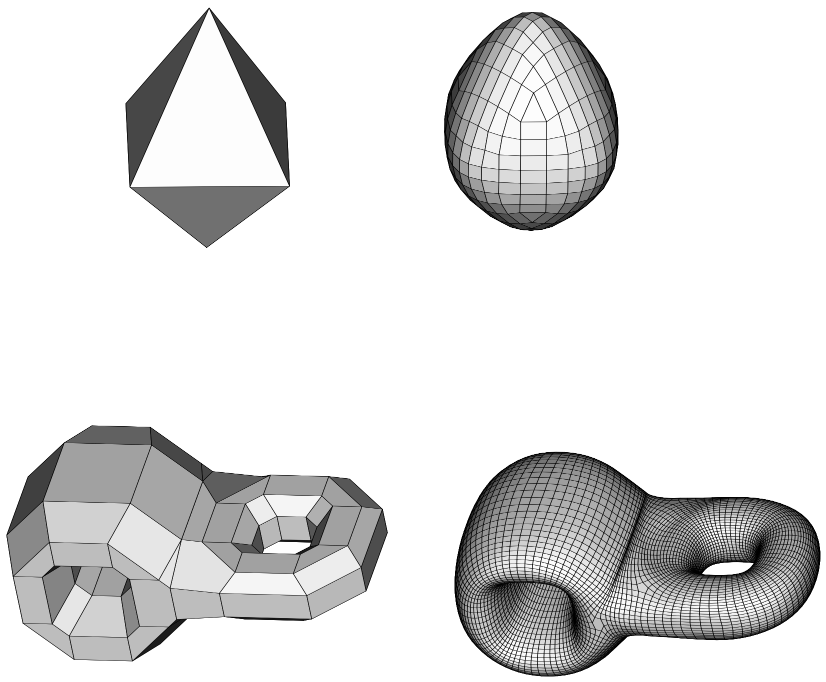

Figure 5: a. Subdividing an octahedron b. A double torus

3 Edge Insertion

Using the technique of the previous section we can build closed smooth surfaces that have tangent plane continuity. In reality, however, we would also like to model objects with boundaries, with creases or with sharp vertices. In Section 3.1 we discuss our overall strategy for edge insertion, and define precisely what we mean by edge insertion; then we give the actual construction in Section 3.2.

3.1 Strategy

To insert sharp edges and vertices on the limit surface, we use a machinery similar to (Hoppe et al., 1994), using a tagging scheme to mark vertices and edges of the initial control polyhedron that we want sharpened and then applying different subdivision rules depending on the tags. The essence of the solution is the computation of these different rules.

Hoppe *et. al.* also describe a scheme for building subdivision surfaces with creases and sharp vertices in a primal triangular subdivision setting. Our approach differs from theirs in that edge insertion is treated as a decomposition problem. Thus our objective is not only to create surfaces with sharp edges, but also smooth surfaces that can split into patches that meet smoothly. For manufacturing purposes, this is an important property because a smooth surface may need to be built from many patches that are manufactured separately, then assembled to create the final surface. This assembly cannot be done using the scheme of Hoppe et. al. since their subdivision rules on the sides of an edge do not reproduce the exact subdivision rules used on the interior.

Our technique for inserting sharp edges on the limit surface is a two step process. In the first step, we insert the edge as a smooth edge, breaking the surface into two patches that meet smoothly, and introducing new control points that entirely control the edge curve. In the second step we move these new control points to make the edge into a sharp edge if desired, or just break the surface across this edge into patches. Clearly we only need to show how to perform the first step that splits the surface since edge control points completely specify the edge curve.

3.2 Rules for Edge Insertion

Starting with a subdivision surface built using the interior subdivision rules of Section 2, we now show how to insert a smooth edge on this surface. Initially we shall not worry about what happens at the end points of the inserted edge (this problem will be dealt with along with vertex insertion in Section 4); thus we only study the case of inserting an edge that goes across a four–sided face.

We formulate the problem as knot–line insertion in the parameter space (actually doubling a knot–line). For our specific box spline, it can be shown that curves on the limit surface that are maps of iso–parametric grid lines are quadratic B–splines with uniform knots. It is known, from the theory of B–splines, that doubling a knot on a quadratic B–spline with uniform knots corresponds to inserting a new control point half–way between two old control points as shown in Figure 6. After doubling the knot, the curve and its control polygon can both be broken into two pieces across this new control point. The new control point may also be moved to introduce a kink on the curve.

Now we return to our box spline and study doubling the horizontal knot–line in the middle between the upper two and the lower two control points in

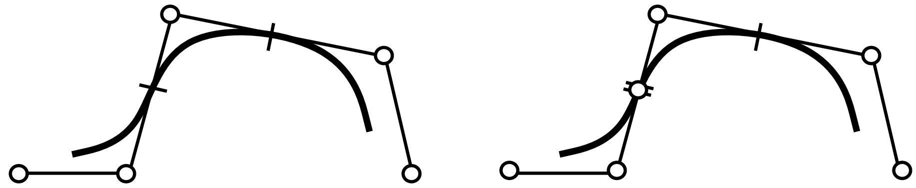

Figure 6: Doubling a knot on a quadratic B–splines with uniform knots. Marks on the curve denote knot values, and white circles denote control points.

Figure 7.a. Inspecting the support of a basis function of this box spline (shown in Figure 2), we notice that the only basis functions with support overlapping the knot–line being doubled are those corresponding to the four control points (1)–(4) in Figure 7.a. Since the images of the two vertical grid lines shown are quadratic B–splines with uniform knots, doubling the middle knot–line creates two new control points (5), (6) half–way between the upper two and the lower two control points as shown in Figure 7.b.

Now, if we can perform knot–line insertion after any number of subdivision steps, we must be able to reproduce the limit surface as two patches, since we might as well perform the edge insertion on the limit surface itself. In other words, the commutativity of subdivision and knot–line insertion guarantees that the final surface will be reproducible from two pieces. This commutativity is the main tool that we use to derive the subdivision rules along and next to edges. Figure 7 demonstrates this commutativity. Control points on the left (right) are before (after) subdivision. Those on the top (bottom) are before (after) knot–line doubling.

Formally, let K be the 6×4 knot–line insertion matrix that takes control points (1)–(4) of Figure 7.a to control points (1)–(6) in Figure 7.b. Then, using the numbering shown in figure for the control points we can write:

$$K = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1/2 & 0 & 0 & 1/2 \\ 0 & 1/2 & 1/2 & 0 \end{bmatrix}. \tag{8}$$

The same matrix is used to insert the knot–line after subdivision, and the regular box spline subdivision matrix S given in (4) takes control points (1)–(4) in Figure 7.a to the four new control points (1)'–(4)' in Figure 7.c.

What we are looking for is a 6×6 matrix N that takes the six control points (1)–(6) in Figure 7.b to the corresponding control points (1)'–(6)' in Figure 7.d. This matrix N describes how control points subdivide on and close to edges, that is, how to perform subdivision in the presence of a nonuniformity.

