Analysis and Application of Subdivision Surfaces

Jean E. Schweitzer

Department of Computer Science and Engineering
University of Washington
Seattle, Washington 98195

August 1996

Technical Report UW-CSE-96-08-02

This technical report is the author's PhD dissertation.

University of Washington

Abstract

Analysis and Application of Subdivision Surfaces

by Jean E. Schweitzer

Chairperson of Supervisory Committee: Professor Anthony D. DeRose
Department of
Computer Science and Engineering

Subdivision surfaces are a convenient representation for modeling objects of arbitrary topological type. In this dissertation, we investigate the analysis of a piecewise smooth subdivision scheme, and we apply the scheme to reconstruct objects from non-uniformly sampled data points.

Defined as the limit of repeated refinement of a mesh of 3D control points, subdivision surfaces require analysis to establish convergence to a well-defined, tangent plane smooth G1 surface. Recent research has focused on analyzing smooth surface schemes in which the rules are symmetrical about each vertex and edge. However, a scheme for creating surfaces with sharp features has rules that do not exhibit this symmetry. In this dissertation, we extend the use of eigenanalysis and characteristic maps to analyze a piecewise smooth subdivision scheme that generalizes quartic triangular B-spline surfaces.

Subdivision surfaces are suitable for optimized surface fitting and have been used in the reconstruction of objects from 3D data. Previous methods have created accurate representations of objects from dense and uniform data samples. As a practical low cost alternative, we present an algorithm for creating a subdivision surface from data sampled uniformly along closed curves and non-uniformly within the regions they enclose.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I wish to thank my advisor Tony DeRose and mathematics professor Tom Duchamp for their help and guidance in this work. It has been a priviledge to be able to work with this team, and I appreciate their encouragement and patience.

I particularly wish to thank Tony for encouraging me to reenter the world of geometric modeling from which I had strayed since my happy introduction to it at the Boeing Company.

I wish to thank Hugues Hoppe whose thesis work provided motivation for my work and Kari Pulli for his assistance in displaying my surfaces.

I also wish to thank all the people in the U.W. Computer Science and Engineering Department who have given me technical and moral support along the long way, including Steve Mann, Michael Lounsbery, Georges Winkenbach, Adam Finkelstein, and Dani Lischinski.

And I want to especially thank Curt Geertgens for his conscientious editing assistance and his warm friendship through these final months of hard work.

Finally, I thank the Spatial MetriX Corporation of Kennett Square, PA for providing me with a data set for my application.

Chapter 1

INTRODUCTION

Subdivision curves and surfaces are valued in geometric modeling applications for their convenience and flexibility. They permit the representation of objects of arbitrary topological type in a form that is easy to design, render and manipulate [12, 25]. While they can be used to model smooth objects, they can also be extended to model objects with boundaries and sharp features [13].

Subdivision schemes that generalize B-spline representations [4, 8, 17, 13] are particularly useful. Although B-spline (and NURB) surface representations are a prevalent representation in geometric modeling, they cannot model objects of non-planar topology or objects possessing sharp features without cumbersome patch stitching and curve trimming. However, B-spline representations are easy to analyze because they are piecewise polynomial (or rational) in form.

Subdivision surfaces, on the other hand, are defined as the limit of repeated refinement of a 3D control point mesh. In general, this limit does not result in a closed form representation. Therefore, conventional methods of analyzing surfaces are inadequate, since they depend on representations such as polynomial or rational functions. Determining whether the limit subdivision surface is well defined and tangent plane continuous has become the subject of more recent research [2, 27]. There are also techniques for deriving exact formulas for points and normals on the limit surfaces [12].

The elegant simplicity and practical advantages of subdivision algorithms have made them useful for particular applications. For example, they are ideal for quickly rendering an object where a piecewise planar model suffices. With the capability to calculate exact points and normals on the limit surface, subdivision algorithms are finding wider acceptance for applications that require smooth surfaces. With the added capability of creating sharp features, subdivision surfaces can be applied to represent even more realistic objects.

Subdivision surfaces are also making fundamental contributions to new application areas in geometric modeling. For example, they are a key element in the multiresolution representation of objects [18], and they are a convenient representation for optimized surface fitting [12, 16].

In this dissertation, we examine two topics related to subdivision surfaces. First, we show how a piecewise smooth subdivision scheme can be analyzed, and we derive formulas for points and tangents on the limit surface. Second, we present an engineering application, for which we describe an algorithm for creating a surface that approximates an object from non-uniformly sampled data points and interpolates boundary curves.

1.1 Previous and Related Work

1.1.1 Analysis of Subdivision Surfaces

Uniform subdivision schemes have been given thorough and formal treatment, including conditions for higher order continuity. See Cavaretta, Dahmen and Micchelli [5], for example. However, these schemes apply only to regular meshes, where the same subdivision rule can be applied at each vertex, edge, or face. In fact, early surface subdivision schemes were motivated by the need to define surfaces over non-regular control meshes.

Subdivision schemes for surfaces were first introduced by Catmull and Clark [4] and by Doo and Sabin [8] in 1978. They generalized bicubic and biquadratic B-splines, respectively, to achieve surfaces of arbitrary topological type by introducing irregularities in a rectangular control mesh. In 1987, Loop [17] presented his scheme that generalizes quartic triangular B-splines (also known as quartic Box-splines).

With various degrees of formality, these authors have given arguments that their schemes converge to tangent plane smooth surfaces at all points. In particular, Doo and Sabin and Loop apply Fourier analysis, taking advantage of the cyclic nature of their schemes.

Ball and Storry [2] provide a more in-depth treatment of schemes on a rectangular mesh. They use Fourier transforms to give an eigenanalysis of the linear subdivision map to find conditions on the scheme that guarantee convergence to a tangent plane smooth surface at extraordinary vertices. In a previous paper [1], these authors take a direct matrix approach to the eigenanalysis.

Halstead, Kass, and DeRose [12] present an eigenanalysis that can be applied to any stationary subdivision map and suggest sufficient conditions for its convergence and for the existence of a well-defined tangent plane at extraordinary points. They also show how formulas for the limit points and normals can be derived from the eigenstructure.

Reif [27] points out that these analyses do not give adequate conditions to sufficiently guarantee smoothness and to rule out self-intersections of the surface at extraordinary vertices. He presents a technique for proving that the limit surface has a regular G^1 parameterization near the extraordinary point. His analysis pertains to subdivision schemes such as those that generalize B-splines; that is, the surface must be represented by analyzable basis functions away from extraordinary points.

Reif constructs a *characteristic map* that utilizes the underlying basis functions as well as the eigenstructure of the subdivision map. He proves that the regularity of the characteristic map along with conditions on the eigenstructure are sufficient for a well-defined tangent plane. He also shows that the injectivity of the characteristic map is required to ensure that the surface is non-intersecting at the extraordinary point.

The above treatments apply to non-uniform subdivision schemes; that is, they do not require using the same rule throughout the mesh. However, they make the assumption that the subdivision rules are symmetric about each vertex, face, or edge. For example, in Loop's scheme, all the neighbor vertices receive the same weight in the rule that determines the new vertex value. This symmetry makes sense for vertices in the interior of a mesh where a uniformly smooth surface is desired. However, a scheme that proposes to treat boundaries or introduce irregularities in curvature or smoothness must allow more asymmetry in its rules. The piecewise smooth scheme of Hoppe et al. [13] does just that in order to introduce sharp features. We note that this lack of symmetry diminishes the applicability of Fourier transforms, as used by Ball and Storry, to the analysis of the piecewise smooth scheme. The more general eigenanalysis as presented by Halstead et al. and the characteristic maps of Reif are more suited to extension of that scheme.

All of the schemes mentioned so far result in surfaces that approximate their control mesh, but the techniques for analyzing them can also be applied to subdivision schemes that interpolate the vertices in a control mesh. For example, the interpolatory subdivision surface of Dyn, Levin, and Gregory's Butterfly scheme [9] has been found to achieve tangent plane smoothness at vertices of valences between 4 and 7, but is not G^1 at a valence 3 vertex.

Besides determining conditions for the convergence and tangent plane smoothness in these schemes, the issue of how to determine and set free parameters to achieve a fair surface has also been investigated. Catmull and Clark and Loop make empirical decisions. Doo and Sabin suggest optimal values for subdivision weighting, and Ball and Storry relate these parameters to eigenvalues.

The question of curvature continuity has most recently been investigated by Reif [26] who gives a lower bound on the degree of underlying basis functions in a scheme like those based on B-splines, and by Prautzsch [23] who gives conditions on the eigenstructure of such a scheme. Their work gives clues to designing schemes for higher order continuity and explains why attempts to design schemes that achieve G^2

smoothness at extraordinary points by generalizing lower order B-splines have been unsuccessful.

1.1.2 Applications of Subdivision Surfaces

Subdivision surfaces will be better understood and see wider application with improved analysis and more useful tools for creating and manipulating them. Some areas where subdivision surfaces are being used with success include:

- Multiresolution analysis

  Because subdivision surfaces are based on a refinement scheme, they are a natural representation for applications that view an object at various levels of detail. Subdivision schemes are particularly valuable for this application because refinement can be applied adaptively. Lounsbery et al. [19] show that multiresolution analysis can be applied to any uniformly convergent subdivision scheme.

- Surface fitting

  Nasri [21] gives a construction for adjusting a control mesh so that the Doo-Sabin [8] subdivision scheme will produce a surface that interpolates the original control mesh. He gives particular attention to a control mesh with a boundary and assures that the resulting surface boundary interpolates quadratic B-splines. In [22], Nasri extends this work to generate surfaces that interpolate normal vectors.

  Halstead, Kass, and DeRose [12] describe a method for constructing a smooth subdivision surface that uses the Catmull-Clark scheme and interpolates input data points and normals. To achieve a fair surface, they minimize an energy function that seeks to penalize sharp variations in curvature. The free variables are the positions of vertices in the control mesh and are subject to the interpolation constraints. To express these constraints, they utilize formulas that give exact values for the points and tangents on the limit surface that correspond to the control vertices. They give a method for evaluating an energy norm (thin plate, for example) over regions of the surface that derive from an extraordinary point in the mesh. This method involves the eigenstructure of the local subdivision matrix.

- Reconstruction of objects from samples of 3D data

  Hoppe et al.'s work [13, 14, 15, 16] provides for the accurate reconstruction of an object from a dense, uniform sampling of 3D data points on its surface. The resulting piecewise smooth surface is represented using a subdivision scheme that generalizes Loop's. Their algorithm automatically determines the topology of the surface and the placement of sharp features such as boundaries, creases, and corners.

  After creating a dense initial mesh that approximates the input data and captures its topology, the mesh is optimized to reduce the number of faces while simultaneously improving the fit to the input data of the piecewise smooth subdivision surface that it defines. This global optimization balances a least squares fitting component with a function that measures mesh size.

  This algorithm relies on having a dense and uniformly sampled data set so that fairness of the surface is entirely determined by the object being measured.

1.2 Contributions of this Dissertation

This dissertation consists of two parts: one theoretical and one practical. In the first part, we analyze a set of piecewise smooth subdivision rules. In the second, we apply the subdivision scheme to the reconstruction of objects from non-uniformly sampled data. The following sections summarize our research contributions from these two activities.

1.2.1 Analysis

We discuss a theoretical basis for designing subdivision rules for various sharp surface features. We present an analysis to determine properties of the limit surface for the rules that define a piecewise smooth subdivision surface. We show that these surfaces are well-defined, have a well-defined tangent plane where expected, and behave as expected at singular points.

The subdivision rules that we analyze do not exhibit the full symmetry of those that have been previously analyzed. Therefore, our analysis provides a more general framework for examining a subdivision scheme. We accomplish the following specific tasks for the piecewise smooth subdivision scheme described in Hoppe et al. [13].

- We give sufficient conditions for a possible asymmetrical subdivision scheme to converge to a well-defined surface with a regular G^1 parameterization. This includes conditions on the eigenstructure of the local subdivision matrices along with a criterion on the characteristic maps that we adapt from Reif's analysis [27]. We also discuss necessary conditions on the eigenstructure for certain types of singularities.

- We present an eigenanlysis of local subdivision matrices for the piecewise smooth subdivision rules that include sharp features, and from it we derive formulas for points and tangents on the limit surface.

- We construct characteristic maps for piecewise smooth subdivision rules. To verify that the scheme is piecewise G^1, we present algorithms for testing the regularity and injectivity of these characteristic maps.

1.2.2 Application

We investigate a practical application of subdivision surfaces, namely, the reconstruction of an object from 3D data that is sampled non-uniformly. In this context, we suggest a strategy for economically sampling an object of arbitrary topological type and offer a procedure for its reconstruction. In this work, we make the following specific contributions:

- We describe a solution to the global problem of creating a surface of arbitrary topological type that interpolates a network of curves and fits a collection of data points interior to the regions bounded by the curves.

- As a sub-problem, we present an algorithm for creating a fair smooth surface of planar topology that interpolates a given boundary and fits a set of non-uniformly placed interior data points.

- In preparation for the surface construction, we present an algorithm for creating a subdivision curve that fits uniformly sampled data.

- We describe an efficient mesh construction to fill a closed region bounded by a set of connected subdivision curves.

1.3 Outline of Dissertation

Chapter 2 introduces subdivision surfaces and supplies background in the techniques we use for analyzing the subdivision schemes. In particular, we present Loop's scheme and describe its relationship to B-spline surfaces. We also describe the concept of eigenanalysis of local subdivision matrices and introduce characteristic maps.

Chapter 3 presents an analysis of the subdivision schemes for piecewise smooth surfaces. We present eigenanalysis of the subdivision rules and derive limit properties of the surface. We construct the characteristic maps for the subdivision rules and describe algorithms for their analysis.

Chapter 4 describes an algorithm for creating a subdivision surface that reconstructs an object from non-uniformly sampled 3D data.

Chapter 5 is a summary.

Chapter 2

BACKGROUND

In this chapter, we provide a brief introduction to subdivision surfaces and give background for the techniques we use to analyze them.

In Section 2.1, we introduce subdivision surfaces and define some terms. We present the rules for Loop's scheme as the foundation for the piecewise smooth subdivision schemes that we analyze and apply. In Section 2.2, we discuss the relationship of B-spline curves and surfaces to the piecewise smooth subdivision scheme and how B-splines provide basis functions for the subdivision curves and surfaces. In Section 2.3, we provide background for analyzing subdivision surfaces by introducing the techniques of eigenanalysis and characteristic maps.

2.1 Introduction to Subdivision Surfaces

A *subdivision* surface results from repeatedly refining a 3D *control mesh*. Figure 2.1 shows an initial control mesh for a piecewise smooth subdivision surface, and Figure 2.2 shows the mesh after one refinement step. In the limit, a piecewise smooth surface results as shown in Figure 2.3.

The topology of the limit surface is determined by the topology of the initial control mesh. A control mesh for the piecewise smooth scheme has only triangular faces, but the number of edges incident to a vertex is arbitrary. This means that the piecewise smooth scheme can represent objects of arbitrary topological type. We also permit the control mesh to have a boundary, as will be discussed in Chapter 3.

In general, the limit surface cannot be represented by a closed form expression. How-

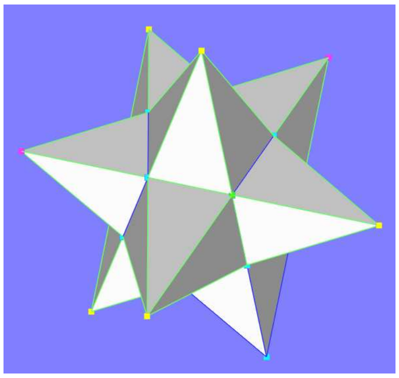

Figure 2.1: An initial control mesh for a piecewise smooth subdivision surface.

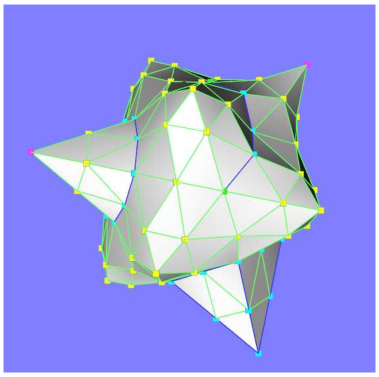

Figure 2.2: The refined control mesh for a piecewise smooth subdivision surface.

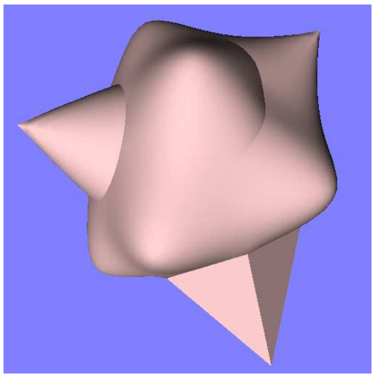

Figure 2.3: The limit surface of the piecewise smooth subdivision scheme.

ever, by restricting the topology of the mesh, we can obtain subdivision surfaces that can be represented as piecewise polynomials. For example, tensor product B-splines can be defined by subdividing a rectangular mesh where every vertex has exactly 4 incident edges. Box-splines are a related representation that use a triangular mesh where every vertex has exactly 6 incident edges.