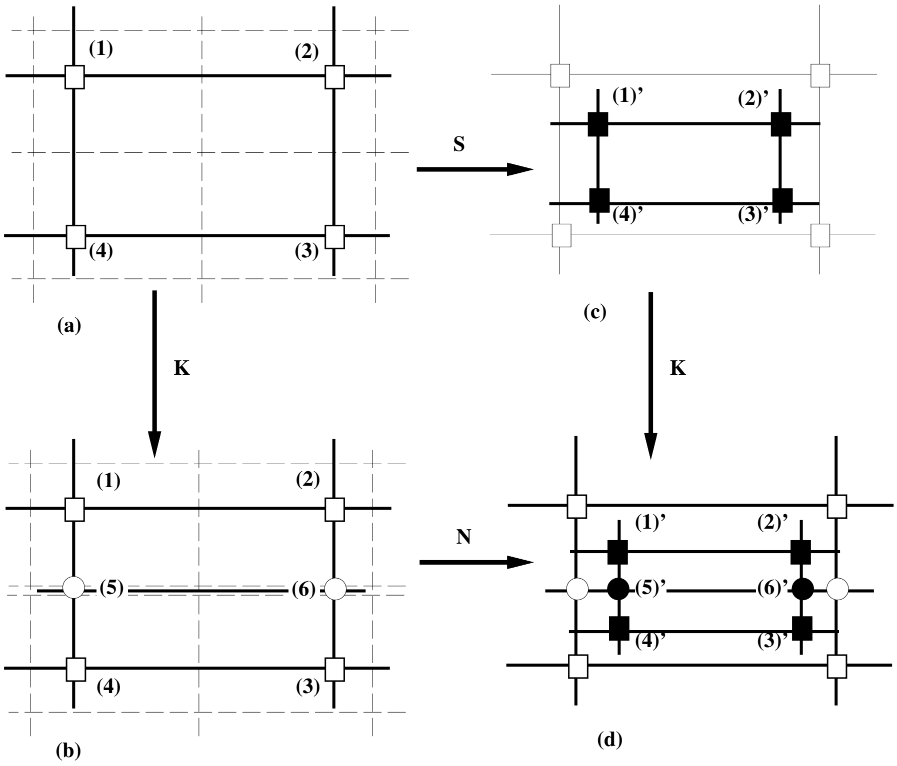

Figure 7: Commutativity of knot–line insertion and subdivision. Control points before (after) subdivision are shown in white (black). Solid lines denote polyhedral edges and dotted lines denote knot–lines.

By the commutativity of knot–line insertion and subdivision, the matrix N must satisfy:

$$N \cdot K = K \cdot S. \tag{9}$$

Assuming (the natural) symmetry between patches on the two sides of the inserted edge (in Figure 7), we can write N as a matrix of unknowns:

$$N = \begin{bmatrix} b_1 & b_2 & b_3 & b_4 & b_5 & b_6 \\ b_2 & b_1 & b_4 & b_3 & b_6 & b_5 \\ b_4 & b_3 & b_1 & b_2 & b_5 & b_6 \\ b_3 & b_4 & b_2 & b_1 & b_6 & b_5 \\ c_1 & c_2 & c_2 & c_1 & c_3 & c_4 \\ c_2 & c_1 & c_1 & c_2 & c_4 & c_3 \end{bmatrix}$$

for some $b_i, i = 1, \ldots, 6$, $c_j, j = 1, \ldots, 4$.

Now solving (9), we have many choices for the entries of N. This freedom is utilized to select values for the entries that satisfy some desirable properties:

1. Separability: Since we want to create a surface that can split across the inserted edge, new control points on one side of this edge must not depend on old control points on the other side of the edge. That is, the computation of (1)'–(2)' (in Figure 7) must not involve control points (3)–(4), similarly the computations of (3)'–(4)' must not involve (1)–(2).
   To satisfy this condition, we must pick $b_3 = b_4 = 0$. Now, substituting in (9) and solving, we get:
   $$b_1 = 1/4, b_2 = 1/4, b_5 = 1/2, b_6 = 0.$$
2. Edge Control: As described in Section 3.1, our strategy is to decompose the edge insertion problem into two steps. First, a smooth edge is inserted, introducing new control points that we call edge control points. In the second step control points are moved to sharpen the inserted edge. To decouple the two steps, the edge curve must depend entirely on edge control points. If this can be achieved, then creating C^0 surfaces will be easy since we need only move edge control points and we are guaranteed that no gaps will be introduced across the inserted edge.
   In terms of the entries of N this constraint translates to requiring that the new edge control points (5)'–(6)' depend only on the old edge control points (5)–(6).
   This implies that
   $$c_1 = c_2 = 0, c_3 = 3/4, c_4 = 1/4.$$

Substituting the values of $b_i, i = 1, \ldots, 6$, and $c_j, j = 1, \ldots, 4$ we get:

$$N = \begin{bmatrix} 1/4 & 1/4 & 0 & 0 & 1/2 & 0 \\ 1/4 & 1/4 & 0 & 0 & 0 & 1/2 \\ 0 & 0 & 1/4 & 1/4 & 1/2 & 0 \\ 0 & 0 & 1/4 & 1/4 & 0 & 1/2 \\ 0 & 0 & 0 & 0 & 3/4 & 1/4 \\ 0 & 0 & 0 & 0 & 1/4 & 3/4 \end{bmatrix}. \quad (10)$$

Using N as a subdivision matrix creates new control points that control an edge on the surface. By moving these new control points we can make this edge into a sharp edge on the surface or split the surface across the edge. In the next subsection we discuss the properties of this edge insertion algorithm.

3.3 Properties of Edge Insertion Algorithm

The only other edge and vertex insertion algorithm on non-tensor-product subdivision surfaces is the algorithm given by Hoppe *et. al* (Hoppe et al., 1994).

One of the important properties of the algorithm presented in Section 3.2, that distinguishes it from Hoppe's algorithm, is that the algorithm we propose enjoys the *space containment* property, while Hoppe's does not.

In the following section we explain the meaning and implications of this property and prove that our edge insertion algorithm indeed achieves space containment. In Section 3.3.2 other desirable properties and differences from Hoppe's scheme are discussed.

3.3.1 Space Containment

One of the fundamental and appealing properties of B–splines is that of *space containment*. In the univariate case, a vector T of nondecreasing knot values defines a linear space of B–splines. The space containment property asserts that for any other vector T_1 of nondecreasing knot values such that $T \subseteq T_1$, any spline curve on the knot vector T can be represented exactly in the space of splines specified by the knot vector T_1. That is, the space of splines defined on the finer knot vector T_1 contains the space of all splines defined on T.

Space containment is an important property of B–splines because efficient algorithms are available for the manipulation of B–splines with special knot configurations. By knot insertion, B–spline curves can be converted into forms that are easier to deal with (for example Bézier form for rendering, or for splitting splines into pieces at the knots). In other contexts, this space containment property is used to perform multiresolution analysis by representing objects in different levels of details.

To prove space containment for our edge insertion scheme, we need to build a parameterization for each surface. Then, we need to show how to express (basis) functions on the coarse parameterization in terms of basis functions on the finer parameterization, after the discontinuity is introduced. In Theorem 1 below, we prove that if the edge insertion matrix (K), interior subdivision matrix (S), and the nonuniform subdivision matrix (N) are chosen to satisfy (9), in addition to some other simple intuitive conditions, then space containment is attained. The proof is outlined in a more general setting than our particular subdivision scheme. Our scheme will be used as an example for illustration.

Before going on to introduce the proof, and for notational convenience, we extend our definitions of the interior subdivision matrix S, the nonuniform subdivision matrix N and the edge insertion matrix, K, (as defined in (7), (10), and (8) respectively for our particular scheme) to operate on full polyhedra instead of individual (quadrilateral) faces. This is done by stringing the local matrices together, since all the basis functions we use have finite supports. Thus we shall use $\mathcal{S}$ to denote the global subdivision matrix (that employs the interior subdivision rules of Section 2 in our case). Also $\mathcal{N}$ extends N by behaving identically as $\mathcal{S}$ everywhere except on (quadrilateral) faces with and close to inserted edges, where it behaves as N.

The matrix $\mathcal{K}$ is a global edge insertion matrix. It leaves all the control points of the polyhedron unchanged, but adds new control points as specified locally by K along the inserted edge. We make the intuitive assumption that $\mathcal{K}$ computes new edge vertices by taking convex combinations of control points within a strip of width w of the control points around the inserted edge. In our particular example $w = 2$ since edge control points are computed by averaging two control points as given by (8).

From the above definitions along with (9) we get:

$$\mathcal{N} \cdot \mathcal{K} = \mathcal{K} \cdot \mathcal{S}.$$

Now, following an approach similar to (DeRose et al., 1993), we can construct a parameterization for the surfaces built by $\mathcal{S}$–subdivision, over the initial polyhedron M^0. Employing this approach, we can parameterize the limit surfaces obtained by $\mathcal{S}$–subdivision using parametric basis functions, $\phi(x)$, as $\phi(x)\mathbf{p}^0$, where $x \in M^0$, and $\mathbf{p}^0$ is the vector of control points of the initial polyhedron.

Similarly we can express the limit surfaces obtained by nonuniform subdivision using $\mathcal{N}$ in terms of other parametric basis functions $\hat{\phi}(x)$, as $\hat{\phi}(x)\hat{\mathbf{p}}^0$, where $x \in M^0$ and $\hat{\mathbf{p}}^0$ is the vector of control points obtained by inserting an edge on M^0. Observe that, by construction,

$$\hat{\mathbf{p}}^0 = \mathcal{K}\mathbf{p}^0.$$

Now we can prove the main result of this section.

Theorem 1 *Let $\mathcal{S}$ be the interior subdivision matrix for a subdivision scheme. Assume $\mathcal{K}$ is an edge insertion matrix for this scheme that computes new edge control points by taking convex combinations on a w–wide strip of control points. Assume further that $\mathcal{N}$ is a nonuniform subdivision matrix, chosen to satisfy: $\mathcal{N} \cdot \mathcal{K} = \mathcal{K} \cdot \mathcal{S}$. Then,*

$$\hat{\phi}(x)\mathcal{K} = \phi(x) \quad x \in M^0.$$

Here $\phi(x)$, and $\hat{\phi}(x)$ are the vectors of parametric basis functions associated with the initial set of control points before, and after edge insertion respectively.

Proof: For a fixed initial vector of control points $\mathbf{p}^0$, we define a sequence of functions $H_i(x)$, associated with subdivision using $\mathcal{S}$ for i rounds. These functions can be expressed as

$$H_i(x) = \phi_i(x)\mathbf{p}^0 \qquad x \in M^0 \tag{11}$$

where $\phi_i(x)$ is the vector of basis functions associated with $\mathbf{p}^0$ after i rounds of subdivision using $\mathcal{S}$. Also define $H(x) = \lim_{i \mapsto \infty} H_i(x)$, and $\phi(x) = \lim_{i \mapsto \infty} \phi_i(x)$. The convergence of subdivision using $\mathcal{S}$ guarantees the existence of these limits. We define $\hat{H}_i(x), \hat{H}(x)$ and $\hat{\phi}(x)$ analogously for nonuniform subdivision using $\mathcal{N}$; thus

$$\hat{H}_i(x) = \hat{\phi}_i(x)\hat{\mathbf{p}}^0 \qquad x \in M^0. \tag{12}$$

These limits also exist by the convergence of $\mathcal{N}$–subdivision. Notice the correspondence, as well as the distinction, between vectors of control points (for example $\mathbf{p}^0, \mathcal{S}^j\mathbf{p}^0$) and the parametric functions defined on the polyhedra specified by these control points ($H_0(x), H_j(x)$ respectively).

Since $\mathcal{N}\cdot\mathcal{K} = \mathcal{K}\cdot\mathcal{S} \Rightarrow \mathcal{N}^i\cdot\mathcal{K} = \mathcal{K}\cdot\mathcal{S}^i$, we have:

$$\begin{aligned} \mathcal{K}\cdot\mathcal{S}^i\mathbf{p}^0 &= \mathcal{N}^i\cdot\mathcal{K}\mathbf{p}^0 \\ &= \mathcal{N}^i\hat{\mathbf{p}}^0 \end{aligned}$$

The set of control points on the right hand side correspond to $\hat{H}_i(x)$. On the left hand side we have $\mathcal{K}\cdot\mathcal{S}^i\mathbf{p}^0$. Since the new points introduced by $\mathcal{K}$ are convex combinations of control points within a strip of width w of control points in $\mathcal{S}^i\mathbf{p}^0$, and subdivision is a self–similar refinement process, this strip shrinks in width with i as shown in Figure 8. It follows that for points not on the inserted edge, though arbitrarily close, we can always find j such that $H_j(x) = \hat{H}_j(x)$. Agreement on the edge itself follows form the continuity of $H(x)$.

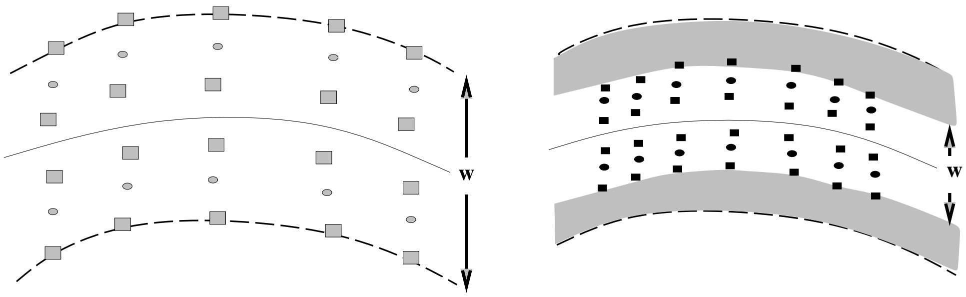

Figure 8: Control points used for edge insertion before (left) and after (right) one round of subdivision.

Thus we conclude that $\lim_{i\to\infty}|H_i(x) - \hat{H}_i(x)| \to 0$. It follows that $\lim_{i\to\infty}|H(x) - \hat{H}(x)| \to 0$, and thus $H(x) = \hat{H}(x)$.
By setting the vectors of initial control points equal to $\delta = \{\cdots, 0, 1, 0, \cdots\}$, we get a representation of the basis functions $\phi(x)$ in terms of $\hat{\phi}(x)$, and thus we get space containment. $\square$

3.3.2 Other Desirable Properties

In addition to the space containment property of the last section, our edge insertion algorithm enjoys the following important properties:

Boundary management: By construction, (the separability condition of Section 3.2), the computations of new control points on one side of an inserted edge do not depend on the control points on the other side of this edge. Actually, nothing in the construction necessitates the existence of control points on both sides of the edge. Thus, the boundary of the control polyhedron can be treated exactly as an inserted edge. (Corner control points still need to be addressed. They will be discussed along with vertex insertion in Section 4).