These restricted meshes are called *regular* because of their symmetry, and subdivision schemes applied to them, such as the B-spline representations, can only model objects of planar topology. In order to model objects of non-planar topology, a more flexible arrangement is used that permits the number of edges per vertex to vary. The number of edges incident to a vertex is called its *valence*. A vertex whose valence adheres to the regular restriction is called *ordinary*; otherwise it is called *extraordinary*.

Loop's scheme (and the piecewise smooth scheme that generalizes it) accommodates extraordinary vertices in a triangular mesh. In fact, these surfaces generalize G^2 quartic Box-splines [13, 17]. Some other schemes that accommodate extraordinary vertices are Doo-Sabin [8], which generalizes biquadratic B-spline surfaces and Catmull-Clark [4], which generalizes bicubic B-splines. The advantage of these schemes is that in regions away from the extraordinary vertices, the surfaces they generate are B-splines, which can be represented as piecewise polynomials in those regions. We will discuss this in more detail in the next section.

We note that all of the schemes mentioned so far create surfaces that approximate the control mesh; that is, the surface does not pass through the control points. However, there are also subdivision schemes that create surfaces that interpolate the points of the mesh. The butterfly scheme of Dyn, Levin, and Gregory [9] is an example of an interpolating subdivision scheme that models objects of arbitrary topological type.

It is useful to define a neighborhood of a vertex in a mesh. The *1-neighborhood* of a vertex is the part of the mesh consisting of the vertex and its nearest neighbor vertices. We can obtain a *2-neighborhood* from a 1-neighborhood by annexing all the 1-neighborhoods of the neighbor vertices. We can use this iteration to define a *k-neighborhood* of a vertex. Figure 2.4 shows a 1-neighborhood and a 2-neighborhood

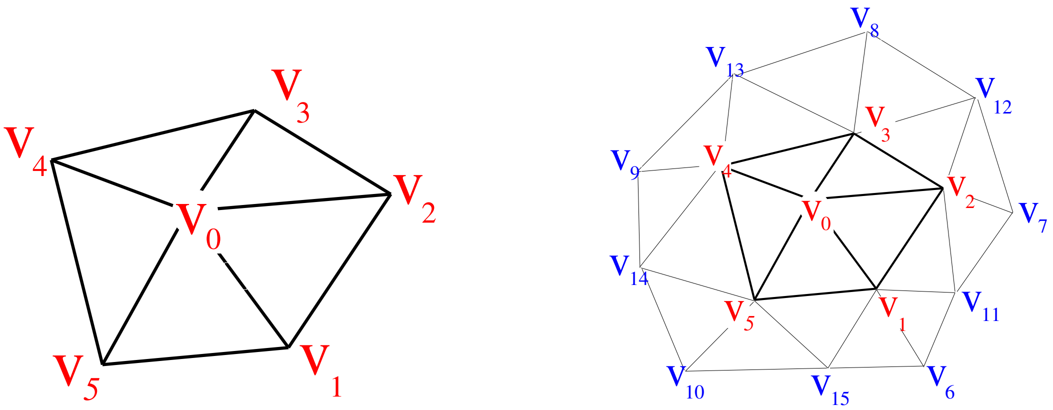

Figure 2.4: 1- and 2-neighborhoods for a valence 5 vertex V_0.

for a vertex of valence 5.

Mesh refinement can be described topologically as splitting each triangular face into 4 faces. Thus, a vertex in the refined mesh corresponds to either a vertex or an edge in the original mesh. It is important to note that the new "edge" vertices will be ordinary; that is, they will have valence 6. This means that repeated subdivision of the mesh has the effect of isolating the extraordinary points because no new ones are introduced.

The vertex positions in the refined mesh are computed according to a set of *subdivision rules*. We can describe the rules by showing how a k-neighborhood in the refined mesh is computed from a k-neighborhood of the corresponding vertex in the original mesh, for some k. The subdivision rules in the piecewise smooth schemes can all be expressed using 1-neighborhoods. For example, let vertex V_0 have neighbors $V_1, V_2, \ldots, V_n$, all in $\mathbb{R}^3$. In the refined mesh, V_0 becomes V_0' and its immediate neighbors will be $V_1', V_2', \ldots, V_n'$, as shown in Figure 2.5 for $n = 5$.

A *vertex rule* tells how to compute the coordinates of vertex V_0' from the coordinates of V_0 and its neighbors V_i. Typically, it is an affine combination. An *edge rule* tells how to compute the coordinates for the new "edge" vertices V_i'.

The rules for the piecewise smooth scheme are given and analyzed in Chapter 3.

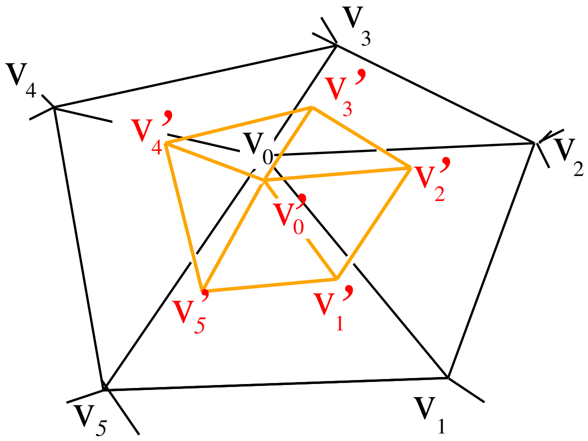

Figure 2.5: Subdivision of a neighborhood of V_0.

That scheme incorporates Loop's rules to create the smooth regions of the surface. We present Loop's rules here to introduce the the concepts and terminology.

V_0' is calculated as an affine combination of positions of the vertices $V_0, V_1, \ldots, V_n$:

$$V_0' = (aV_0 + bV_1 + \cdots + bV_n)/8,$$

where $a = 8 - nb$ and $b = (1/n)(5 - (3 + 2\cos(\frac{2\pi}{n}))^2/8)$.

A new neighbor V_i' is calculated as an affine combination of the end vertices of its corresponding edge V_0, V_i and the edge's nearest neighbors V_{i-1}, V_{i+1}:

$$V_i' = (3V_0 + V_{i-1} + 3V_i + V_{i+1})/8,$$

for $i = 1, \ldots, n$.

It is convenient to represent these rules in diagrams called *masks*. They show the relative weights in the affine combinations for the vertex or edge rule. Figure 2.6 shows the vertex and edge masks for Loop's scheme, where we have $\alpha(n) = a/b$.

Loop's subdivision surfaces are defined as the limit of repeated application of these rules. We call the result the *limit surface*. There is not, in general, a closed form expression for the surface; we cannot, for example, represent all points on the surface

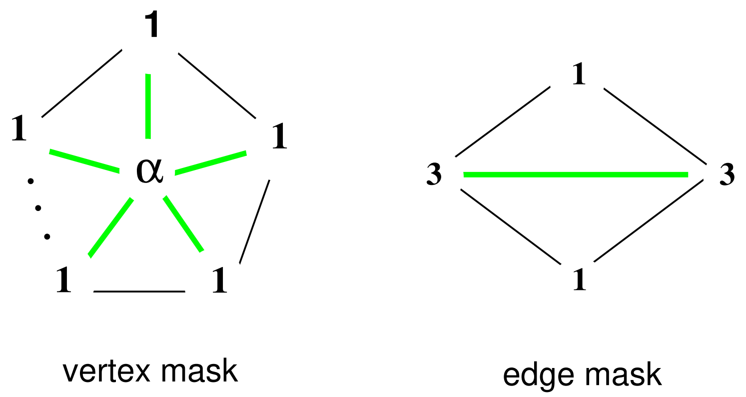

Figure 2.6: Subdivision masks for Loop's vertex and edge rules.

as a polynomial or rational polynomial function. However, we do have exact expressions for computing limit points and limit tangent vectors for points in the control mesh.

In Chapter 3, we will show that the limit position of vertex V_0 is given by

$$V_0^\infty = (wV_0 + V_1 + \cdots + V_n)/(w+n),$$

where $w = 3/b$ and b is as above.

We also show that a well-defined tangent plane at V_0 on the limit surface will be spanned by two vectors that can be written as

$$\vec{v}_1 = \sum_{i=1}^{n} \cos(\frac{2\pi(i-1)}{n})V_i \quad \text{and} \quad \vec{v}_2 = \sum_{i=1}^{n} \sin(\frac{2\pi(i-1)}{n})V_i.$$

2.2 B-spline Representations and Subdivision Schemes

In this section, we discuss the connection of B-spline representations to subdivision schemes, and we show how they play a fundamental role in constructing a basis function for a subdivision curve or surface. This will prove key to the analysis of the piecewise smooth subdivision surface that we discuss in Chapter 3. It can also useful in applications that use them.

B-spline representations, like subdivision curves and surfaces, are defined using a control mesh, but the connection goes even deeper. B-spline curves and surfaces are invariant under refinement of their control net, meaning there is a subdivision scheme that can be used to generate them. In the case of quartic Box-splines, that scheme is Loop's when applied to a regular triangular mesh. For this reason, we can say that Loop's scheme generalizes quartic Box-splines. The advantage of this observation is that away from extraordinary points, there is a piecewise polynomial function that can be used to represent and study the surface. In this section, we provide some more detail about the connection between B-spline curves and Box-spline surfaces and the subdivision schemes that we analyze and apply.

2.2.1 B-splines and Subdivision Curves

We will show how uniform cubic B-spline curves are related to subdivision curves. We illustrate the connection by using a particular type of subdivision curve that is a restricted case of the "crease curves" generated in Hoppe et al. [13] for creases. We use these particular curves in Chapter 4 as the boundary curves of the surfaces created by the algorithm presented there.

B-spline Curves

It has long been known that B-spline curves can be created using subdivision of a control polygon [3, 7, 10]. At each refinement step for a uniform cubic B-spline, new knots are inserted evenly into a uniform knot sequence. The corresponding new vertices are the midpoints of the segments between control points, and new values for the original control points are computed from the old value and those of its two neighbors by the combination:

$$V_i' = V_{i-1} + 6V_i + V_{i+1},$$

which can be expressed by the mask, 1-6-1. (We ignore the end points for now.) Vertices on the limit curve can be computed from the control points at any stage of

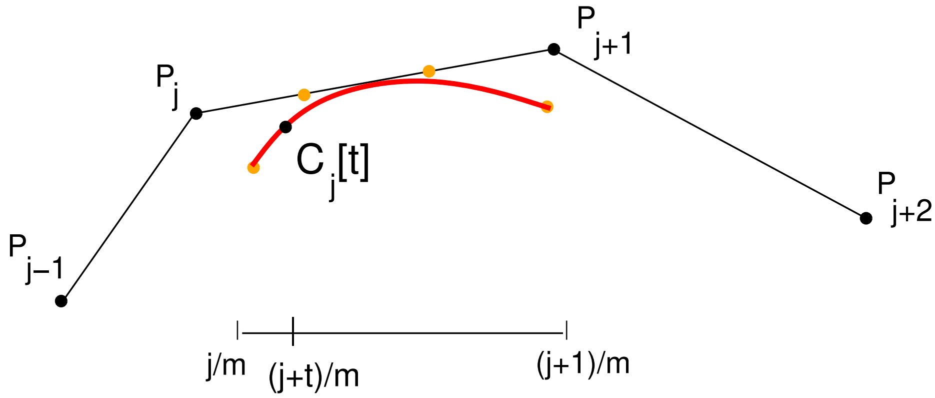

Figure 2.7: A uniform cubic B-spline curve segment $C_j(t)$, $t \in [0, 1]$.

refinement using the mask, 1-4-1.

We will find it useful to have a representation for the basis functions for the uniform cubic B-splines. In the next section, we will show how that representation can be used to compute basis functions for a subdivision curve.

Since a uniform cubic B-spline is a composite of Bézier curve segments, we can represent the curve as a piecewise polynomial as follows. Let the B-spline control points be designated by P_j, for $j = 0, 1, \ldots, m$. Their limit points will correspond to parameter values $u = j/m$ for the entire curve, $u \in [0, 1]$.

A single Bézier segment has a support of 4 consecutive control points, which we write in the array $\mathcal{P}_j = (P_{j-1}, P_j, P_{j+1}, P_{j+2})^T$. A polynomial segment of the B-spline curve can then be represented as

$$C_j(t) = Bseg(t) \cdot \mathcal{P}_j,$$

where $t \in [0, 1]$, and $Bseg(t)$ represents the basis function for that segment. Figure 2.7 illustrates the placement of this curve segment in the control net.

We can compute the basis function $Bseg(t)$, as follows. Let A be the matrix that converts the 4 B-spline control points to the 4 Bézier control points for a curve segment. We have:

$$A = (1/6)\begin{pmatrix} 1 & 4 & 1 & 0 \\ 0 & 4 & 2 & 0 \\ 0 & 2 & 4 & 0 \\ 0 & 1 & 4 & 1 \end{pmatrix}.$$

Let $Bern^{(3)}(t)$ be the array of the degree 3 Bernstein polynomials for $t \in [0,1]$:

$$Bern^{(3)}(t) = ((1-t)^3, 3(1-t)^2 t, 3(1-t)t^2, t^3).$$

Then we can compute the basis function as

$$Bseg(t) = Bern^{(3)}(t) \cdot A.$$

We note that for a uniform B-spline curve, the same basis function $Bseg(t)$ is used for each segment. If, however, we do special handling of the spline end points, the uniformity will be lost. In that case, different basis functions will be needed when the end control points are in the support for the segment. This will be the case for the non-uniform subdivision curves that we discuss in the next section.

Subdivision Curves

In this section, we define a subdivision curve that is a uniform cubic B-spline curve away from its end points. This particular curve will coincide with the boundary curves used for the surfaces in Chapter 4 and is a restricted version of the crease and boundary curves generated by the piecewise smooth subdivision scheme of Hoppe et al. [13]. In Chapter 3, we show that the rules for creating creases in that scheme can be uncoupled to simply create a subdivision curve.

The subdivision mask 1-6-1 is applied to all vertices that are not end points. This is the same mask that is used for the vertices in the B-spline curve. All edges that don't include the end points are subdivided using the mask 1-1. In this scheme, the end points are fixed. The edge connected to the end point is split using the mask

3-5 where 3 is applied to the end point. We call this mask the *special edge rule.* It is important to note that subdivision of the curve isolates the effects of the end control point and the special edge rule that is applied at the end interval.

In the last section, we described the B-spline curve as being composed of a finite sequence of cubic Bézier curve segments. We can describe this subdivision curve in a similar fashion, but with a non-Bézier segment at each end. Now if we subdivide the parts of the control polygon that control the end segments, we gain Bézier segments along with smaller non-Bézier end segments. We can repeat this indefinitely and conceive of the subdivision curve as being composed of infinite sequences of cubic Bézier segments that converge at the end points. Figure 2.8 illustrates subdividing the control polygon to evaluate the subdivision curve as a B-spline near an end point.

This viewpoint is the foundation for our method of calculating basis functions for this subdivision curve, which we denote as $Curve(u)$. We again assume that the control points P_j have parameter values j/m, for $j = 0, 1, \ldots, m$,

Clearly, we can use the B-spline basis functions $Bseg(t)$ to represent these subdivision curves on intervals that do not include an end point.

Thus, for $j/m \leq u \leq (j+1)/m$, the subdivision curve is

$$Curve(u) = Bseg(mu - j) \cdot \mathcal{P}_j,$$

where $\mathcal{P}_j = (P_{j-1}, P_j, P_{j+1}, P_{j+2})^T$ includes no end control point.

However, we can also define basis functions for these curves near an end point. The key idea is that, for any given parameter value u, we perform repeated local subdivision of the control polygon until u falls in an interval supported by a new control polygon that includes no end control point. Then, on that sub-interval, the basis functions $Bseg(t)$ can be applied to the new control points for that sub-interval.

For $(1/m)(1/2)^k \leq u \leq (3/2)(1/m)(1/2)^k$, the subdivision curve is

$$Curve(u) = Bseg(2^{k+1}mu - 2) \cdot \mathcal{P}^k,$$

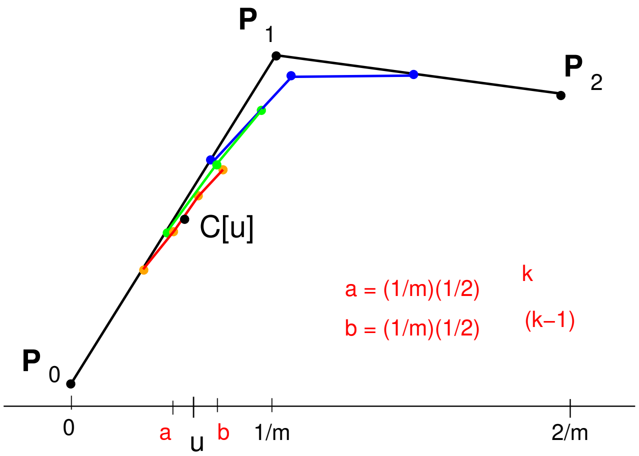

Figure 2.8: Subdividing to obtain an interval away from the left end point to have a B-spline segment on a subdivision curve.

where $\mathcal{P}^k = (P_0^k, P_1^k, P_2^k, P_3^k)$ are control points after k subdivisions. If $m+1$ is the original number of control points, then P_1^k has parameter value $(1/m)(1/2)^k$.

Since the end points of this subdivision curve are fixed, we have values for Curve(0) and Curve(1). Thus, we have shown that $Curve(u)$ can be evaluated for every real $u \in [0,1]$. Further analysis of the subdivision scheme would be needed to tell whether $Curve(u)$ is continuous and differentiable at $u = 0$ and $u = 1$.

The method we presented here for evaluating $Curve(u)$ using the B-spline basis functions can be a useful model in the analysis of the scheme. A similar model applies to surfaces.

2.2.2 Box-splines and Subdivision Surfaces

We discuss the relationship between quartic Box-splines and the subdivision surfaces generated by Loop's scheme. This is relevant to the piecewise smooth scheme of Hoppe et al. because it incorporates Loop's scheme for its smooth regions. We will use this relationship in Chapters 3 and 4 when we compute properties of the piecewise smooth subdivision surfaces.

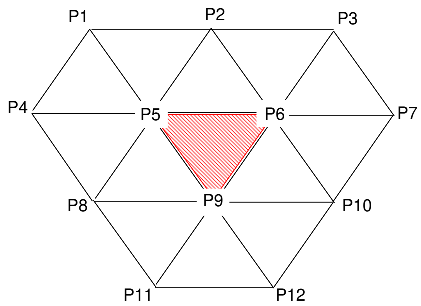

Figure 2.9: The control net for a quartic Box-spline patch.

Box-splines

Analogous to B-spline curves, Box-spline surfaces can be generated using subdivision of a control net [17]. The refinement masks for uniform quartic Box-splines are as in Figure 2.6, taking $n = 6$ and obtaining $\alpha(n) = 10$.

A quartic Box-spline surface is composed of triangular quartic Bézier patches. Each patch can be represented as

$$Patch(u,v,w) = Bern^{(4)}(u,v,w) \cdot Q \cdot \mathcal{P}.$$

$Bern^{(4)}(u,v,w)$ is an array of the 15 Bernstein monomials of degree 4:

$$\begin{aligned} Bern^{(4)}(u,v,w) = (&u^4, 4u^3v, 4u^3w, 6u^2v^2, 12u^2vw, 6u^2w^2, 4uv^3, \\ &12uv^2w, 12uvw^2, 4uw^3, v^4, 4v^3, 6v^2w^2, 4vw^3, w^4). \end{aligned}$$

$\mathcal{P}$ is an array of the 12 control points for the patch, arranged as shown in Figure 2.9.

The matrix Q converts these 12 Box-spline control points to the 15 control points for a quartic Bézier patch:

$$Q = (1/24)\begin{pmatrix} 2 & 2 & 0 & 2 & 12 & 2 & 0 & 2 & 2 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 12 & 3 & 0 & 3 & 4 & 0 & 0 & 0 \\ 1 & 3 & 0 & 0 & 12 & 4 & 0 & 1 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 8 & 4 & 0 & 4 & 8 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 10 & 6 & 0 & 1 & 6 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 & 8 & 8 & 0 & 0 & 4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 4 & 3 & 0 & 3 & 12 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 6 & 6 & 0 & 1 & 10 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 6 & 10 & 0 & 0 & 6 & 1 & 0 & 0 \\ 0 & 3 & 1 & 0 & 4 & 12 & 0 & 0 & 3 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 2 & 0 & 2 & 12 & 2 & 2 & 2 \\ 0 & 0 & 0 & 0 & 3 & 4 & 0 & 1 & 12 & 3 & 0 & 1 \\ 0 & 0 & 0 & 0 & 4 & 8 & 0 & 0 & 8 & 4 & 0 & 0 \\ 0 & 1 & 0 & 0 & 3 & 12 & 1 & 0 & 4 & 3 & 0 & 0 \\ 0 & 2 & 2 & 0 & 2 & 12 & 2 & 0 & 2 & 2 & 0 & 0 \end{pmatrix}.$$

Loop's Subdivision Surfaces

As mentioned earlier, Loop's subdivision yields a quartic Box-spline surface on a regular mesh because for valence 6 vertices the same refinement masks are used. Thus, away from extraordinary points, Loop's subdivision surface has the Box-spline basis for patches that derive from regions determined by a regular control net as shown in Figure 2.9.

When the control net contains an extraordinary vertex in its interior, as shown in Figure 2.10, we can still find a value for a basis function at a point arbitrarily close to that vertex. Analogous to our treatment of subdivision curves, we subdivide until the point is in a face with a regular control net. The control net shown in Figure 2.11 has been subdivided once to show 3 sub-patches with regular control nets and one (shown solid color) with a valence 5 vertex in its control net.

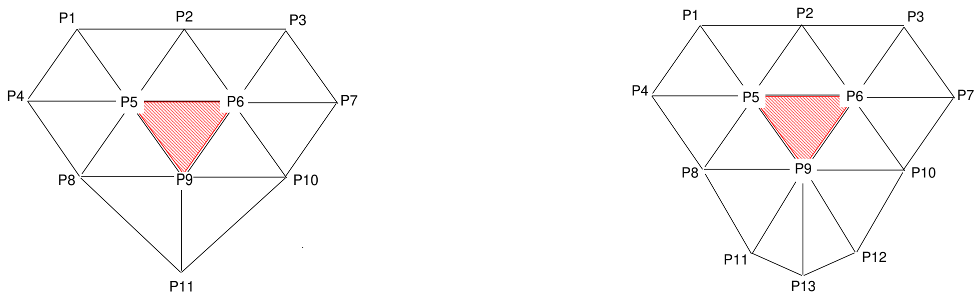

Figure 2.10: Control nets for patches with a valence 5 and 7 vertex, respectively.

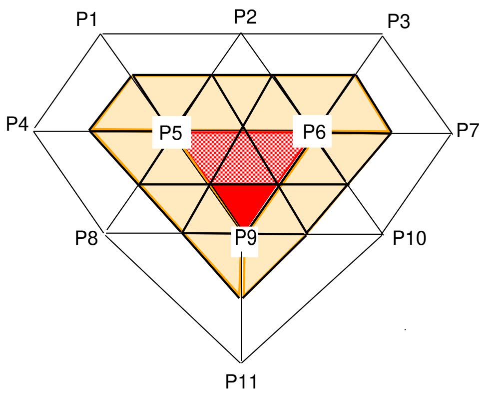

Figure 2.11: The control net for a patch with a valence 5 vertex. After one subdivision there are 3 Box-spline sub-patches. (Shown cross-hatched.)

2.3 *Analyzing the Limit Surface*

For a particular scheme, we wish to prove that the repeated application of the subdivision rules results in a well-defined surface with the properties that we expect. For the piecewise smooth scheme, for example, we need to establish convergence, non-self-intersection, and G^1 smoothness (well-defined tangents) where not designated as sharp. At sharp features, we wish to verify the type of singularity.

At ordinary points, a Loop surface is a G^2 quartic Box-spline and, as such, can be analyzed as a piecewise polynomial function. At extraordinary points, special techniques are needed because we don't have a closed mathematical form for the surface.

In this section, we provide some background for the techniques we employ to analyze the limit surface at extraordinary points. We encode the subdivision rules in matrices and show how their eigenstructure provides information about the limit surface. We also introduce the technique of *characteristic maps* developed by Urlich Reif [27] for proving smoothness of a subdivision surface at extraordinary points.

2.3.1 Local Subdivision Matrices

The subdivision process can be represented using matrix operations because the subdivision rules are linear combinations of the mesh vertices. Let M^0 denote the array $(V_0, V_1, \dots V_n)^T$ of vertices constituting some mesh neighborhood of vertex V_0. We can use an $n{+}1$ by $n{+}1$ matrix S_n to encode the rules for computing the neighborhood M^1 of V_0' in the subdivided mesh.

$$\begin{pmatrix} V_0' \\ V_1' \\ \vdots \\ V_n' \end{pmatrix} = S_n \begin{pmatrix} V_0 \\ V_1 \\ \vdots \\ V_n \end{pmatrix},$$

or, more concisely,

$$M^1 = S_n \cdot M^0.$$

The subsequent neighborhoods M^k, $k = 1, 2, \ldots$, can be derived by

$$M^k = S_n \cdot M^{k-1} = S_n^k M^0.$$

For example, the local subdivision matrix for a 1-neighborhood of a vertex of valence 5 in Loop's scheme can be written as

$$S_5 = (1/8)\begin{pmatrix} a & b & b & b & b & b \\ 3 & 3 & 1 & 0 & 0 & 1 \\ 3 & 1 & 3 & 1 & 0 & 0 \\ 3 & 0 & 1 & 3 & 1 & 0 \\ 3 & 0 & 0 & 1 & 3 & 1 \\ 3 & 1 & 0 & 0 & 1 & 3 \end{pmatrix},$$

where $a = 8 - 5b$ and $b = (1/5)(5 - (3 + 2\cos(\frac{2\pi}{5}))^2/8)$.

2.3.2 Convergence of the Subdivision Scheme

If the subdivision scheme is well-defined, these neighborhoods will contract to a single point in the limit. Thus as $k \to \infty$, each point V_i^k in M^k will approach the limit point V_0^∞.

How do we know whether a given subdivision scheme does in fact converge to a well-defined surface at each vertex? How can we compute the coordinates of the limit points if they exist?

Eigenanalysis of the local subdivision matrix provides sufficient conditions for convergence and yields a formula for the limit point if it exists. To simplify our analysis, we require that the matrix not be defective, that is, that there exist $n + 1$ linearly independent eigenvectors. Sufficient conditions for convergence are:

- There must a single largest eigenvalue, $\lambda_0 = 1$.
- Its corresponding right eigenvector must be $R_0 = (1, 1, \ldots, 1)$.

It can be shown [8, 12] that the limit point is

$$V_0^\infty = \sum_{i=0}^{n} l_i V_i,$$

where $L_0 = (l_0, l_1, \ldots, l_n)$ is the dominant left eigenvector, normalized so that $l_0 + l_1 + \ldots + l_n = 1$. (Also see Davis [6] for a general discussion of limits of matrix products.)

We can check these conditions on Loop's scheme. An eigenanalysis of this scheme is provided in Section 3.3. The local subdivision matrix for a vertex of valence n in Loop's scheme can be written as

$$S_n = (1/8)\begin{pmatrix} a & b & b & b & \cdots & \cdots & b & b \\ 3 & 3 & 1 & 0 & \cdots & \cdots & 0 & 1 \\ 3 & 1 & 3 & 1 & \cdots & \cdots & 0 & 0 \\ 3 & 0 & 1 & 3 & 1 & 0 & \cdots & 0 \\ . & . & . & . & . & . & . & . \\ 3 & 0 & \cdots & \cdots & 0 & 1 & 3 & 1 \\ 3 & 1 & 0 & \cdots & \cdots & 0 & 1 & 3 \end{pmatrix},$$

where $a = 8 - nb$ and $b = (1/n)(5 - (3 + 2\cos(\frac{2\pi}{n}))^2/8)$.

This $n+1$ by $n+1$ matrix is non-defective and fulfills the sufficient conditions for convergence as follows. Because the rows of S_n sum to 1, $\lambda_0 = 1$ is an eigenvalue, and its corresponding right eigenvector is $(1, 1, \ldots, 1)$. In fact, λ_0 is the single dominant eigenvalue. The corresponding left eigenvector is $(w, 1, 1, \ldots, 1)/(w+n)$ where $w = 3/b$. We can conclude that the limit vertex is given by

$$V_0^\infty = (wV_0 + V_1 + \cdots + V_n)/(w+n).$$

2.3.3 Sufficient Conditions for a Well-Defined Tangent Plane

How can we know if a subdivision scheme produces a surface that has a well-defined tangent plane at the limit point of an extraordinary point? The answer can be obtained in part from analyzing the eigenstructure of the local subdivision matrix S_n. It has been suggested [8, 12] that conditions on the sub-dominant eigenvalues are sufficient for this, but Reif [27] gives evidence that a stronger criterion is needed. His criterion is that the characteristic map for the subdivision rule be regular, that is, have a non-vanishing Jacobian. We will defer discussion of that map for now and discuss the condition on the eigenstructure.

Let $\lambda_0, \lambda_1, \ldots, \lambda_n$ be the eigenvalues of S_n in non-increasing order. If we assume regularity of the characteristic map for the subdivision rule, then the subdivision rule will converge to a well-defined tangent plane if

$$\lambda_0 = 1 > \lambda_1 \geq \lambda_2 > \lambda_3.$$

For Loop's scheme, we find the 2 sub-dominant eigenvalues are

$$\lambda_1 = \lambda_2 = (3 + 2\cos(2\pi/n))/8.$$

Since $\lambda_3 = (3 + 2\cos(4\pi/n))/8$ is smaller than λ_1 and λ_2, the condition is satisfied.

The schemes of Doo-Sabin, Catmull-Clark, and Loop all treat the edges about a vertex with the same rule. For these symmetric schemes, the sub-dominant eigenvalues λ_1 and λ_2 are equal. In his analysis [27], Reif states their equality as a requirement, but his arguments can be extended to include the case where $\lambda_1 > \lambda_2 > \lambda_3$. In Chapter 3 we use this fact as we analyze the asymmetric schemes described in Hoppe et al. [13] for creating creases and corners.

2.3.4 Formulas for Tangent Vectors

Formulas for tangent vectors at V_0^∞ are found by using the left eigenvectors corresponding to the two sub-dominant eigenvalues λ_1, λ_2. If $L_1 = (a_0, a_1, \ldots, a_n)$ and $L_2 = (b_0, b_1, \ldots, b_n)$ are the left eigenvectors, then the tangent vectors are in the span of two vectors and can be represented as

$$\vec{v}_1 = a_0 V_0 + a_1 V_1 + \cdots + a_n V_n,$$

and

$$\vec{v}_2 = b_0 V_0 + b_1 V_1 + \cdots + b_n V_n.$$

The normal is found as $\vec{v}_1 \times \vec{v}_2$.

In Loop's scheme, the sub-dominant eigenvectors can be represented as

$$(0, c_1, c_2, \ldots, c_n) \quad \text{and} \quad (0, s_1, s_2, \ldots, s_n),$$

where $c_i = \cos(\frac{2\pi(i-1)}{n})$ and $s_i = \sin(\frac{2\pi(i-1)}{n})$. Thus, the tangent vectors are given by the formulas we presented in Section 2.1.

2.3.5 Sufficient Conditions for a Regular G^1 Parameterization

For schemes that are generalizations of piecewise polynomial representations, such as the ones we consider, subdivision converges to a smooth surface except at extraordinary vertices. Loop's scheme yields a quartic Box-spline surface when applied to a regular triangular mesh. Since subdivision isolates the extraordinary vertices, the limit surface is G^2 except at extraordinary vertices. When the rules for sharp features are considered, we recall that subdivision gives us only ordinary smooth vertices arbitrarily close to the sharp edges and points. This means that these surfaces will also be G^2 except at these isolated sharp edges and points.

We hope that at the extraordinary vertices and sharp features, the surface will be well formed and G^1. To guarantee this, we must show that the surface has a regular G^1 parameterization at those points. This is the task of the 2D characteristic maps that are defined by Reif [27]. He proves that if the characteristic map of a subdivision rule is regular and injective, then the surface is regular at the vertices where that rule is applied.

Without justification, we can give a constructive definition of the characteristic map for a subdivision rule. We illustrate it using Loop's scheme. The map will be defined

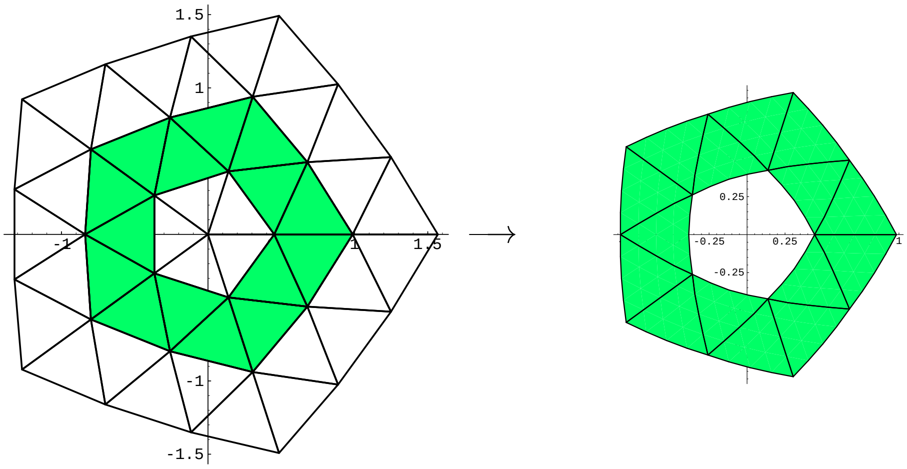

Figure 2.12: Characteristic map for valence 5 extraordinary vertex.

as an annular shaped 2D surface using the Box-spline basis functions on a (topologically) regular 2D control mesh, which we construct as follows.