Edge curve is a B–spline: The rules computed in Section 3.2 for subdivision along an inserted edge, as given by the last two rows of N in (10), are exactly the rules used by Chaikin's algorithm to compute a quadratic B–spline with uniform knots. This is another important property because there are available techniques (e.g. Blossoming (Goldman, 1990)) for computing the control points of a B–spline given other representations of the curve. It also allows for the use of the full arsenal of B–spline design tools to design the shapes of edges and boundaries.

Figure 9 shows a torus with an inserted edge across the middle faces and moving the control points of the inserted curve for sharpenning the edge or splitting the surface.

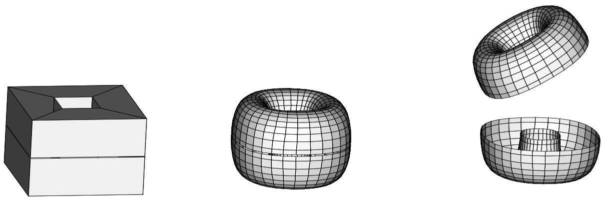

Figure 9: Moving contol points to split across an inserted edge.

We remark that our subdivision scheme produces C^1 piecewise quadratic surfaces on regular connectivity regions of the polyhedron, compared to the piecewise C^2–quartic surfaces produced by Hoppe *et. al.* In fact, this is the main reason we have chosen our case study box spline subdivision scheme, since it produces surfaces with the lowest possible polynomial degree for a C^1 scheme.

3.4 Examples

Figure 10 shows some surfaces with sharp edges and with boundaries.

4 Inserting Vertices

The techniques of Section 3 enable us to insert (possibly sharp) edges across four sided polyhedral faces. Realistically the requirement that faces be four sided is not a severe restriction to impose since, as shown in Figure 4, after one

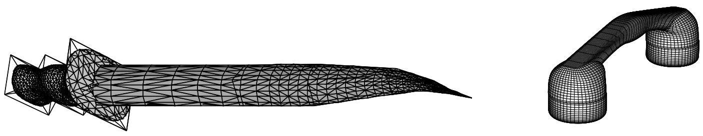

Figure 10: Surfaces with (sharp) edges and boundaries

round of subdivision all faces become four–sided except for a few extraordinary faces. Also, after more and more subdivision larger and larger regions of these extraordinary faces are covered with four–sided faces. Thus if we want to insert an edge that goes through but not to the centroid of an extraordinary face, we can use the already developed technique, though after some number of subdivision steps. The edges we can insert, however, are only maps of iso–parametric grid lines.

In this section we address the problem of how to insert a (possibly sharp) vertex at the centroid of an extraordinary face, and also the important question of what to do at the end points of an inserted edge, or if the edge terminates at the centroid of an extraordinary face.

Returning to our objective of building surfaces that can split/decompose, vertex insertion means:

1. Giving a scheme for breaking up a polyhedral surface, around the centroid of an arbitrary face, into pieces that can be subdivided separately then assembled to regenerate the surface obtained by subdividing the whole polyhedron using the interior subdivision rules of Section 2.

2. Specifying user control over the individual polyhedral pieces so that the inserted vertex can be made into a sharp vertex.

We begin by giving the construction for the insertion of a vertex of valence four at the centroid of a four–sided face in Section 4.1, then address general valence vertices in Section 4.2.

4.1 Vertices of Valence Four

As with edge insertion, the main idea here is the commutativity of vertex insertion and subdivision. Using the indexing of control points shown in Figure 11, we can write the vertex insertion matrix K_4 and the nonuniform subdivision matrix N_4 (assuming natural symmetry) as:

$$K_4 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1/2 & 1/2 & 0 & 0 \\ 0 & 1/2 & 1/2 & 0 \\ 0 & 0 & 1/2 & 1/2 \\ 1/2 & 0 & 0 & 1/2 \\ 1/4 & 1/4 & 1/4 & 1/4 \end{bmatrix}, \qquad N_4 = \begin{bmatrix} b_1 & b_2 & b_3 & b_2 & b_4 & b_5 & b_5 & b_4 & b_6 \\ b_2 & b_1 & b_2 & b_3 & b_4 & b_4 & b_5 & b_5 & b_6 \\ b_3 & b_2 & b_1 & b_2 & b_5 & b_4 & b_4 & b_5 & b_6 \\ b_2 & b_3 & b_2 & b_1 & b_5 & b_5 & b_4 & b_4 & b_6 \\ c_1 & c_1 & c_2 & c_2 & c_3 & c_4 & c_5 & c_4 & c_6 \\ c_2 & c_1 & c_1 & c_2 & c_4 & c_3 & c_4 & c_5 & c_6 \\ c_2 & c_2 & c_1 & c_1 & c_5 & c_4 & c_3 & c_4 & c_6 \\ c_1 & c_2 & c_2 & c_1 & c_4 & c_5 & c_4 & c_3 & c_6 \\ d_1 & d_1 & d_1 & d_1 & d_2 & d_2 & d_2 & d_2 & d_3 \end{bmatrix}.$$

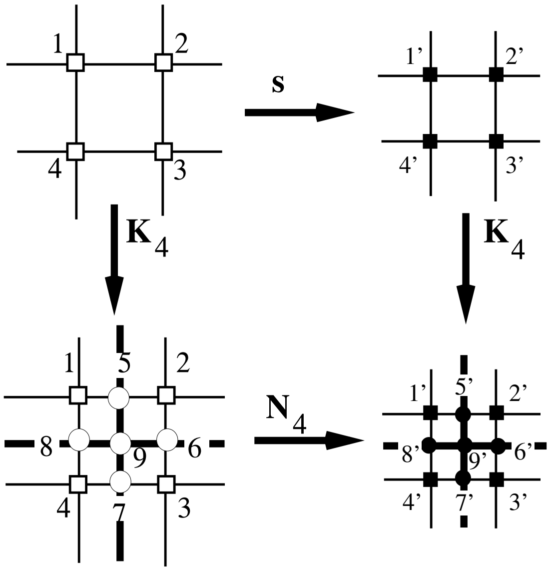

Figure 11: Vertex insertion, valence 4

The entries of N_4 are computed from the commutativity diagram (Figure 11) by solving the system of equations:

$$N_4 \cdot K_4 = K_4 \cdot S, \tag{13}$$

where S is the interior four–sided face subdivision matrix of the regular box spline (4).
The solution must satisfy the following properties:

1. Separability: The surface can be broken into four patches around the inserted vertex. This means that new control points in each patch must

depend only on old control points in the same patch. In terms of the entries of N_4 this implies that:

$$b_2 = b_3 = b_5 = 0 \quad \text{and} \quad c_2 = c_5 = 0.$$

2. Interpolation of inserted vertex: Since the surface before vertex insertion does not have any holes, all four patches must interpolate the inserted vertex. One subdivision rule that guarantees this property is obtained by picking $d_1 = d_2 = 0$ and $d_3 = 1$.

3. Edge curves in each patch are controlled entirely by control points on the edge. This condition implies that $c_1 = 0$.

4. Now solving (13) for the remaining entries of N_4 we get:

$$b_1 = b_6 = 0, \quad b_4 = 1/2, \quad c_3 = c_6 = 1/2, \quad c_4 = 0.$$

Thus,

$$N_4 = \begin{bmatrix} 0 & 0 & 0 & 0 & 1/2 & 0 & 0 & 1/2 & 0 \\ 0 & 0 & 0 & 0 & 1/2 & 1/2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1/2 & 1/2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1/2 & 1/2 & 0 \\ 0 & 0 & 0 & 0 & 1/2 & 0 & 0 & 0 & 1/2 \\ 0 & 0 & 0 & 0 & 0 & 1/2 & 0 & 0 & 1/2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1/2 & 1/2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{14}$$

The matrix N_4 in (14) solves the first part of the vertex insertion problem: how to break the surface into four patches around a valence four vertex and how to subdivide these patches independently to get a smooth surface identical to the surface obtained by subdividing the whole control polyhedron using the box spline rules. The second part, concerning changing this inserted vertex into a sharp vertex, is easy since all four patches interpolate the inserted vertex. To maintain C^0 continuity of the composite surface, however, control points common between patches (e.g. control points (5)–(8) in Figure 11) need to be moved consistently in neighboring patches.

Two observations on the matrix N_4 in (14) are in order:

- Extending our approach in Section 3 of treating boundaries as inserted edges, it is easy to see that we can use the subdivision matrix N, for subdivision at and close to corners by treating a corner control point as an inserted vertex of valence four.

- As shown in Section 3, inserted edges are quadratic B–splines with uniform knots. Using the subdivision matrix N_4 in (14) at the end point of an edge corresponds to doubling the knot at the end of the corresponding B–spline curve.

4.2 Arbitrary Valence Vertices

Ideally we would like to treat the vertex insertion problem uniformly, independent of the valence. To do that we need to draw a commutativity diagram similar to Figure 11 and use the corresponding system of equations to derive the special subdivision rules for vertex insertion. However, some problem arises in using this technique when the valence is not 4. The next subsection explains this problem. Section 4.2.2 presents our approach to getting around this problem, and Section 4.2.3 gives the actual construction.

4.2.1 The Decomposition Problem

To illustrate the problem with extending the valence 4 approach to other valences, we examine the problem of inserting a vertex at the centroid of the pentagonal face shown in Figure 12.

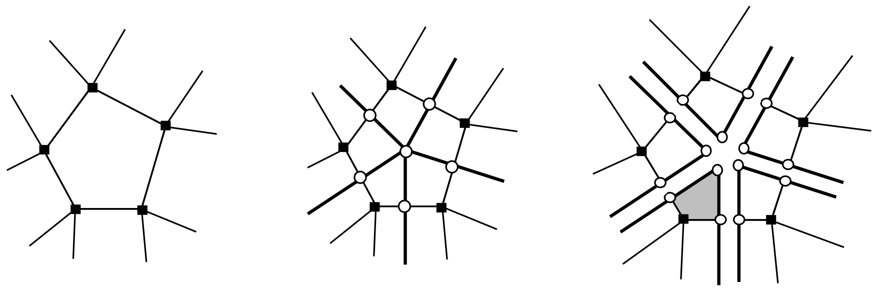

Figure 12: Vertex insertion, valence 5. Before insertion, points in the shaded part of the surface depend on 5 control points (n in general), while they depend on exactly 4 control points after vertex insertion.

Before inserting the vertex at the centroid of the pentagon, points in the shaded area (of Figure 12) are computed using the mask in (7), thus they belong to a five dimensional space spanned by all five vertices of the pentagon (marked with black squares in the figure). To split this face without perturbation into five patches, we must reproduce these points in the shaded area using only the four control points enclosing the shaded patch. Since points computed from these four points span only a four dimensional space, this reproduction can not generally be done.

For valence four vertices this problem does not arise because space dimensions match. For the general case, the reason for the above problem (we call it *decomposition problem* henceforth) is that after splitting the surface, the subdivision algorithm operating on individual patches does not have enough information to reproduce the corresponding part of the original surface.

4.2.2 A Solution to the Decomposition Problem

We solve the decomposition problem by passing to the subdivision process, *on each patch*, the data of all the vertices of the extraordinary face. Thus a vertex inserted at the centroid of an extraordinary face is treated as a special vertex. It carries the coordinates of all the vertices of the enclosing face. We call such vertex a *composite vertex.*

The decomposition of the surface for the example in Figure 12 can now be thought of as shown in Figure 13.

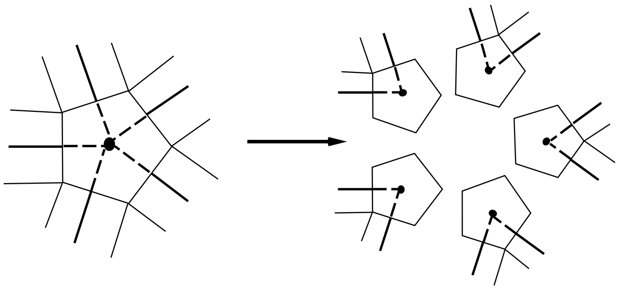

Figure 13: Patch decomposition of a valence five face

4.2.3 Subdividing Composite Vertices

If we decompose patches around extraordinary faces using composite vertices, we shall have enough information subdividing each patch to reproduce the corresponding part of the original surface. This solves the break–up part of the vertex insertion problem.

Now, we study how patches with composite vertices at their corners subdivide. We go back to our current example of a five sided face for illustration. One of the five patches that we get is shown in Figure 14.

In Figure 14, control points marked with circles control the shape of the patch while those marked with squares can be computed from the circled ones. Now the procedure to subdivide a patch with a composite vertex at the corner is as follows:

Algorithm 1 *Subdivision with Composite Vertices:*

1. *Subdivide all interior faces using the interior rules in (7) and faces on the boundaries, other than those containing composite vertices, using the nonuniform rules in (10).*

2. *Perform interior subdivision, using the rules in (7), on the vertices of the composite vertex.*

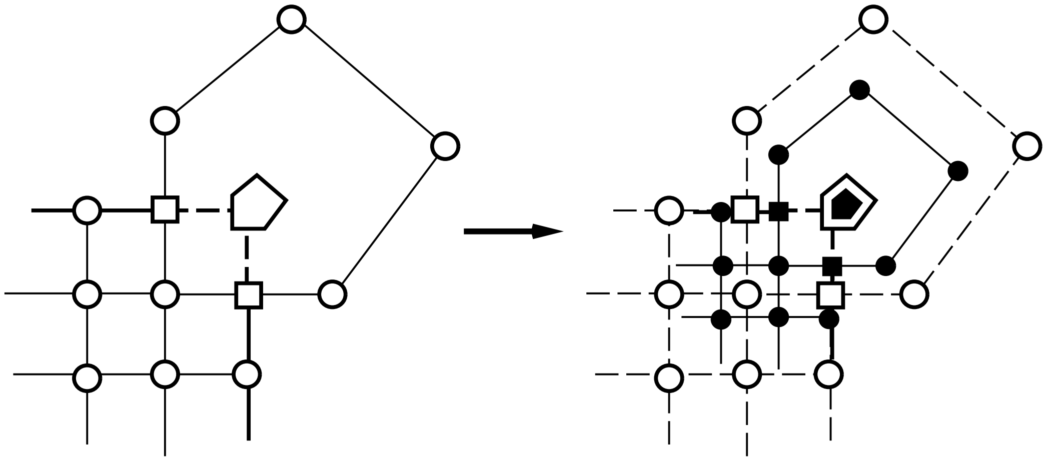

Figure 14: One of the five patches obtained by inserting a valence five vertex. The composite vertex at the centroid contains the information of all five vertices of the pentagon. White (black) vertices are those before (after) subdivision

3. *Compute two new mid–side control points (marked with black squares for the valence five example of Figure 14) by averaging the control points of the corresponding two subdivided sides.*
4. *Save the subdivided polygon of the composite vertex as new composite vertex.*

This subdivision algorithm (Algorithm 1) results in a decomposition of the smooth surface around the centroid of an arbitrary face. The following Proposition shows that the surface resulting from the vertex insertion process has tangent plane continuity everywhere on the extraordinary face.