Let R_1 and R_2 be the right eigenvectors that correspond to the sub-dominant eigenvalues $\lambda_1, \lambda_2 < 1$, of the local subdivision matrix. For this purpose we extend the matrix to cover a 3-neighborhood M.

For each vertex V_i in the neighborhood M, we construct a corresponding 2D control point

$$P_i = (R_1[i], R_2[i])$$

and form the corresponding 2D mesh. The mesh is shown on the left in Figure 2.12 for a valence 5 extraordinary vertex using Loop's scheme. In this example, the extraordinary vertex V_0 corresponds to the point $P_0 = (0, 0)$.

If we apply subdivision to this mesh, we get a 2D surface in the limit. If we take only the portion of this surface that corresponds to the ring of faces in the mesh that map to Box-spline patches, we have an annular shaped surface that excludes a central region containing the limit point of the extraordinary point. The faces and their resulting patches are shown shaded in Figure 2.12.

We can define the characteristic map of the subdivision rule to be this annular shaped surface. The properties of this quartic Box-spline map depend on the arrangement of its control points. The map shown in Figure 2.12 certainly appears regular and injective, so that we would like to conclude that at valence 5 vertices the Loop subdivision scheme would converge to a G^1 surface. Our analytic check on this map's regularity and injectivity agrees with this judgement.

In Chapter 3, we explain how we define, construct, and analyze the characteristic maps for the various rules of the piecewise smooth subdivision scheme.

We close this section with a remark to clarify our use of some terminology. When we say that a scheme such as B-splines is G^1, we mean that it is "generically" G^1. One must be aware that it is possible to position the control points so that a singularity will occur. A familiar example is a cubic Bézier curve segment that has a cusp. However, there are only a set of measure zero such positions for a particular scheme. When the control points are not in this set, we say they are *in general position* and, hence, define a G^1 curve or surface.

If the characteristic map of a subdivision scheme is regular and injective, then its control mesh provides one such control point positioning that is in general position for that scheme.

Chapter 3

ANALYSIS OF PIECEWISE SMOOTH SUBDIVISION SURFACES

In this chapter, we describe our results in analyzing properties of piecewise smooth subdivision surfaces. We use the subdivision scheme described in Hoppe, et al. [13] for creating piecewise smooth surfaces. This scheme is an extension of Loop surfaces. Loop introduced extraordinary points by generalizing the subdivision rules that yield quartic Box-splines. Hoppe, et al. introduced the sharp features: darts, creases, and corners. We present the rules for these schemes with the inclusion of conical and cusp points. We also slightly extend the definitions of darts, creases, and corners.

In the sections that follow, we show how the subdivision rules produce the singularities anticipated for these features. We describe our approach to analyzing smoothness of the surface at extraordinary smooth points, darts, boundaries, creases, and corners. We also derive formulas for limit points and tangent vectors on these surfaces.

The chapter is organized as follows. In Section 3.1, we describe the rules for the piecewise smooth subdivision scheme and summarize the results of our analysis of it. In Section 3.2, we explain how we apply eigenanalysis and characteristic maps to analyze the limit surface. We present the resulting eigenanalysis in Section 3.3 and our analysis of the characteristic maps in Section 3.4.

3.1 *Summary of the Piecewise Smooth Subdivision Rules and Analysis Results*

Our subdivision scheme, an extension of Loop's scheme, is applied to a triangular mesh of arbitrary topological type. In Loop's scheme, the subdivision rules are applied to each vertex and edge with the same intention, that of creating a smooth surface. The piecewise smooth subdivision scheme extends Loop's scheme by adding special rules that are applied to designated vertices and edges to create sharp features. In this section, we define the sharp features, explain how they are specified in the mesh, and present the special subdivision rules that are applied to the mesh to generate the sharp features.

We assumed that the limit surface is G^1 smooth except at sharp features, which we specify as follows. *Creases* are smooth curves along which the surface is G^0 rather than G^1. We specify a crease by a sequence of connected vertices in the surface mesh. *Corners* are vertices that lie on 2 or more creases and must be end points of creases. If the surface has a boundary, it is treated as one or more crease curves, connected by corner vertices. We define a *dart* as a crease that ends in a smooth vertex. Thus, we refer to a smooth vertex incident to one or more creases, as a dart vertex. For now, we omit from our analysis a crease that does not end at a corner or at a smooth vertex. Identifying and analyzing all the possible types of singular features is left for future work.

We also define two types of isolated sharp points where the surface is G^0. These types are differentiated by the behavior of the limiting tangent vectors on curves as they approach the point. The first type is useful for creating a cone-like feature, and hence we call it a *conical point.* The space of limiting tangent vectors at a conical point is of dimension 3. The second type of sharp point gives a cusp-like appearance, and we accordingly call it a *cusp point.* The space of limiting tangent vectors at a cusp point is of dimension one, that is, the tangent space collapses to a line.

In Chapter 2, Figure 2.1 shows an initial control mesh for the piecewise smooth surface shown in Figure 2.3. The vertices and edges are colored to show the type of subdivision rule that is to be applied to them. To vertices colored in yellow and to edges colored in green, the rules for a smooth surface (Loop's rules) are applied. Vertices and edges colored in blue belong to creases. The blue vertex in the lower portion of the picture becomes a corner because is lies on multiple creases. The green vertex at the end of the central crease is a dart and the smooth rule is applied to it. Vertices colored in pink will be isolated sharp points. The one at the left is a conical point, and the one in the upper right resembles a cusp. A data structure appended to the control mesh keeps track of the specific designations for these features.

In the following subsections, we describe the local subdivision rules that are applied to the mesh to create these surface features. For each rule, we describe the properties that the limit surface will possess, and we give formulas for calculating limiting positions and tangent vectors on the surface. Figure 3.1 shows the categories of subdivision masks for the rules for these features.

3.1.1 Smooth and Dart Vertices

The surface is meant to be G^1 at both smooth and dart vertices, and Loop's rules are used for them both. Here we call them the *smooth vertex rule* and the *smooth edge rule.* We introduced these rules in Section 2.1 but repeat them here.

The subdivision rule for a smooth valence n vertex is

$$V_0' = (aV_0 + bV_1 + \cdots + bV_n)/8,$$

where $a = 8 - nb$ and $b = (1/n)(5 - (3 + 2\cos(\frac{2\pi}{n}))^2/8)$.

The smooth edge rule gives the nearest neighbors to V_0' as

$$V_i' = (3V_0 + V_{i-1} + 3V_i + V_{i+1})/8,$$

for $i = 1, \ldots, n$. This rule still applies when V_0 is a dart vertex and V_i is on a crease.

In Sections 3.3.1 and 3.4.1, we discuss our analysis of the limit surface from these rules. We conclude that at least for all practical valences, the surface is well-defined and G^1

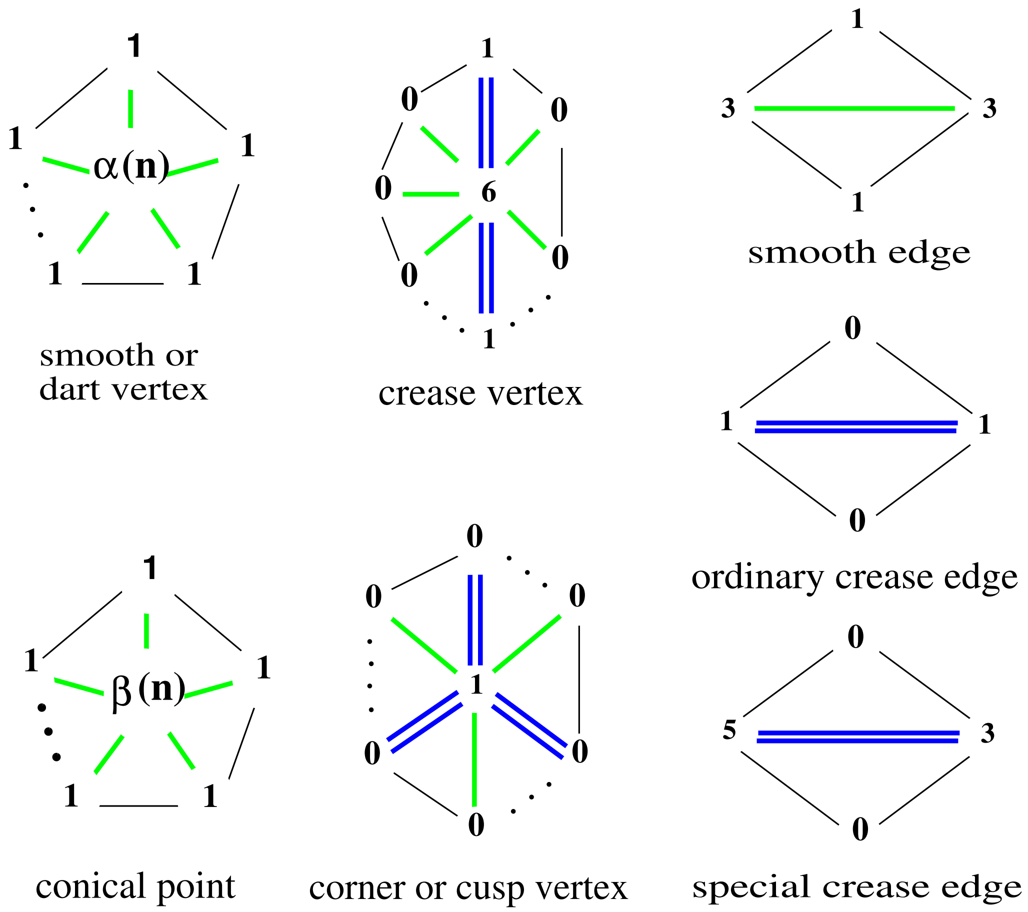

Figure 3.1: Subdivision masks for various feature types.

at smooth extraordinary points. We will derive formulas for the limit positions and tangent vectors from the eigenanalysis of the local subdivision matrix for the smooth rules in Section 3.3.1, but we show the results here.

The limit position of vertex V_0 is given by

$$V_0^{\infty} = (wV_0 + (V_1 + \cdots + V_n))/(w + n),$$

where $w = 3/b$ and b is as above.

The tangent space at V_0 is spanned by the two vectors

$$\vec{v}_1 = \sum_{i=1}^{n} \cos(\frac{2\pi(i-1)}{n})V_i$$

and

$$\vec{v}_2 = \sum_{i=1}^{n} \sin(\frac{2\pi(i-1)}{n})V_i.$$

3.1.2 Crease and Boundary Vertices

We wish our surfaces to behave the same along boundaries as along crease curves. Thus, we employ the same rules at the vertices and edges along a boundary as we do along a crease.

We omit from this discussion dart vertices and edges connecting the dart vertex to the crease because we apply the smooth vertex and edge rules to them as has been discussed.

A crease can be computed separately from the surface as a subdivision curve because the subdivision surface rules only depend on vertices along the crease. A boundary curve may be computed in the same way. These curves are cubic B-splines except at extraordinary crease vertices (see below), such as their end points.

We define two types of vertices along a crease, ordinary and extraordinary. An *ordinary crease vertex* is not a corner (or a dart) and has exactly 2 smooth neighbor vertices on each side of the crease. (In case of a boundary, there is only one side.)

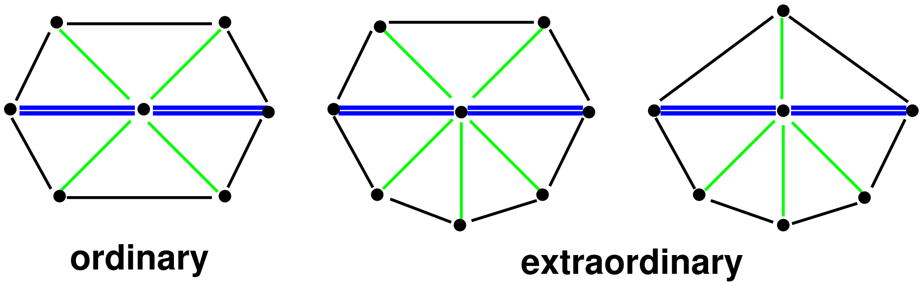

Figure 3.2: Configurations for ordinary and extraordinary crease vertices.

Thus, the total number of neighbors of an ordinary crease vertex is 6 (or 4 if a boundary). Otherwise, the vertex is called an *extraordinary crease vertex*. Figure 3.2 illustrates these two types of crease vertices. Since corner vertices are end points of creases, we include them as extraordinary creases vertices in our discussion unless explicitly excluded.

At an ordinary crease vertex, the surface on each side of the crease is a G^2 quartic Box-spline. (See Section 3.2.1 for a proof.) At an extraordinary crease vertex, we claim that the surface bordering the crease is G^1 for all practical valences. We explain our rationale in Section 3.5.

We will see that we can uncouple the subdivision rules for the surfaces on each side of a crease. This will simplify our analysis, and it allows us to use separate neighborhoods at a crease vertex to represent the two surfaces bordering the crease. It also simplifies our presentation of the subdivision rules. Thus, in this discussion, we will determine the valence n of a crease vertex by counting only neighbor vertices along the crease and on one surface side. If V_0 is the crease vertex, then its neighbors along the crease are V_1 and V_n, and vertices $V_2, \ldots, V_{n-1}$ are smooth vertices away from the crease. Figure 3.3 shows subdivision of a boundary or crease neighborhood of valence 5.

The rule for all non-corner vertices along a crease is given by

$$V_0' = (6V_0 + V_1 + V_n)/8.$$

The new edge crease vertices are calculated by either the *ordinary crease edge rule*

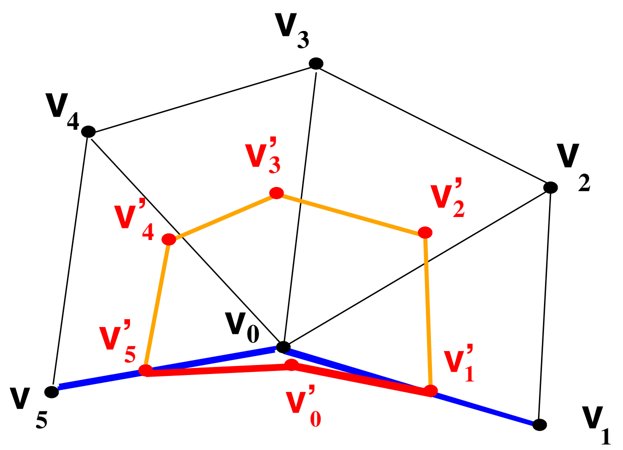

Figure 3.3: Subdivision of a neighborhood of a valence 5 crease vertex V_0.

or by the *special crease edge rule.* The ordinary crease edge rule applies when both vertices are ordinary or both are extraordinary, and is given by

$$V_1' = (V_0 + V_1)/2.$$

The special crease edge rule applies when exactly one of the vertices is extraordinary, and is given by

$$V_1' = (3V_0 + 5V_1)/8.$$

Here V_0 is an extraordinary crease vertex and V_1 is an ordinary crease vertex.

We will see that there is some arbitrariness in selecting this rule, but also that the ordinary rule is not appropriate for these cases. (See Section 3.5.)

In Sections 3.3.2 and 3.4.2, we discuss our analysis of the limit surface from these rules. We conclude that, at least for all practical valences (except for possibly $n = 3$ which is left for future work) the surface is well-defined and G^1 at the (non-corner) extraordinary crease points. We derive formulas for the limit positions and tangent vectors using the eigenanalysis of the local subdivision matrix for the ordinary and extraordinary rules in Section 3.3.2, but we show the results here.

The formulas we give for these limit positions and tangent vectors assume that the central vertex is not adjacent to an extraordinary crease (or corner) vertex or to a dart. The limit formulas for vertices where one or both neighbors are extraordinary can be obtained by first calculating the result of a single subdivision and then applying the following formulas. The masks for other combinations of vertices are shown in Appendix A.

The limit position of a ordinary crease vertex V_0 is

$$V_0^\infty = (4V_0 + V_1 + V_n)/6,$$

where V_1 and V_n are assumed to be ordinary vertices. Note that this is the rule for obtaining the end points of the Bézier curve segments on a B-spline curve.

The limit position of an extraordinary (non-corner) crease vertex V_0 is

$$V_0^\infty = (3V_0 + V_1 + V_n)/5,$$

where V_1 and V_n are assumed to be ordinary vertices.

The tangent space at points along the crease is spanned by vectors $\vec{v}_1$ and $\vec{v}_2$. The vector $\vec{v}_1$ is tangent to the crease curve, and $\vec{v}_2$ is the cross tangent vector. For both ordinary and extraordinary crease vertices V_0 with $n \geq 2$, we have

$$\vec{v}_1 = V_1 - V_n,$$

where V_1 and V_n are both ordinary. Appendix A shows masks for the cases where one or both neighbors are extraordinary.

At a ordinary crease vertex, the valence is 4, and we have

$$\vec{v}_2 = (V_2 + V_3)/2 - (V_1 + 2V_0 + V_4)/4,$$

where V_1 and V_4 are both either ordinary. The masks for the cases where the neighbors may not be ordinary may be calculated by first calculating the neighbors after one subdivision and then applying this formula.

For an extraordinary crease vertex, for $n \geq 4$, we have

$$\vec{v}_2 = \sum_{i=1}^{n} w_i V_i$$

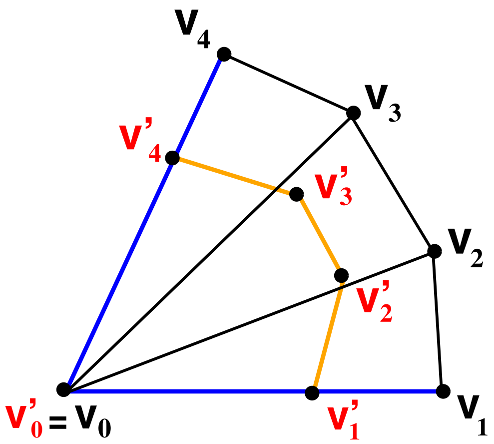

Figure 3.4: Subdivision of a neighborhood of a valence 4 corner vertex V_0.

where $w_1 = w_n = \sin(\frac{\pi}{n-1})/(2\cos(\frac{\pi}{n-1}) - 2)$ and $w_i = \sin(\frac{\pi(i-1)}{n-1})$ for $i = 2, \ldots, n-1$. This formula also assumes that V_1 and V_n are both either ordinary or extraordinary crease vertices. Again, the masks for the cases where the neighbors may not be ordinary may be calculated by first calculating the neighbors after one subdivision and then applying this formula.

If $n = 2$, we have

$$\vec{v}_2 = V_1 + V_2 - 2V_0,$$

where V_1 and V_2 are ordinary crease vertices.

The eigenanalysis will show that the valence case of $n = 3$ requires special attention which we defer to future work.

3.1.3 Corner Vertices

We have defined a corner as the end point of two or more crease curves. Analogous to our discussion of the crease rules, we can separate the analysis of a surface region bordered by two creases at a corner. We can thus describe the rules for subdivision at a corner in terms of a neighborhood for that surface region. We accordingly define the valence of a corner vertex to be the number of edges incident to the corner in this neighborhood. Figure 3.4 shows subdivision for such a corner neighborhood at a vertex of valence 4.

The following rules show the subdivision at a corner of valence n. A corner vertex is fixed; thus, we have

$$V_0' = V_0.$$

The new vertices on the edges adjacent to the corner use the special crease edge rule that is given by

$$V_1' = (3V_0 + 5V_1)/8,$$

where V_0 is the corner vertex and V_1 is an ordinary crease vertex.

However, if V_1 is an extraordinary crease vertex, then the ordinary crease edge rule is applied. It is given by

$$V_1' = (V_0 + V_1)/2.$$

This has the effect of separating the extraordinary vertices with ordinary crease vertices so that with further subdivisions the extraordinary crease vertices will become isolated.

In Sections 3.3.3 and 3.4.3, we discuss our analysis of the limit surface from these rules. We conclude, that at least for all practical valences, the surface is well-defined and G^1 at the (non-corner) extraordinary crease points. We derive formulas for the limit positions and tangent vectors using the eigenanalysis of the local subdivision matrix for the corner rule in Section 3.3.3, but we show the results here.

The limit position of a corner vertex is clearly itself because it never changes, so that we have

$$V_0^\infty = V_0.$$

Thus, the surface interpolates the corner vertices.

The tangent space for a smooth surface region at a corner is spanned by the vectors $\vec{v}_1$ and $\vec{v}_2$ that lie along the crease curves bounding the region. We find that

$$\vec{v}_1 = V_1 - V_0 \quad \text{and} \quad \vec{v}_2 = V_n - V_0.$$

These formulas apply whether or not V_1 and V_n are ordinary crease vertices.

3.1.4 Conical and Cusp points

We define two more sets of vertex subdivision rules to create conical and cusp points. These are symmetric rules like the smooth rule, and the same edge rule is applied to every edge incident to the vertex. In fact, the smooth edge rule is used for all edges incident to a conical or a cusp point.

The vertex rule for a conical point is given by

$$V_0' = (aV_0 + bV_1 + \cdots + bV_n)/8,$$

where $a = 8 - nb$ and $b = (2/n)(1 - \cos(\frac{2\pi}{n}))$.

This rule is a variation of the smooth vertex rule. For this particular choice of a and b, the limiting tangent vectors of curves approaching the conical point do not lie in a single plane. Thus, no tangent plane exists at V_0.

The limit vertex position is given by

$$V_0^{\infty} = (wV_0 + V_1 + \ldots + V_n)/(w + n),$$

where $w = 3/b$ and b is as above.

We can also consider a cusp-like singularity where all approaching tangent vectors converge to a single direction. For this cusp point, we keep the vertex unchanged, so that

$$V_0' = V_0.$$

The limiting tangent vector at the cusp point is given by

$$\vec{v} = V_0 - (V_1 + \ldots + V_n)/n.$$

The ordinary smooth edge rule applies to both of these schemes so that the singularity is isolated at these points.

3.2 Analysis of the Limit Surface

In this section, we describe our approach to analyzing the smoothness properties of a piecewise smooth subdivision surface.

In Section 3.2.1, we define piecewise smoothness in the context of subdivision surfaces. We identify the regions covered by the G^2 basis functions and the types of vertices to which the eigenanalysis and characteristic maps must be applied to prove convergence and smoothness. In Section 3.2.2, we justify our extension of the eigenstructure conditions for smoothness and convergence for the piecewise smooth rules. We also show how the eigenanalysis is used to derive limit points and tangent vectors and to explain singularities.

3.2.1 Proving Piecewise Smoothness

In this section, we define piecewise smoothness in the context of subdivision surfaces. We have defined a crease as a G^1 curve that creates a G^0 discontinuity in the surface. Thus, a crease cuts the surface into two G^1 smooth regions. Boundary curves are defined using the same subdivision rules as creases. We will justify the claim that the piecewise smooth rules define a surface that is G^1 up to the crease (or boundary). Corners are where 2 or more crease (or boundary) curves meet. We claim that each surface region bounded by the creases is G^1 up to the corner.

We will see that the subdivision rules for vertices and edges along creases and at corners can be uncoupled to separately define the subdivision surfaces of the regions that are separated by the crease curves. See Figure 3.5. This means that our analysis of the surface along a crease or at a corner need only consider one of these regions at a time.

In Chapter 2, we stated that G^2 smoothness away from the extraordinary vertices in Loop's scheme is guaranteed by the properties of the C^2 Box-splines, which are piecewise quartic polynomials. This is because subdivision isolates the extraordinary

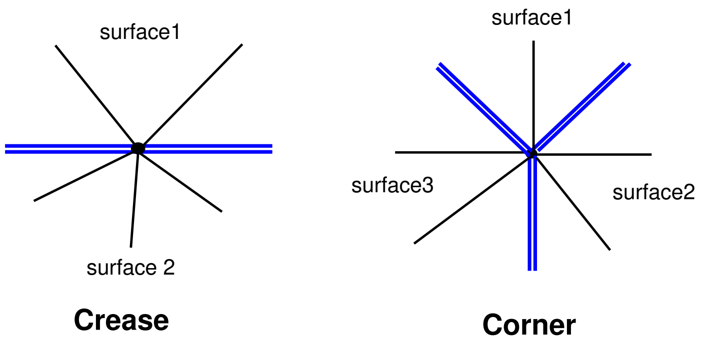

Figure 3.5: Crease curves separate smooth surface regions.

vertices so that a regular control net (consisting only of ordinary vertices) can be found arbitrarily close to an extraordinary vertex. At ordinary vertices, the limit surface is a quartic Box-spline.

We apply this same logic to the piecewise smooth scheme. First, we note that the rules for the smooth vertices and edges are exactly those of Loop's scheme. Second, each of the piecewise smooth rules serves to isolate the extraordinary smooth vertices as well as the crease edges and sharp points (conical and cusp vertices). Hence, we conclude that the surface must be G^2 except at extraordinary smooth points, along creases, and at sharp points.

For the smooth extraordinary vertices, we expect to apply eigenanalysis and characteristic maps to establish convergence to a G^1 smooth surface. At conical and cusp points, we expect the eigenanalysis to predict their singular behavior.

Along a crease, we desire to show that each surface that borders the crease converges to a well-defined tangent plane smooth surface at each point of the crease. Unlike the smooth regions, we have not yet shown that we can isolate the points where smoothness is in question. Luckily, it turns out that we can indeed do this. In fact, we can show that along a crease (or boundary) the bordering surfaces are G^2, except at extraordinary crease vertices. The proof, due to Tom Duchamp, follows.

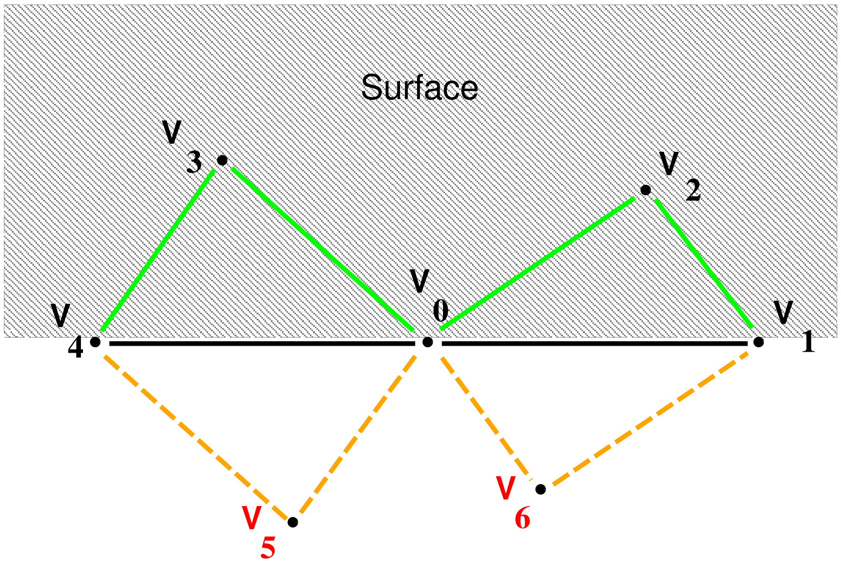

Figure 3.6: Construction for proving smoothness at an ordinary crease vertex.

Proof of G^2 at Ordinary Crease Vertices.

Consider a single side of a crease at an ordinary crease vertex V_0. The valence of V_0 must be 4. There are two crease neighbors V_1 and V_4 and two non-crease neighbors, V_2 and V_3. We define two temporary vertices V_5 and V_6 on the other side of the crease to define a regular local neighborhood for V_0. See Figure 3.6. We claim that we can position V_5 and V_6 so that applying the smooth ordinary rule to this neighborhood results in the same limit surface (when trimmed by the crease curve) as would result from the crease rules applied to the original vertices alone. This means that the crease rules define a surface at V_0 that is a quartic Box-spline and hence is G^2. We now verify this claim.

We set

$$V_5 = V_0 + V_4 - V_3 \text{ and } V_6 = V_0 + V_1 - V_2.$$

Applying the smooth rule to these vertices results in

$$V_0' = (10V_0 + V_1 + \ldots + V_6)/16 = (6V_0 + V_1 + V_4)/8$$

and

$$V_1' = (3V_0 + 3V_1 + V_2 + V_6)/8 = (V_0 + V_1)/2,$$

$$V_4' = (3V_0 + V_3 + 3V_4 + V_5)/8 = (V_0 + V_4)/2.$$

Thus, the crease rules are maintained for the vertices along the crease. The non-crease vertices maintain the smooth rule.

We also need to show that the new vertices V_5' and V_6' have the proper positioning for the next subdivision. It is easy to verify that $V_5' = V_0' + V_4' - V_3'$ and $V_6' = V_0' + V_1' - V_2'$.

Thus, we have shown that the crease subdivision rules at an ordinary crease vertex result in a G^2 surface because the resulting surface is identical to the quartic Box-spline surface that results from using the ordinary smooth rule at that vertex.

Since the crease and corner subdivision rules isolate the extraordinary crease vertices and corner vertices, we now have only isolated points along the crease at which we need further proof of smoothness of the bordering surface.

At this points, we have determined that our piecewise smooth subdivision scheme leads to G^2 surfaces except at extraordinary points, extraordinary crease vertices, corners, and sharp points. It is to these isolated excepted points that we now turn our attention.

3.2.2 Extending the Conditions for G^1 at Extraordinary Points

Restating our goals, we wish to show that the surface is G^1 at extraordinary smooth vertices (including darts), and we wish to show that each surface region bordering a crease curve is G^1 at extraordinary crease (and corner) vertices.

To achieve these goals, we will apply the two key analysis techniques that we introduced in Section 2.3, namely, eigenanalysis of local subdivision matrices and construction and analysis of characteristic maps adapted from those defined by Reif [27].

These methods were developed in the context of the symmetric rules for generating smooth surfaces, such as the schemes of Doo-Sabin and Catmull-Clark. As such, these methods apply to the rules (Loop's) for smooth vertices in the piecewise smooth scheme. However, the rules for creases and corners lack the symmetry assumed in the analysis of those previous schemes. Therefore, in order to apply those techniques to the piecewise smooth scheme's rules, some extensions must be made, which we now describe.

In Section 2.3, we presented conditions on the eigenstructure and characteristic map for a subdivision scheme to converge to a tangent plane continuous (G^1) surface at an extraordinary point. We review those conditions here and give some justification for applying them to the crease and corner rules in the piecewise smooth scheme. We also justify the formulas for the limit points and tangent vectors.

Let S be the local subdivision matrix for a neighborhood M^0 of some vertex V_0. We assume that S is non-defective. Let $\lambda_0, \lambda_1, \ldots, \lambda_m$ be the eigenvalues of S in non-increasing order, and let $\{R_i\}_{i=0}^m$ and $\{L_i\}_{i=0}^m$ be orthonormal sets of the corresponding right and left eigenvectors.

Sufficient conditions for the limit surface to have a regular G^1 parameterization at V_0 are:

- $1 = \lambda_0 > \lambda_1 \geq \lambda_2 > \lambda 3$;
- $R_0 = (1, 1, \ldots, 1)^T$;
- The characteristic map is regular and injective.

The limiting position of V_0 is

$$V_0^\infty = L_0 \cdot M^0,$$

and the limiting tangent space at that point is spanned by the vectors

$$\vec{v}_1 = L_1 \cdot M^0 \quad \text{and} \quad \vec{v}_2 = L_2 \cdot M^0.$$

Allowing $\lambda_1 > \lambda_2$ is an extension to the conditions presented previously by Halstead et al. [12] and Reif [27]. We suggest that equality of these sub-dominant eigenvalues is a consequence of the symmetry of the scheme rather than a requirement for the existence of a limiting tangent plane. We now give some explanation for making this extension. In the following discussion, the extraordinary point can be a smooth, crease, or corner point, and their respective neighborhoods are as defined previously in this chapter.

Modeling the Problem

Let M^0 denote the array $(V_0, V_1, \ldots, V_m)^T$ of vertices constituting a d-neighborhood of a extraordinary vertex V_0. We choose $d \geq 2$, so that M^0 contains enough control points to define a non-trivial portion of the limit surface. We assume there are no sharp points in the neighborhood, and that V_0 is the only extraordinary vertex. (We could always subdivide to get such a neighborhood about V_0.)

If S is the $m+1$ by $m+1$ matrix that encodes the rules for deriving the subsequent d-neighborhoods M^k, $k = 1, 2, \ldots$, then

$$M^k = S^k \cdot M^0.$$

For the limit point V_0^∞ to exist, repeated application of S must contract the neighborhoods M^k (and the surface portions they define) to a single point. For the tangent plane to be well-defined at this point, normals to the surface portions defined by the M^k must all converge to a single vector. To verify that the above conditions lead to these results, we consider the following model.

Let s_0 denote the (limit) subdivision surface whose control mesh is M^0. (This is a portion of the surface defined by the whole control mesh of which M^0 is a part.) From Sections 2.2.2 and 3.2.1 we know that the subdivision surface is a quartic Box-spline except at smooth and crease extraordinary points. Thus, s_0 is a quartic Box-spline surface except at the limiting position (if it exists) of V_0.

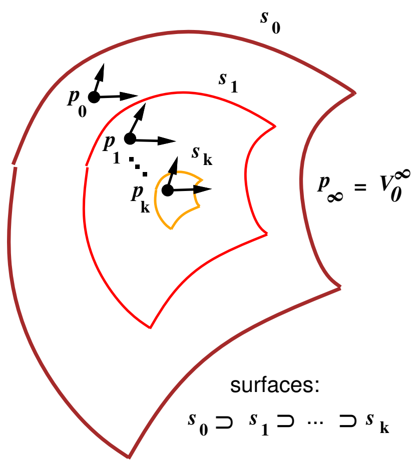

Figure 3.7: A sequence of points p_k approaching the limit point V_0^∞.