Proposition 1 *Patches meeting with a common composite vertex form a tangent plane continuous limit surface over the common face.*

Proof: The result follows directly from the construction since every point is computed using the exact same formulas before or after vertex insertion. □

For the second part of the vertex insertion problem, that is sharpening the inserted vertex, it suffices to move the vertices of the composite vertex differently between patches. This step, however, must be performed very carefully to avoid introducing gaps into the surface.

4.2.4 Examples

Figure 15.a shows a patch with a valence three vertex inserted at its centroid, while Figure 15.b shows a valence five example.

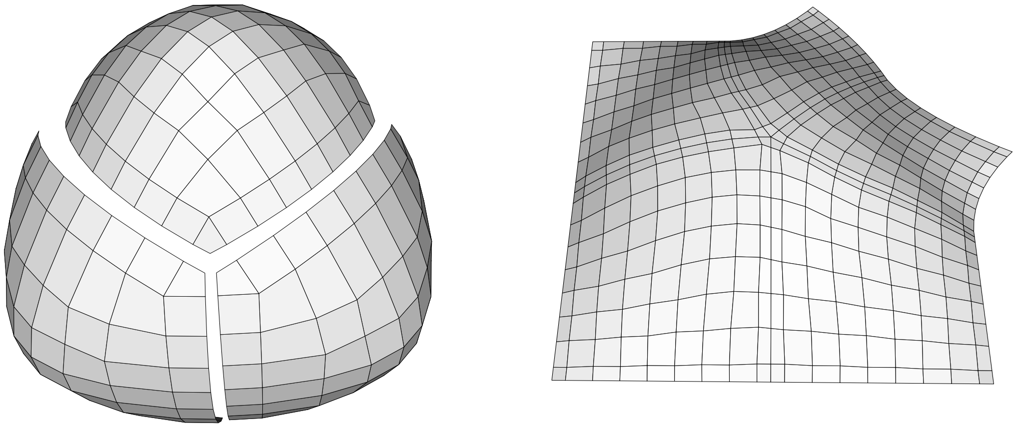

Figure 15: Vertex insertion examples.

5 Other Subdivision Schemes

Other subdivision schemes can make use of the methodology developed in this paper for edge and vertex insertion. Here we list a set of factors that need to be considered when extending the approach to other subdivision schemes:

- A major factor in deciding the difficulty of the extension is the (size of the) support of basis functions. For the case–study box spline only two rows of control points entirely control the surface midway between them, generally this may be bigger and more new subdivision rules may need to be developed.
- The Edge control property followed from the fact that we were doubling a knot on a quadratic B–spline. For other subdivision schemes the inserted edges need not be quadratic B–splines, and the effect of the edge insertion in the parameter domain needs to be studied.
- The commutativity idea remains valid and can still be used in other settings to derive nonuniform subdivision rules.

6 Conclusions

A technique for edge and vertex insertion on a class of C^1 subdivision surfaces is presented. The approach uses the commutativity of knot insertion and subdivision to derive subdivision rules along and close to edges and vertices. The main advantage of the surfaces built using the proposed approach is that they can be split exactly into pieces across edges or centroids of faces with an arbitrary number of sides. The approach also allows for subdivision next to boundaries, handles open polyhedra, and allows for the insertion of sharp edges and vertices on the limit surfaces.

Some questions are still left to be answered. For example, we would like to know if there exist subdivision rules that build tangent plane continuous limit surfaces but allow for the complete decomposition of the surface around extraordinary vertices without the need for the composite vertices machinery. Extending our technique to other subdivision schemes is another interesting research problem that needs more investigation.

References

Catmull, E. and Clark, J. (1978). Recursively generated B-spline surfaces on arbitrary topological meshes. *Computer-aided Design*, 10, 350–355.

Chaikin, G. (1974). An algorithm for high speed curve generation. *Computer Graphics and Image Processing*, 3, 346–349.

Davis, P. (1979). *Circulant Matrices.* Wiley Interscience, New York.

de Boor, C., Höllig, K., and Riemenschneider, S. (1993). *Box Splines.* Springer–Verlag.

DeRose, T., Lounsbery, M., and Warren, J. (1993). Multiresolution analysis for surfaces of arbitrary topological type. Technical Report 93-10-05, University of Washington, Department of Computer Science and Engineering.

Doo, D. and Sabin, M. (1978). Behaviour of recursive division surfaces near extraordinary points. *Computer-aided Design*, 10, 356–360.

Dyn, N. and Levin, D. (1990). A butterfly subdivision scheme for surface interpolation with tension control. *ACM Transactions on Graphics*, 9, 160–169.

Farin, G. (1986). Triangular Bernstein-Bézier patches. *Computer Aided Geometric Design*, 3(2), 83–128.

Goldman, R. (1990). Blossoming and knot insertion algorithms for B–spline curves. *Computer Aided Geometric Design*, 7, 69–81.

Goodman, T. and Unsworth, K. (1994). Injective bivariate maps. Technical Report CS 94/02, University of Dundee, Mathematics and Computer Science.

Hoppe, H., DeRose, T., Duchamp, T., Halstead, M., Jin, H., McDonald, J., Schweitzer, J., and Stuetzle, W. (1994). Piecewise smooth surface construction. *Computer Graphics*, pages 295–302.

Loop, C. (1987). Smooth subdivision based on triangles. Master's thesis, University of Utah.

Nasri, A. (1997). Curve interpolation in recursively generated b-spline over arbitrary topology. *Computer Aided Geometric Design*, 14, 13–30.

Peters, J. and Reif, U. The simplest subdivision scheme for smoothing polyhedra. *ACM Transactions on Graphics*. To appear.

Reif, U. (1995). A unified approach to subdivision algorithms near extraordinary vertices. *Computer Aided Geometric Design*, 12, 153–174.

Riesenfeld, R. (1975). On Chaikin's algorithm. *Computer Graphics and Image Processing*, 4, 304–310.

Warren, J. (1995). Subdivision methods for geometric design. Unpublished Monograph.

Appendix: Proof of Tangent Plane Continuity and Regularity

The first C–condition is built into the construction of the subdivision mask of Section 2.2.2. Thus, we only need to verify the second C–condition.