Now let s_1 be the portion of the limit surface defined by the control mesh $M^1 = S \cdot M^0$. Then surface s_1 is contained in surface s_0. By continuing, we can obtain a sequence of nested surfaces $s_0 \supset s_1 \supset s_2 \supset \cdots$ that we wish to show converge to the single point V_0^∞. See Figure 3.7.

Finding the Limit Point

To show convergence, we trace an arbitrary point through this sequence. Let p_0 be a point on s_0 that is far enough away from the singularity V_0^∞ so that it lies in a quartic Box-spline patch defined by control points in M^0. Then, we can write p_0 using the quartic Box-spline basis functions as

$$p_0 = \mathcal{B}(p_0) \cdot M^0,$$

where $\mathcal{B}(p_0)$ is an array of the quartic Box-spline basis functions arranged to fit the ordering of control vertices in M^0. From Section 2.2.2, we know that all but 12 of these functions will be zero-valued at p_0. We also know that these basis functions sum to 1, that is, $\sum_{i=0}^{m} \mathcal{B}(p_0)[i] = 1$. Let p_k be the corresponding point on the surface s_k, that is, $p_k = \mathcal{B}(p_0) \cdot M^k$.

Now M^0 is an $m+1$ by 3 real matrix, so it can be expanded in the basis of the $m+1$ column vectors R_i that are the right eigenvectors of the non-defective subdivision matrix S. We can write this as

$$M^0 = \sum_{i=0}^{m} R_i \; m_i,$$

where $m_i \in \mathbb{R}^3$ and $m_i = L_i \cdot M^0$. (Since $L_i \cdot R_j = \delta_{i,j}$.)

The neighborhoods $M^k = S^k \cdot M^0$ then become

$$M^k = \sum_{i=0}^{m} S^k \cdot R_i \; m_i = \sum_{i=0}^{m} \lambda_i^k R_i \; m_i$$

because $S \cdot R_i = \lambda_i \; R_i$.

Therefore,

$$p_0 = \sum_{i=0}^{m} (\mathcal{B}(p_0) \cdot R_i) \; m_i$$

and

$$p_k = \sum_{i=0}^{m} \lambda_i^k (\mathcal{B}(p_0) \cdot R_i) \; m_i.$$

We can easily see now that if $\lambda_0 = 1$ is the single largest eigenvalue (in absolute value) and $R_0 = (1, 1, \ldots, 1)^T$, then as $k \to \infty$, $p_k \to m_0$. (Since $\mathcal{B}(p_0) \cdot R_0 = 1$.) From the above, we know that $m_0 = L_0 \cdot M^0$.

Since p_0 is an arbitrary point on the surface s_0, we conclude that under the given conditions, the subdivision operation contracts the surface to a single limit point

$$V_0^\infty = \text{Ł}_0 \cdot M^0.$$

Finding the Tangent Plane

To show the existence of the tangent plane at V_0, we derive a pair of vectors that span it. For this derivation, we assume that the surface s_0 has a regular G^1 parameterization at all points except the limit point. Therefore, all the s_k are regular G_1

surfaces except at V_0^∞, so at each of the points p_k, the surface has continuous first partial derivatives with non-zero Jacobian.

Reif [27] proves that the regularity and injectivity of the characteristic map is necessary for this assumption. In addition, the control mesh M_0 must also be in general position. (Recall the remark at the end of Section 2.3.5.)

Now we consider the tangent vectors at p_k and examine their limiting behavior. At p_0 we can represent two tangent vectors as

$$\vec{p_u} = \mathcal{B}_u(p_0) \cdot M^0 \text{ and } \vec{p_v} = \mathcal{B}_v(p_0) \cdot M^0,$$

where $\mathcal{B}_u$ and $\mathcal{B}_v$ are the derivatives of the basis functions in two coordinate directions u and v. Since the basis functions $\mathcal{B}$ sum to 1, their derivative functions sum to 0.

We once again expand M^0 in the basis of right eigenvectors of the local subdivision matrix S. But we first notice that $\mathcal{B}_u(p_0) \cdot R_0 = 0$ and $\mathcal{B}_v(p_0) \cdot R_0 = 0$ because of the above remark and recalling that $R_0 = (1, 1, \ldots, 1)$. Therefore, the indexing in this expansion begins with 1.

$$\vec{p_u} = \sum_{i=1}^{m} (\mathcal{B}_u(p_0) \cdot R_i)\ m_i \text{ and } \vec{p_v} = \sum_{i=1}^{m} (\mathcal{B}_v(p_0) \cdot R_i)\ m_i.$$

Using Gauss elimination, we can rewrite these two vectors with adjusted coefficients as

$$\vec{x}_0 = (\mathcal{X}(p_0) \cdot R_1)\ m_1 + \sum_{i=3}^{m} (\mathcal{X}(p_0) \cdot R_i)\ m_i$$

and

$$\vec{y}_0 = (\mathcal{Y}(p_0) \cdot R_2)\ m_2 + \sum_{i=3}^{m} (\mathcal{Y}(p_0) \cdot R_i)\ m_i.$$

We can assume that the coefficients of m_1 and m_2 are both non-zero by the regularity of the surface at p_0. (See Reif [27].) We assume that m_1 and m_2 are linearly independent in $\mathbb{R}^3$ because the control points in M^0 are in general position.

Now we can trace these vectors through the surfaces s_k at the points p_k to obtain

$$\vec{x}_k = \lambda_1^k (\mathcal{X}(p_0) \cdot R_1)\ m_1 + \sum_{i=3}^{m} \lambda_i^k (\mathcal{X}(p_0) \cdot R_i)\ m_i$$

and

$$\vec{y}_k = \lambda_2^k(\mathcal{Y}(p_0) \cdot R_2)\ m_2 + \sum_{i=3}^{m} \lambda_i^k(\mathcal{Y}(p_0) \cdot R_i)\ m_i.$$

These vectors will vanish in magnitude as $k \to \infty$. Therefore, we examine the limit of the normalized vectors, $\vec{x}_k/\|\vec{x}_k\|$ and $\vec{y}_k/\|\vec{y}_k\|$.

This gives us

$$\frac{\vec{x}_k}{\|\vec{x}_k\|} = \frac{(\mathcal{X}(p_0) \cdot R_1)\ m_1 + \sum_{i=3}^{m}(\lambda_i/\lambda_1)^k(\mathcal{X}(p_0) \cdot R_i)\ m_i}{\|(\mathcal{X}(p_0) \cdot R_1)m_1 + \sum_{i=3}^{m}(\lambda_i/\lambda_1)^k(\mathcal{X}(p_0) \cdot R_i)\ m_i\|},$$

where we have divided the numerator and denominator by λ_1.

Similarly, we have

$$\frac{\vec{y}_k}{\|\vec{y}_k\|} = \frac{(\mathcal{Y}(p_0) \cdot R_2)\ m_2 + \sum_{i=3}^{m}(\lambda_i/\lambda_2)^k(\mathcal{Y}(p_0) \cdot R_i)\ m_i}{\|(\mathcal{Y}(p_0) \cdot R_2)\ m_2 + \sum_{i=3}^{m}(\lambda_i/\lambda_2)^k(\mathcal{Y}(p_0) \cdot R_i)\ m_i\|}.$$

Now as $k \to \infty$ these vectors become

$$\vec{v}_1 = \frac{(\mathcal{X}(p_0) \cdot R_1)\ m_1}{\|(\mathcal{X}(p_0) \cdot R_1)\ m_1\|}$$

and

$$\vec{v}_2 = \frac{(\mathcal{Y}(p_0) \cdot R_2)\ m_2}{\|(\mathcal{Y}(p_0) \cdot R_2)\ m_2\|}$$

because λ_1 and $\lambda_2 > \lambda_i$, for $i = 3, \ldots, m$.

We can further simplify these equations by cancelling out the common scalar factors $\mathcal{X}(p_0) \cdot R_1$ and $\mathcal{Y}(p_0) \cdot R_2$. This results in vectors that are independent of the choice of starting point p_0: $\vec{v}_1 = m_1/\|m_1\|$ and $\vec{v}_2 = m_2/\|m_2\|$. We further recall that $m_1 = L_1 \cdot M^0$ and $m_2 = L_2 \cdot M^0$.

Therefore, we have established that the limiting tangent plane is defined by the vectors

$$\vec{v}_1 = L_1 \cdot M^0 \quad \text{and} \quad \vec{v}_2 = L_2 \cdot M^0.$$

The regularity of the surface at the limit point also requires that the surface have a non-vanishing Jacobian at that point. This is guaranteed by the conditions stated at the beginning of this section, if we can apply Reif's theorem [27], which says that the regularity and injectivity of the characteristic map (defined on a neighborhood near the extraordinary point) implies the regularity of the surface at the extraordinary point.

In order to apply Reif's theorem to the piecewise smooth rules, we must define the characteristic maps for them and show that they meet the conditions for the theorem. A formal treatment of this is beyond the scope of this document. However, in Section 3.4, we show by construction how we define the characteristic maps for the smooth, crease, and corner rules and how we test them for regularity and injectivity.

3.3 Eigenanalysis of the Local Subdivision Matrices

As discussed in Section 2.3, the eigenanalysis of the local subdivision matrices can be used to predict the behavior of the limit surface. In this section, we show our calculation of the eigenvalues and relevant eigenvectors for each of the local subdivision matrices in our scheme. We show that they meet the criteria for convergence and, where appropriate, tangent plane smoothness of the surface at a vertex to which the rule would be applied. We use them to derive the formulas for limit positions and tangents on the surface.

The rules in our scheme depend only on the immediate neighborhood of a vertex or an edge. Hence, we can define an $n+1$ by $n+1$ subdivision matrix S_n to encode the rules for deriving the subsequent 1-neighborhoods of a vertex V_0 of valence n to obtain

$$\begin{pmatrix} V_0' \\ V_1' \\ \vdots \\ V_n' \end{pmatrix} = S_n \begin{pmatrix} V_0 \\ V_1 \\ \vdots \\ V_n \end{pmatrix}.$$

For the crease and corner rules, V_1 and V_n are crease vertices. (Recall from Sections 3.1.2 and 3.1.3 how we defined the 1-neighborhoods for these rules.) In our analysis, we assume that V_1 and V_2 are ordinary crease vertices, and that $V_2, \ldots, V_{n-1}$ are ordinary smooth vertices. (If they are not, one subdivision step would make them so.) Similarly, for the smooth, conical, and cusp point rules we assume $V_1, \ldots, V_n$ are all ordinary smooth vertices.

We construct characteristic polynomials for the subdivision matrices and find the eigenvalues for the matrices as their zeros. Recall that the characteristic polynomial for an $n+1$ by $n+1$ matrix S_n is defined as

$$P_{S_n}(\lambda) = det(S_n - \lambda I_{n+1}),$$

where I_{n+1} is the $n+1$ by $n+1$ identity matrix.

We solve $P_{S_n}(\lambda) = 0$ for the eigenvalues λ_i, for $i = 0, \ldots, n$. We assume the eigenvalues are ordered so that λ_0 is dominant. For all our subdivision matrices, we find that $\lambda_0 = 1$ and is strictly greater than all the other eigenvalues. This condition is required for convergence of the subdivision scheme (recall Section 2.3).

For each λ_i, we can find the right and left eigenvectors by solving for v in the linear systems $S_n v = \lambda_i v$ and $v S_n = \lambda_i v$ respectively. We denote the right and left eigenvectors by R_i and L_i.

Since our subdivision rules are affine combinations of the mesh vertices, the rows of the S_n sum to 1. This means that corresponding to eigenvalue $\lambda_0 = 1$ is eigenvector $R_0 = (1, 1, \ldots, 1)$. This is another required condition for convergence as described in Section 2.3.

We further recall from Section 2.3 that our primary interest is in the left eigenvectors of the dominant and sub-dominant eigenvalues, namely, $L0$, L_1, and L_2 because they give us the limit positions and tangent vectors of the surface at that vertex.

Thus, we find that the limit position of V_0 is computed from the product

$$V_0^\infty = L_0 \cdot (V_0, V_1, \ldots, V_n)^T.$$

If they exist, the limit tangent vectors at V_0 are given by

$$\vec{v}_1 = L_1 \cdot (V_0, V_1, \ldots, V_n)^T$$

and

$$\vec{v}_2 = L_2 \cdot (V_0, V_1, \ldots, V_n)^T.$$

In the following subsections, we perform eigenanalyses of the local subdivision matrices for the smooth, crease, corner, cusp, and conical vertex rules. We use the results to derive the formulas presented in Section 3.1 for the limit points and tangent vectors.

3.3.1 Smooth Rule

The subdivision matrix for a 1-neighborhood of a smooth vertex of valence n is the $n+1$ by $n+1$ matrix

$$Smooth_n = (1/8) \begin{pmatrix} a & b & b & b & \cdots & \cdots & b & b \\ 3 & 3 & 1 & 0 & \cdots & \cdots & 0 & 1 \\ 3 & 1 & 3 & 1 & \cdots & \cdots & 0 & 0 \\ 3 & 0 & 1 & 3 & 1 & 0 & \cdots & 0 \\ . & . & . & . & . & . & . & . \\ 3 & 0 & \cdots & \cdots & 0 & 1 & 3 & 1 \\ 3 & 1 & 0 & \cdots & \cdots & 0 & 1 & 3 \end{pmatrix},$$

where $a = 8 - nb$ and $b = (1/n)(5 - (3 + 2\cos(\frac{2\pi}{n}))^2/8)$.

This matrix can be written in block form as

$$Smooth_n = (1/8) \begin{pmatrix} a & b \ldots b \\ 3 & \\ : & (8C_n) \\ 3 & \end{pmatrix},$$

where C_n is the circulant matrix whose rows are n permutations of the n dimensional array $(3, 1, 0, \ldots, 0, 1)$.

We can show that the characteristic polynomial for the matrix $Smooth_n$ is related to the characteristic polynomial of C_n from which it thus inherits eigenvalues. The following equation is derived in Appendix B:

$$(\frac{5}{8} - \lambda)\ P_{Smooth_n}(\lambda) = (1-\lambda)(\frac{a-3}{8} - \lambda)\ P_{C_n}(\lambda). \qquad (1)$$

From the theory of circulant matrices (see, for example, Davis [6]), we find that the eigenvalues of C_n are

$$\gamma_j = (3 + 2\cos(\frac{2\pi(j-1)}{n}))/8,$$

for $j = 1, \cdots n$. This is also derived in Appendix B.

Thus, for C_n, 5/8 is always the dominant eigenvalue ($j = 1$); if n is even, $\lambda_{1+n/2} = 1/8$; and the other eigenvalues are all of multiplicity 2 because $\gamma_j = \gamma_{n-j+2}$ for $j = 2, \cdots, \lfloor (n+1)/2 \rfloor$.

Equation (1) then shows us that, except for 5/8, $Smooth_n$ inherits all of C_n's eigenvalues. Additionally, it has eigenvalue 1 and one eigenvalue $(a-3)/8$ that depends on the vertex mask. For our choice of a, we find that this eigenvalue is the square of $\gamma_2 = (3+2\cos(\frac{2\pi}{n}))/8$, which is also an eigenvalue. Since $\gamma_2 < 1$, its square is strictly less than itself. This then leaves the dominant eigenvalue $\lambda_0 = 1$ and sub-dominant eigenvalue $\lambda_1 = \lambda_2 = \gamma_2$.

As we have already remarked, the right eigenvector is $R_0 = (1, 1, \ldots, 1)$. It is easy to verify that the left eigenvector is $L_0 = (w, 1, \ldots, 1)$, where $w = 3/b$.

The block structure of $Smooth_n$ also lets its eigenvectors inherit elements from the eigenvectors of the circulant matrix C_n. For the multiplicity 2 eigenvalue, $\gamma_2 = (3 + 2\cos(\frac{2\pi}{n})/8$, C_n has pairs of orthogonal eigenvectors,

$$(c_1, c_2, \ldots, c_n) \quad \text{and} \quad (s_1, s_2, \ldots, s_n),$$

where $c_i = \cos(\frac{2\pi(i-1)}{n})$ and $s_i = \sin(\frac{2\pi(i-1)}{n})$. It is useful to know that $\sum_{i=1}^{n} c_i = 0$, and $\sum_{i=1}^{n} s_i = 0$. (See Davis [6].)

It is then easy to verify that for the sub-dominant eigenvalues $\lambda_1 = \lambda_2 = (3 + 2\cos(\frac{2\pi}{n}))/8$, the left and right eigenvectors for $Smooth_n$ are

$$L_1 = R_1 = (0, c_1, c_2, \ldots, c_n)$$

and

$$L_2 = R_2 = (0, s_1, s_2, \ldots, s_n).$$

3.3.2 Crease Rule

We now turn to the analysis of the subdivision matrices for ordinary as well as extraordinary crease vertices.

Ordinary Crease Rule

Recall that an ordinary crease vertex has valence 4; that is, it has exactly 2 non-crease vertices in its neighborhood (on one side of the crease). For this analysis, we assume that these crease neighbors are ordinary vertices.

The subdivision matrix for a 1-neighborhood of an ordinary crease vertex is the 5 by 5 matrix

$$Crease_{ord} = (1/8)\begin{pmatrix} 6 & 1 & 0 & 0 & 1 \\ 4 & 4 & 0 & 0 & 0 \\ 3 & 1 & 3 & 1 & 0 \\ 3 & 0 & 1 & 3 & 1 \\ 4 & 0 & 0 & 0 & 4 \end{pmatrix}.$$

By interchanging rows 3 and 5 and columns 3 and 5 in the determinant $P_{Crease_{ord}}(\lambda) = det(Crease_{ord} - \lambda I_5)$, we can see that $Crease_{ord}$ has the same characteristic polynomial as the block triangular matrix

$$(1/8)\begin{pmatrix} 6 & 1 & 1 & 0 & 0 \\ 4 & 4 & 0 & 0 & 0 \\ 4 & 0 & 4 & 0 & 0 \\ 3 & 0 & 1 & 3 & 1 \\ 3 & 1 & 0 & 1 & 3 \end{pmatrix}.$$

Therefore, the characteristic polynomial of $Crease_{ord}$ is

$$P_{Crease_{ord}}(\lambda) = P_B(\lambda)\, P_A(\lambda),$$

where

$$B = (1/8)\begin{pmatrix} 6 & 1 & 1 \\ 4 & 4 & 0 \\ 4 & 0 & 4 \end{pmatrix} \quad \text{and} \quad A = (1/8)\begin{pmatrix} 3 & 1 \\ 1 & 3 \end{pmatrix}.$$

We easily find that

$$P_B(\lambda) = (1-\lambda)(1/2-\lambda)(1/4-\lambda)$$

and

$$P_A(\lambda) = (1/2-\lambda)(1/4-\lambda).$$

Therefore the eigenvalues are:

$$\lambda_0 = 1, \quad \lambda_1 = \lambda_2 = 1/2, \quad \lambda_3 = \lambda_4 = 1/4.$$

The right eigenvector is $R_0 = (1,1,1,1,1)$ and the corresponding left eigenvector is $L_0 = (4,1,0,0,1)$. L_0 gives the formula for the limit point of an ordinary crease vertex.

The right eigenvectors corresponding to the eigenvalue 1/2 can be written:

$$R_1 = (0,0,1,1,0) \quad \text{and} \quad R_2 = (0,1,1/2,-1/2,-1).$$

The corresponding left eigenvectors give the formulas for the tangent vectors

$$L_1 = (0,1,0,0,-1) \quad \text{and} \quad L_2 = (1,1/2,-1,-1,1/2).$$

Extraordinary Crease Rule

Recall that an extraordinary crease vertex has valence $n \geq 2$. For this analysis, we assume that the crease neighbors are ordinary vertices.

The subdivision matrix for a 1-neighborhood of an extraordinary crease vertex of valence n is the $n+1$ by $n+1$ matrix

$$Crease_n = (1/8)\begin{pmatrix} 6 & 1 & 0 & 0 & \cdots & \cdots & 0 & 1 \\ 3 & 5 & 0 & 0 & \cdots & \cdots & 0 & 0 \\ 3 & 1 & 3 & 1 & 0 & \cdots & \cdots & 0 \\ 3 & 0 & 1 & 3 & 1 & 0 & \cdots & 0 \\ . & . & . & . & . & . & . & . \\ 3 & 0 & \cdots & \cdots & 0 & 1 & 3 & 1 \\ 3 & 0 & 0 & \cdots & \cdots & 0 & 0 & 5 \end{pmatrix}.$$

We can again take advantage of block structure to simplify our search for eigenvalues and vectors obtaining

$$Crease_n = (1/8)\begin{pmatrix} 6 & 1 & 0 & \dots & 1 \\ 3 & 5 & 0 & \dots & 0 \\ 3 & 1 & & & 0 \\ : & : & & (8A_{n-2}) & : \\ 3 & 0 & & & 1 \\ 3 & 0 & 0 & \dots & 5 \end{pmatrix},$$

where

$$A_{n-2} = (1/8)\begin{pmatrix} 3 & 1 & 0 & \dots & 0 \\ 1 & 3 & 1 & \dots & 0 \\ . & . & . & \dots & . \\ 0 & \dots & 0 & 1 & 3 \end{pmatrix}.$$

By rearranging rows and columns in the determinant $P_{Crease_n}(\lambda) = det(Crease_n - \lambda I_{n+1})$, we can see that $Crease_n$ has the same characteristic polynomial as a block triangular matrix with diagonal blocks $8A_{n-2}$ and

$$B = (1/8)\begin{pmatrix} 6 & 1 & 1 \\ 3 & 5 & 0 \\ 3 & 0 & 5 \end{pmatrix}.$$

From the theory of block triangular matrices, we find the characteristic polynomial of $Crease_n$:

$$P_{Crease_n}(\lambda) = P_B(\lambda)\; P_{A_{n-2}}(\lambda).$$

Thus, $Crease_n$ inherits all of the eigenvalues of B and A_{n-2}.

We easily find that

$$P_B(\lambda) = (1-\lambda)(5/8-\lambda)(3/8-\lambda).$$

The eigenvalues of A_{n-2} are

$$\alpha_j = (3 + 2\cos(\frac{\pi j}{n-1}))/8,$$

for $j = 1, \cdots, n-2$. (See Appendix B.) They are simple (of multiplicity 1) and strictly bounded above by 5/8.

Now we can assemble the eigenvalues of $Crease_n$. The dominant eigenvalue of $Crease_n$ is $\lambda_0 = 1$, and the sub-dominant eigenvalues are

$$\lambda_1 = 5/8 \;\text{ and }\; \lambda_2 = (3 + 2\cos(\frac{\pi}{n-1})/8).$$

For $n > 3$, $\lambda_2 > \lambda_3 = 3/8$. Thus, the eigenspace of the two sub-dominant eigenvalues λ_1, λ_2 is of dimension 2 for $n > 3$.

For $n = 3$, however, $\lambda_2 = \lambda_3 = 3/8$. We have to give special treatment to this case in our analysis of smoothness because it fails the sufficient condition for having a well-defined tangent plane, namely that $\lambda_2 > \lambda_3$.

If $n = 2$, the eigenvalues of $Crease_n$ are clearly only those of B, which are $1, 5/8$ and $3/8$. While these eigenvalues meet the criteria for a well-defined tangent plane, we find that the characteristic map for this valence is a special case. The analysis of this case follows closely the analysis of the crease as a subdivision curve that is the crease, as one might expect.

Since the rows of the matrix $Crease_n$ all sum to 1, the right eigenvector for eigenvalue λ_0 is $R_0 = (1, 1, \ldots, 1)$. It is easy to verify that the left eigenvector is $L_0 =$

$(3, 1, 0, \ldots, 0, 1)$. This gives us the formula for the limit point of an extraordinary crease vertex for all $n \geq 2$.

The eigenvector corresponding to value $\lambda_1 = 5/8$ can be written as

$$R_1 = (0, x_1, x_2, \ldots, x_n),$$

where $x_j = (n + 1 - 2j)/(n - 1)$ for $j = 1, \ldots, n$. Notice that $x_1 = 1$ and $x_n = -1$. The corresponding left eigenvector is $L_1 = (0, 1, 0, \ldots, 0, -1)$.

The block structure of $Crease_n$ also lets its eigenvectors inherit elements from the eigenvectors from those of the banded matrix A_{n-2} for $n > 2$. We find that the right eigenvector corresponding to the sub-dominant eigenvalue $\lambda_2 = (3 + 2\cos(\frac{\pi}{n-1}))/8$ can be written as

$$R_2 = (0, s_1, s_2, \ldots, s_n),$$

where $s_j = \sin(\frac{\pi(j-1)}{(n-1)})$ for $j = 1, \ldots, n$. Notice that $s_1 = s_n = 0$.

For $n > 2$, we find that the corresponding left eigenvector can be written as

$$L_2 = (0, y, s_2, \ldots, s_{n-1}, y),$$

where $y = \sin(\frac{\pi}{(n-1)})/(2\cos(\frac{\pi}{n-1}) - 2)$.

If $n = 2$, the right eigenvectors corresponding to eigenvalues $5/8$ and $3/8$ are

$$R_1 = (0, 1, -1) \ \text{ and } \ R_2 = (-2, 3, 3)$$

and the corresponding left eigenvectors are

$$L_1 = (0, 1, -1) \ \text{ and } \ L_2 = (-2, 1, 1).$$

3.3.3 Corner Rule

The subdivision matrix for a 1-neighborhood of a corner vertex of valence n is the $n+1$ by $n+1$ matrix

$$Corner_n = (1/8)\begin{pmatrix} 8 & 0 & 0 & 0 & 0 & \cdots & \cdots & 0 \\ 3 & 5 & 0 & 0 & 0 & \cdots & \cdots & 0 \\ 3 & 1 & 3 & 1 & 0 & \cdots & \cdots & 0 \\ 3 & 0 & 1 & 3 & 1 & 0 & \cdots & 0 \\ . & . & . & . & . & . & . & . \\ 3 & 0 & \cdots & \cdots & 0 & 1 & 3 & 1 \\ 3 & 0 & \cdots & \cdots & 0 & 0 & 0 & 5 \end{pmatrix}.$$

The eigenanalysis of $Corner_n$ is very similar to that for $Crease_n$ because the matrix has a similar block structure with the same banded matrix A_{n-2}. We can rewrite $Corner_n$ as

$$Corner_n = (1/8)\begin{pmatrix} 8 & 0 & 0 & \ldots & 0 \\ 3 & 5 & 0 & \ldots & 0 \\ 3 & 1 & & & 0 \\ : & : & & (8A_{n-2}) & : \\ 3 & 0 & & & 1 \\ 3 & 0 & 0 & \ldots & 5 \end{pmatrix}.$$

By rearranging rows and columns in the determinant $P_{Corner n}(\lambda) = det(Corner_n - \lambda I_{n+1})$, we can see that $Corner_n$ has the same characteristic polynomial as a block triangular matrix with diagonal blocks A_{n-2} and

$$(1/8)\begin{pmatrix} 8 & 0 & 0 \\ 3 & 5 & 0 \\ 3 & 0 & 5 \end{pmatrix}.$$

From the theory of block triangular matrices, we find that the characteristic polynomial of $Corner_n$ is

$$P_{Corner n}(\lambda) = (1-\lambda)(5/8-\lambda)^2\, P_{A_{n-2}}(\lambda)).$$

Although $Corner_n$ inherits all of the eigenvalues of A_{n-2}, this time they are all dominated by the *double* eigenvalue 5/8. Thus, the sufficient conditions on the eigenvalues are met for the corner rule for $n \geq 2$.

Again, for $\lambda_0 = 1$, we have the right eigenvector $R_0 = (1, 1, \ldots, 1)$. It is easy to verify that the left eigenvector is $L_0 = (1, 0, 0, \ldots, 0, 0)$.

The eigenvectors corresponding to $\lambda_1 = \lambda_2 = 5/8$ can be written as

$$R_1 = (0, n-1, n-2, \ldots, 1, 0), \quad R_2 = (0, 0, 1, 2, \ldots, n-1),$$

and the corresponding left eigenvectors are

$$L_1 = (-1, 1, 0, \ldots, 0), \quad L_2 = (-1, 0, \ldots, 0, 1).$$

3.3.4 Conical and Cusp Rules

The subdivision matrix for a 1-neighborhood of a conical vertex of valence n is the $n+1$ by $n+1$ matrix

$$Conical_n = (1/8) \begin{pmatrix} a & b & b & b & \cdots & \cdots & b & b \\ 3 & 3 & 1 & 0 & \cdots & \cdots & 0 & 1 \\ 3 & 1 & 3 & 1 & \cdots & \cdots & 0 & 0 \\ 3 & 0 & 1 & 3 & 1 & \cdots & \cdots & 0 \\ 3 & 0 & 0 & 1 & 3 & 1 & \cdots & 0 \\ . & . & . & . & . & . & . & . \\ 3 & 0 & 0 & \cdots & \cdots & 1 & 3 & 1 \\ 3 & 1 & 0 & \cdots & \cdots & 0 & 1 & 3 \end{pmatrix},$$

where $a = 8 - nb$, and $b = (2/n)(1 - \cos(\frac{2\pi}{n}))$.

Notice that the structure of this matrix is identical to that of the smooth vertex. Only now we have different values for a and b.

The dominant eigenvalue is again $\lambda_0 = 1$ and we find $R_0 = (1, 1, \ldots, 1)$ and $L_0 = (w, 1, 1, \ldots, 1)$ where $w = 3/b$ with b as above.

From the eigenanalysis for $Smooth_n$, we recall that only one eigenvalue depends on a/b and it is equal to $(a-3)/8$. For the value we now have for a, this eigenvalue becomes $(3+2\cos(\frac{2\pi}{n}))/8$. However, this is also the value of the next largest eigenvalue, and it has double multiplicity. Therefore, we now have a sub-dominant eigenvalue with triple multiplicity, that is, $\lambda_1 = \lambda_2 = \lambda_3$. By designing our rule to have this eigenstructure, we have chosen to have a vertex where the space of limiting tangents is of dimension 3. This is what we desire for a singularity that behaves like the tip of a cone. (Reif [27] shows that if $\lambda_1 = \lambda_2$, then it is necessary for $\lambda_1 = \lambda_2 > \lambda_3$ for a well-defined tangent plane.)

We can experiment with values for a/b and get a range of surface types. When we take $b = 0$, we get a cusp-like type of singularity. Thus, the subdivision matrix for a 1-neighborhood of a cusp vertex of valence n is the $n+1$ by $n+1$ matrix

$$Cusp_n = (1/8)\begin{pmatrix} 8 & 0 & 0 & 0 & \cdots & \cdots & 0 & 0 \\ 3 & 3 & 1 & 0 & \cdots & \cdots & 0 & 1 \\ 3 & 1 & 3 & 1 & \cdots & \cdots & 0 & 0 \\ 3 & 0 & 1 & 3 & 1 & \cdots & \cdots & 0 \\ 3 & 0 & 0 & 1 & 3 & 1 & \cdots & 0 \\ . & . & . & . & . & . & . & . \\ 3 & 0 & 0 & \cdots & \cdots & 1 & 3 & 1 \\ 3 & 1 & 0 & \cdots & \cdots & 0 & 1 & 3 \end{pmatrix}.$$

Since the rows of the matrix all sum to 1, the dominant right eigenvector for eigenvalue $\lambda_0 = 1$ is $R_0 = (1, 1, \ldots, 1)$. It is easy to verify that the left eigenvector is $L_0 = (1, 0, 0, \ldots, 0, 0)$.

The eigenanalysis can again take advantage of the circulant block C_n because

$$Cusp_n = (1/8)\begin{pmatrix} 8 & 0 & 0 & \ldots & 0 \\ 3 & & & & \\ : & & (8C_n) & & \\ 3 & & & & \end{pmatrix}.$$

This structure of $Cusp_n$ leads to the characteristic polynomial

$$P_{Cuspn}(\lambda) = (1-\lambda)P_{Cn}(\lambda).$$

Thus, $Cusp_n$ inherits all of the eigenvalues of C_n, and those are dominated by the simple eigenvalue 5/8. The next largest eigenvalue is $(3+2\cos(2\pi/n))/8$ and has multiplicity 2. That is, we have eigenvalues $\lambda_0 > \lambda_1 > \lambda_2 = \lambda_3$. It is necessary that $\lambda_1 > \lambda_2$ for the cusp to occur.

The eigenvector corresponding to value 5/8 can be written $R_1 = (0, 1, 1, \ldots, 1)$, and the corresponding left eigenvector is $L_1 = (n, -1, -1, \ldots, -1)$.

3.4 Characteristic Maps

In this section, we show the construction and analysis of the characteristic maps for the rules in our subdivision scheme. We use the characteristic maps to test for the regularity of the limit surface at an extraordinary smooth, extraordinary crease, or corner vertex. As discussed in Section 3.2, these are the points at which the surface cannot be represented as a quartic Box-spline. If the characteristic map corresponding to a type of vertex for a particular valence is regular and injective, then the surface is a regular G^1 embedding at a vertex of that type and valence. We construct the characteristic maps for smooth vertices of valence ≥ 3, and crease and corner vertices of valence ≥ 2. We omit for now the crease vertex of valence 3, which, as explained earlier, is a special case and left for future work. A crease of valence 2 is also a special case that we omit from this discussion. Its analysis follows closely that of the crease as a subdivision curve.

In the following subsections, we describe, in general, how we construct the maps. We then explain our method for testing the regularity and injectivity of the maps. And finally, we show graphs of the maps for the smooth, crease, and corner rules of various valences, and discuss our test results.