In Reif's analysis subdivision is treated as a hole filling process. Each face is treated as a hole that gets filled gradually by subdivision. Tangent plane continuity of the limit surface on the inside of a face is proved by:

1. Computing the second and third eigenvectors of the subdivision matrix and pairing their entries to get a set of (two-dimensional) control points.
2. Subdividing the polyhedron obtained by connecting the resulting control points to get a surface.
3. An annular region of the resulting subdivision surface is called the characteristic map of the subdivision scheme. The width of this annular region depends on the specific subdivision scheme and the size of the support of its basis functions. Reif (Reif, 1995) proves that if this surface is regular and forms an injective map of the parameter space then the subdivision scheme produces tangent plane continuous and regular limit surfaces.

We begin the proof by recalling one standard result from linear algebra (Davis, 1979), which states that the eigenvectors of the $n \times n$ circulant matrix S_n, corresponding to the repeated second largest eigenvalue, can be written as:

$$v_2 = \mathcal{R}e\left[1, \omega, \omega^2, \ldots, \omega^{(n-1)}\right]^T, \quad v_3 = \mathcal{I}m\left[1, \omega, \omega^2, \ldots, \omega^{(n-1)}\right]^T, \qquad (15)$$

where $\omega = e^{\frac{2\pi i}{n}}$.

Actually any pair of eigenvectors v_2, v_3 of S_n corresponding to eigenvalue $1/2$ are not unique since by scaling and taking linear combinations of any such pair we get other eigenvectors for S_n. However, the selection in (15) bears a special advantage because it leads to simple formulas for the eigenvector entries and exploits the symmetry, around the origin, of the eigenvector entries.

For the rest of the proof we shall use a single vector with complex entries (V) to represent both eigenvectors v_2, v_3. This vector V is defined by: $V = v_2 + iv_3$. Using this interpretation, plotting v_2 against v_3 corresponds to plotting the entries of V in the complex plane. For example, from (15) we have

$$V = \left[1, \omega, \omega^2, \ldots, \omega^{(n-1)}\right]^T \quad \text{where} \quad \omega = e^{\frac{2\pi i}{n}}. \tag{16}$$

Henceforth we shall fix ω to denote the n^{th} root of unity $e^{(2\pi i/n)}$, and θ to denote the angle $2\pi/n$, thus $\omega = e^{i\theta}$.

To compute the characteristic map for the subdivision scheme at hand we need to compute an extension $\overline{V}$ of the eigenvector V in (16), to a 3–ring of control points around an n–sided face. This requires building the subdivision matrix $\overline{S}_n$ over a 3–ring of control points. Figure 16 shows the control points in the support of $\overline{S}_n$.

From here on we shall use double indexing for the control points, as shown in Figure 16. Control points are arranged in blocks of 9–points each. The first index denotes the number of the control point within the block, while the second index denotes the block number.

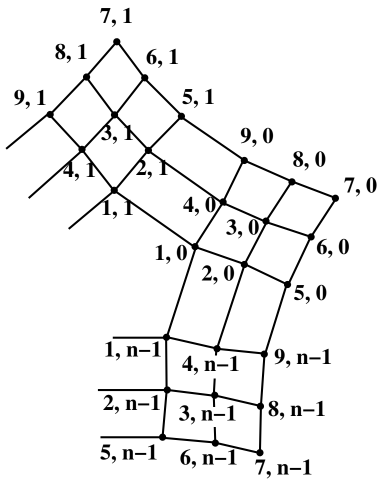

Figure 16: The support of the extended subdivision matrix $\overline{S}_n$ over a 3–ring.

The following lemma gives formulas for the entries of $\overline{V}$.

Lemma 1 *Let $\overline{V}$ be an eigenvector of $\overline{S}_n$ (the extension of S_n to a 3–ring of control points) that extends the eigenvector V in (16). Then, using the indexing of control points in Figure 16, we have:*

$$\begin{aligned}
&1. \quad \overline{V}_{1,j} = \omega^j. && (17)\\
&2. \quad \overline{V}_{2,j} = (2+\omega^{-1})\omega^j. && (18)\\
&3. \quad \overline{V}_{3,j} = (3+(\omega+\omega^{-1})/2)\omega^j. &&\\
&4. \quad \overline{V}_{4,j} = (2+\omega)\omega^j. &&\\
&5. \quad \overline{V}_{5,j} = (3+2\omega^{-1})\omega^j. &&\\
&6. \quad \overline{V}_{6,j} = (4+\omega^{-1}+(\omega+\omega^{-1})/4)\omega^j. &&\\
&7. \quad \overline{V}_{7,j} = (5+(\omega+\omega^{-1}))\omega^j. &&\\
&8. \quad \overline{V}_{8,j} = (4+\omega+(\omega+\omega^{-1})/4)\omega^j. &&\\
&9. \quad \overline{V}_{9,j} = (3+2\omega)\omega^j. &&
\end{aligned}$$

where $\omega = e^{2\pi i/n}$ and $j = 0, \ldots, n-1$

Proof:
Grouping control points with indices $\{1, j\}$, $j = 0, \ldots n-1$ together, and similarly for control points with indices $\{k, j\}$, $k = 2, \ldots 9$, we can write $\overline{S}_n$ in block form as:

$$\overline{S}_n = \frac{1}{4}\begin{bmatrix} 4S_n & 0 & 0 & 0 & 0 & \ldots & 0 \\ 2I+L & I & 0 & 0 & 0 & \ldots & 0 \\ 2I & I & 0 & I & 0 & \ldots & 0 \\ 2I+R & 0 & 0 & I & 0 & \ldots & 0 \\ I & 2I & 0 & L & 0 & \ldots & 0 \\ I & 2I & I & 0 & 0 & \ldots & 0 \\ 0 & I & 2I & I & 0 & \ldots & 0 \\ I & 0 & I & 2I & 0 & \ldots & 0 \\ I & R & 0 & 2I & 0 & \ldots & 0 \end{bmatrix}$$

where I is the $n \times n$ identity matrix, R, L are $n \times n$ permutation matrices specified by their first rows:

$$\begin{aligned} R[1,i] &= [0,1,0,\ldots,0]. \\ L[1,i] &= [0,0,\ldots,0,1]. \end{aligned}$$

Now we solve for the entries of $\overline{V}$ using the property that $\overline{V}$ is an eigenvector of $\overline{S}_n$ corresponding to eigenvalue $1/2$. That is,

$$\overline{S}_n \cdot \overline{V} = \overline{V}/2, \qquad (19)$$

and we notice:

1. Formula (17) is a restatement of (16).

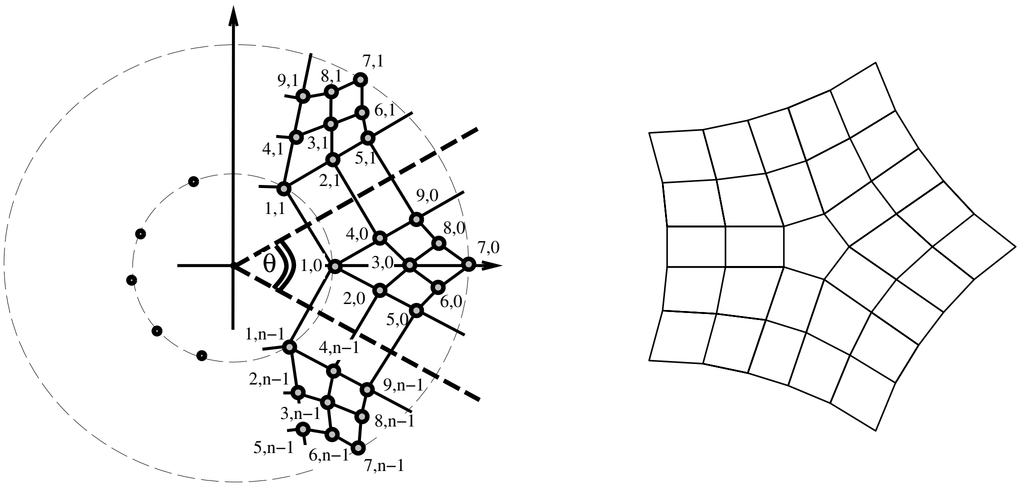

Figure 17: Plotting the entries of $\overline{V}$ in the complex plane. A valence 5 example is also shown to the right.

2. Using the second block row of $\overline{S}_n$ in (19) we get one equation in one unknown $\overline{V}_{2,j}$:

   $$\omega^j/2 + \omega^{j-1}/4 + \overline{V}_{2,j}/4 = \overline{V}_{2,j}/2,$$

   which solves to (18).

   By similar computations, utilizing the block form of $\overline{S}_n$, we obtain the rest of the formulas in the lemma statement. □

Figure 17 shows the layout of the entries of $\overline{V}$ in the complex plane, and a valence five example. Inspecting the geometry in Figure 17, and using the values of the entries of $\overline{V}$ computed in Lemma 1, we observe the following:

Lemma 2 *In the plot of the eigenvector $\overline{V}$ computed in Lemma 1, (Figure 17), the following statements hold:*

1. *The plot of V is n-way symmetric around the origin.*
2. *The control points with index $\{k, j\}$, $k = 1, \ldots, 9$ for fixed $j \in \{0, \ldots, n-1\}$ lie in the sector specified by :*

$$\left.\begin{array}{l} \sin((j-1/2)\theta)x - \cos((j-1/2)\theta)y < 0 \\ \sin((j+1/2)\theta)x - \cos((j+1/2)\theta)y > 0 \end{array}\right\} \tag{20}$$

where $\theta = 2\pi/n$.

Proof:
The proof of the first part follows directly from Lemma 1 by observing that multiplication by the n^{th} root of unity ω corresponds to a positive rotation by an angle $2\pi/n$ about the origin of the complex plane.

For the second part, substitute the entries of $\overline{V}$ computed in Lemma 1 in (20). For example, substituting for (x, y) by $(\mathcal{R}e[\overline{V}_{4,j}], \mathcal{I}m[\overline{V}_{4,j}])$, and using,

$$a = (j - 1/2)\theta, \quad b = j\theta, \quad c = (j+1)\theta, \quad d = (j + 1/2)\theta$$

we get:

$$\begin{aligned}
\sin(a)x - \cos(a)y &= sin(a)(2\cos(b) + \cos(c)) - \cos(a)(2\sin(b) + \sin(c)) \\
&= 2\sin(a)\cos(b) + \sin(a)\cos(c) - 2\cos(a)\sin(b) - \cos(a)\sin(c) \\
&= 2\sin(a-b) + \sin(a-c) \\
&= 2\sin(-\theta/2) + \sin(-\theta/2) < 0 \qquad \text{for all } n \geq 3
\end{aligned}$$

Similarly,

$$\begin{aligned}
\sin(d)x - \cos(d)y &= \sin(d)(2\cos(b) + cos(c)) - \cos(c)(2\sin(b) + \sin(c)) \\
&= 2\sin(a)\cos(b) + \sin(a)\cos(c) - 2\cos(a)\sin(b) - \cos(a)\sin(c) \\
&= 2\sin(a-b) + \sin(a-c) \\
&= 2\sin(\theta/2) + \sin(-\theta/2) > 0 \qquad \text{for all } n \geq 3 \quad \square
\end{aligned}$$

The last lemma asserts that the control points of the eigenvector $\overline{V}$ with indices $\{k, j\}$, $k = 1, \ldots, 9$ lie in the sector specified by the two lines (20). For example, the control points with indices $\{1, 0\}$–$\{9, 0\}$ lie in the sector of angle θ enclosed by the dotted lines in Figure 17.

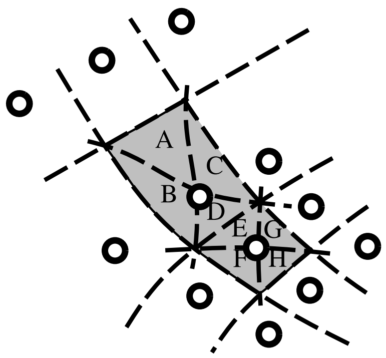

Figure 18: Dotted lines are isoparametric lines. We need to prove injectivity and regularity only over the shaded region.

It follows that it is sufficient to analyze the smoothness (injectivity and regularity) of the characteristic map only on one sector of the map shown in Figure 17. In particular, drawing the iso–parametric lines as shown in Figure 18, and employing symmetry, we need to prove the injectivity and regularity of the limit surface only on the shaded triangular regions marked (A)–(H).

To perform this analysis (of injectivity and regularity) we express the limit surface in terms of Bézier coefficients. This is done by expressing one basis function of the box spline at hand in Bézier form using the property that the basis function is C^1 piecewise–quadratic and utilizing the conditions given by (Farin, 1986) for tangent plane continuity of a surface expressed in triangular Bézier form across two triangles. Now we (use Mathematica to) derive the

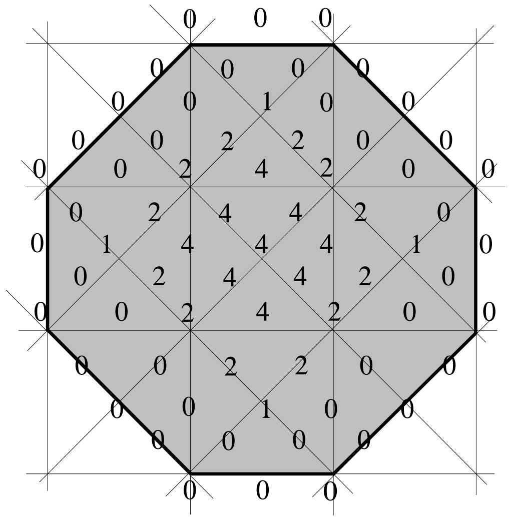

Figure 19: Bézier coefficients of a basis function of the box spline (normalized by 16).

Bézier coefficients of the limit surface over the shaded area in Figure 18.

Injectivity of the limit surface is verified using the characterization by (Goodman and Unsworth, 1994) of the conditions under which Bézier coefficients produce injective bivariate maps. For a quadratic, their condition amounts to verifying nine inequalities on the Bézier coefficients. Regularity is also proved using the Bézier representation, by computing the Jacobian over each of the shaded triangles and showing that it does not vanish. The Mathematica code used to compute the Bézier coefficients and verify both the injectivity and regularity of the characteristic map is available from Rice University's WEB-page under "http://www.cs.rice.edu/jwarren/edgeins.ma".

Ayman Habib,
Parametric Technology Crop.,
2590 N. First St.,
San Jose, CA 95131.
ahabib@sj.ptc.com

Joe Warren,
Computer Science Dept.,
Rice University,
6100 South Main St.
Houston, TX 77251–1892,
jwarren@cs.rice.edu