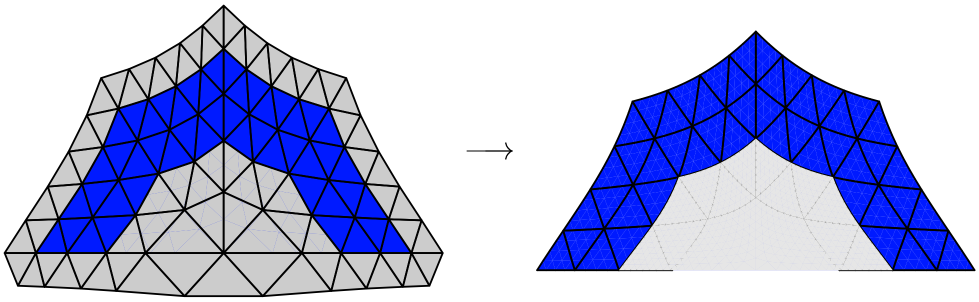

Figure 3.8: Characteristic map for the valence 5 crease rule.

3.4.1 Procedure to Construct Characteristic Maps

In Section 2.3.5, we showed the construction of a characteristic map for Loop's smooth subdivision rules. Figure 2.12 showed the map as the shaded annular shaped surface. This map exhibits the symmetry of the rules that treat all the neighbor vertices in the same fashion. Reif assumed that his characteristic maps would be used for such symmetric rules for which the sub-dominant eigenvalues are equal.

In order to apply this method to our subdivision rules, it must be extended to permit an unequal pair of sub-dominant eigenvalues and subdivision rules that admit an unsymmetric treatment of edges about the vertex because these are conditions of our crease and corner rules.

Figure 3.8 shows the map for a valence 5 crease rule as the shaded "half annular" surface. The shape of the map follows from the shape of the crease neighborhood, which is bounded by the crease edges. A 5-neighborhood was used to create the control mesh for the map, and two rows of faces appear in the shaded portion of the control mesh map as Box-spline patches to the shaded surface that is the image of the characteristic map. A 5-neighborhood is used to exclude the special crease edges (connecting the extraordinary vertex) from the control mesh for the map. An extra layer of vertices is added along the crease edge of the control mesh to complete the mesh for patches bordering the crease. The positioning of these extra vertices and their justification will be explained in the following sections. The effect of the unequal eigenvalues on the map will become apparent as we describe how we analyze the maps.

We now give a general description of the procedure we follow in constructing these maps for the smooth, crease, and corner rules. For a vertex of valence n, we explain how to construct the 2D control net for the map and how to apply the quartic Box-spline basis functions to it.

1. *Create the subdivision matrix S_m for a 2-neighborhood of the vertex.*

   The subdivision matrix S_m for a 2-neighborhood is an extension of the $n+1$ by $n+1$ matrix S_n for a 1-neighborhood for the vertex. We assume that none of the neighbor vertices are extraordinary. For the smooth, crease, and corner subdivision rules, S_n corresponds respectively to the matrices $Smooth_n, Crease_n$, and $Corner_n$ that were given in Section 3.3. Full examples of S_m for the smooth, crease, and corner rules are shown in Appendix C. Their general structure is of interest to us here.

   For a valence n smooth vertex, S_m has dimension $m+1 = 3n+1$, with a block structure
$$S_m = \begin{pmatrix} S_n & 0 & 0 \\ E & (1/16)I_n & F \\ G & 0 & (1/8)I_n \end{pmatrix},$$
   where I_n is the n by n identity matrix.

   For a valence n crease or corner vertex, S_m has dimension $m+1 = 3n$, with a block structure
$$S_m = \begin{pmatrix} S_n & 0 & 0 \\ E & (1/16)J_n & F \\ G & 0 & (1/8)I_{n-1} \end{pmatrix},$$

where J_n is like the identity matrix except the first and last rows have diagonal element $= 2$ instead of 1.

By rearranging rows and columns of either version of S_m, we get a matrix that has the same blocks along the diagonal but is now block triangular. Since the eigenvalues of a block triangular matrix are the union of the eigenvalues of the matrices along its diagonal, we easily see that S_m inherits all of the eigenvalues of S_n plus additional eigenvalues $1/8$ and $1/16$. The eigenvalues of S_n for the smooth, crease, and corner rules, given in Section 3.3, are all $\geq 1/8$. We conclude that the sub-dominant eigenvalues of S_m are the same as those of S_n.

2. *Find the right sub-dominant eigenvectors for the matrix S_m.*

   The right eigenvectors R_1 and R_2 of the subdivision matrix S_m correspond to the sub-dominant eigenvalues λ_1 and λ_2, which are also the sub-dominant eigenvalues of S_n. It is easy to verify from the structure of the matrix S_m that R_1 and R_2 are extensions of the corresponding eigenvectors for S_n. Those eigenvectors were given in Section 3.3 for the matrices $Smooth_n, Crease_n$, and $Corner_n$. The additional elements of R_1 and R_2 are found by solving the linear systems $S^*_m \cdot R_1 = \lambda_1 R_1$ and $S^*_m \cdot R_2 = \lambda_2 R_2$, where S^*_m consists of the rows of S_m excluding the rows from S_n. Examples of the extended eigenvectors are shown in Appendix C.

3. *Construct the 2D control net for the map.*

   We construct a control net for the characteristic map by using the elements of the right eigenvectors for coordinates. We start with points

$$V_i = (R_1[i], R_2[i]),$$

for $i = 0, 1, \ldots, m$. With these vertex values, we generate a 2D control net using the topology of the 2-neighborhood on which the subdivision matrix S_m is defined. Thus, vertex V_i corresponds to the *ith* vertex in the 2-neighborhood.

V_0 is the central vertex. Appendix C shows the topology and the numbering of the vertices in these neighborhoods for the smooth, crease, and corner rules.

We next perform subdivision to increase the size of this mesh and isolate the effects of the extraordinary central vertex and any special crease edge. For the smooth rule, we perform one subdivision. For the crease and corner rules, we subdivide twice.

We note that we could have started with a subdivision matrix for a larger neighborhood to directly obtain a suitable mesh. In the smooth case, we would have needed a 3-neighborhood, and for the crease and corner cases, a 5-neighborhood. However, finding formulas for the eigenvectors for these extended matrices would have been more work than using our subdivision machinery to do it for us. Because the mesh is in an eigenspace, the two methods produce identical meshes.

The control net for the characteristic map is defined as the subset of this mesh that excludes any portion affected by the extraordinary vertex rule or the special crease edge rule. In the case of the smooth rule, the resulting control net has an annular shape and has n-fold symmetry about the origin. For the crease rule, it is a " half annulus", and for the corner, it is a "quarter annulus."

For the crease and corner rules, we perform an additional step. We extend the mesh over the crease edges between ordinary crease vertices to create an equivalent mesh without crease vertex and edge designations. To do this, we use the construction described in Section 3.2.1 and illustrated in Figure 3.6. By positioning the added vertices correctly, the resulting control net will yield the same subdivision surface, but now only ordinary smooth rules are used.

Note that the resulting "annular" control nets for the smooth, crease, and corner characteristic maps are now regular triangular meshes.

4. *Define the map on an annular region in the mesh.*

   We are now ready to define the characteristic map because we have a construction for its control net. The map is simply a subdivision surface defined on its particular "annular" shaped control net. Since the control net contains no extraordinary vertices or crease edges, only the ordinary (smooth) subdivision rules are used, and the resulting "annular" shaped surface will be a quartic Box-spline. Thus, we can use the quartic Box-spline basis functions to represent the map as a piecewise polynomial.

   The characteristic map takes each non-boundary face in the control net to a quartic Box-spline patch. Let $\mathcal{P}$ denote the array $(P_1, P_2, \ldots, P_{12})^T$ of control points for the patch, arranged as in Figure 2.9. Recalling Section 2.2.2, we can represent the patch as a $4th$ degree polynomial in matrix form as

   $$Patch(u, v, w) = Bern^{(4)}(u, v, w) \cdot Q \cdot \mathcal{P},$$

   where $Bern^{(4)}(u, v, w)$ is an array of the 15 Bernstein monomials of degree 4, and Q is a real 15×12 matrix. This form will prove very useful when we calculate the Jacobian of the map to test for regularity and investigate the boundary curves to verify the injectivity of the map.

In the next sections, we explain in some detail how we analyze the characteristic maps. In particular, we present algorithms for testing the regularity and injectivity of the map for the smooth, crease, and corner subdivision rules with a selected valence.

3.4.2 Testing for Regularity

We present an algorithm for testing the regularity of the characteristic map. The algorithm checks that the Jacobian of the map is never zero for each patch in the annular surface region that the map defines. The algorithm uses an expression for the Jacobian as a function of the quartic Box-spline control points $\mathcal{P}$ for the patch.

Representing the Jacobian

Because the map has values in $\mathrm{I\!R}^2$, we can write each patch as $Patch(u,v,w) = (x(u,v,w), y(u,v,w))$. Noting that $w = 1 - u - v$, we can think of $Patch$ as a function of u and v. Then we can write its Jacobian as

$$J(u,v,w) = det \begin{pmatrix} \left(\frac{\partial x}{\partial u} - \frac{\partial x}{\partial w}\right) & \left(\frac{\partial y}{\partial u} - \frac{\partial y}{\partial w}\right) \\ \left(\frac{\partial x}{\partial v} - \frac{\partial x}{\partial w}\right) & \left(\frac{\partial y}{\partial v} - \frac{\partial y}{\partial w}\right) \end{pmatrix}.$$

Since the quartic Box-spline patch can be represented as a $4th$ degree polynomial, its Jacobian is a $6th$ degree polynomial that we can represent as

$$J(u,v,w) = Bern^{(6)}(u,v,w) \cdot Coef(\mathcal{P}),$$

where $Bern^{(6)}(u,v,w)$ is an array of the 28 Bernstein monomials of degree 6, and $Coef(\mathcal{P})$ is an array of 28 real numbers that depend on the patch control points $\mathcal{P}$. We refer to $Coef(\mathcal{P})$ as the Bernstein coefficients for the Jacobian of the patch.

Conditions for the Jacobian to be nonzero

We wish to verify that the Jacobian of the patch is never 0. Since the Bernstein monomials are non-negative and form a partition of unity on the standard domain triangle, it suffices to show that each Bernstein coefficient $Coef_k(\mathcal{P})$ is positive. We will now show how we calculate these coefficients from the 12 control points in array $\mathcal{P}$ for a given patch.

Since our control points are in $\mathrm{I\!R}^2$, we can write $\mathcal{P} = (X, Y)$, where X and Y are column vectors of the x and y coordinates of the control points P_i for the patch. We also abuse notation and write $Coef(\mathcal{P}) = Coef(X, Y)$.

We find that the k-th Bernstein coefficient for the Jacobian can be written in the matrix form

$$Coef_k(X,Y) = 16\ X^T \cdot M^{(k)} \cdot Y,$$

where $M^{(k)}$ is a real 12×12 matrix, for $k = 1, \dots, 28$. In Appendix D, we show this derivation in some detail and give values for the matrices $M^{(k)}$.

Procedure for testing the Jacobian

The algorithm that tests the regularity of the characteristic map is run for a given vertex type (smooth, crease, or corner) and for a given valence n. The algorithm can be summarized in the following steps.

1. Generate the control net for the characteristic map of a particular rule type and valence.

2. For each patch in the resulting surface,

    (a) Find the control points $\mathcal{P}$.

    (b) Calculate the coefficients $Coef_k(\mathcal{P})$, for $k = 1, \ldots, 28$.

    (c) Verify that these coefficients are all positive.

3.4.3 Testing for Injectivity

Here we present an algorithm for testing the injectivity of the characteristic map for the smooth, crease, or corner rule of a particular valence n. This algorithm checks specific properties of the control net for the map that we will identify and demonstrate as being sufficient for the injectivity of the map.

We assume that the map has been shown to be regular. We can then rely on a fact from differential topology that says we need only demonstrate the injectivity of the boundaries of the "annular" surface region defined by the map.

We will define these boundaries for the smooth, crease, and corner characteristic maps. Then we will present conditions on the boundary and prove that they are sufficient for the injectivity of the map. Finally, we describe the algorithm that tests for these conditions from the control net for a particular characteristic map.

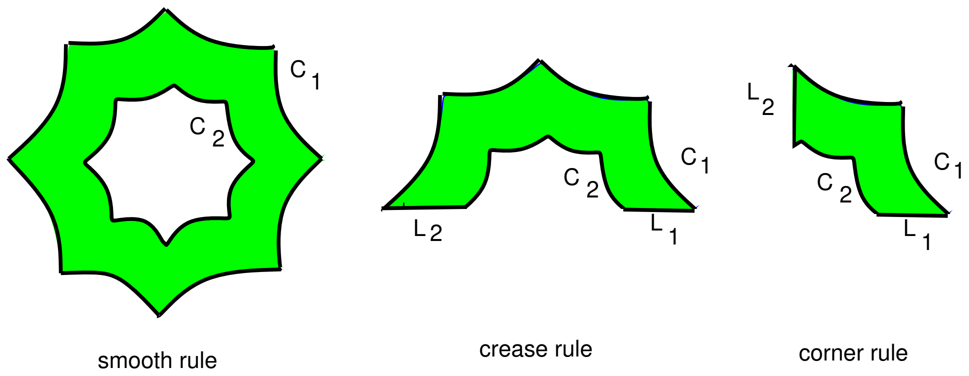

Figure 3.9: Characteristic maps define a bounded region.

Defining the boundaries

We describe the boundaries for the characteristic maps for the smooth, crease, and corner rules as follows. See Figure 3.9.

The smooth rule's characteristic map defines an annulus with outer and inner boundary curves C_1 and C_2, respectively. These curves are related by a dilation because $C_2(s) = \lambda C_1(s)$, where λ is the sub-dominant eigenvalue for the smooth rule of a particular valence. Recall that this eigenvalue has multiplicity 2. These curves have n-fold symmetry about the origin because the control net for the map has that symmetry.

The crease rule's characteristic map defines a "half annulus" with outer and inner boundary curves C_1 and C_2 plus two line segments L_1 and L_2 that lie away from the origin along the positive and negative x-axes, respectively. If we write $C_1(s) = (x(s), y(s))$, then we have $C_2(s) = (\lambda_1 x(s), \lambda_2 y(s))$, where $\lambda_1 > \lambda_2$ are the sub-dominant eigenvalues for the crease rule of a particular valence. C_1 and C_2 are, therefore, related by a linear map. C_1 and C_2 are each symmetric with respect to the y-axis.

The corner rule's characteristic map defines a "quarter annulus" with outer and inner boundary curves C_1 and C_2 plus two line segments L_1 and L_2 that lie away from the origin along the positive x and y-axes, respectively. The curves C_1 and C_2 are related by a dilation because $C_2(s) = \lambda C_1(s)$, where λ is the sub-dominant eigenvalue for the corner subdivision matrix. (Recall that $\lambda = 5/8$.)

Conditions on the boundary

We now identify conditions that we will prove are sufficient for the injectivity of the characteristic map for the smooth, crease, and corner subdivision rules. These conditions are properties of the inner boundary curve that are easy for our algorithm to check.

- For the smooth, crease, and corner rules, we require that the inner boundary curve $C_2(s)$ be radially ordered about the origin. That is, if $s_1 < s_2$, then $C_2(s_1) \times C_2(s_2) > 0$. We also require that the curves not pass through the origin or circle it more than once.

- For the crease rule, we also require that the curve $C_2(s)$ lie in quadrants I and II and that it be transversal to the family of curves, $h_s(t) = (\lambda_1^t\, x(s), \lambda_2^t\, y(s))$. That is, $h_s'(t) \times C_2'(s) > 0$ for all s, where h_s' and C_2' denote the first derivatives. See Figure 3.10.

- For the corner rule, we also require that the curve $C_2(s)$ lie in quadrant I.

Proving the sufficiency of the conditions

For the characteristic map to be injective, each component of the region boundary must be injective, and the separate boundary components must be disjoint.

We first note that if the above conditions hold on C_2, they must also hold on C_1. This is because if $C_1(s) = (x(s), y(s))$, then $C_2(s) = (\lambda_1 x(s), \lambda_2 y(s))$, where $1 > \lambda_1 \geq \lambda_2 > 0$, as described above. This relationship allows us to confine our testing to

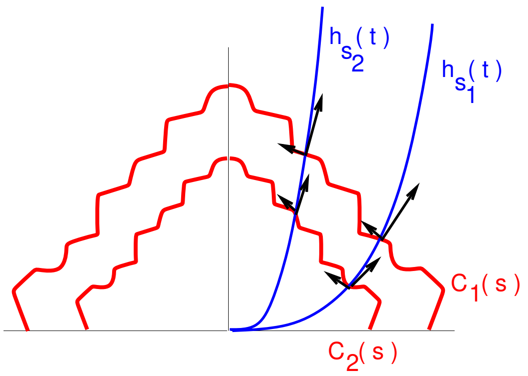

Figure 3.10: The family of curves $h_s(t)$ is transversal to curves $C_1(s)$ and $C_2(s)$.

properties of the inner boundary curves. The injectivity of C_2 implies the injectivity of C_1, for example.

The injectivity of C_2, and hence that of C_1, follows from the first condition. If the curve were self-intersecting, then there would be a radial line that intersected it at more than one point. Then we would have $C_2(s_1) \times C_2(s_2) = 0$, for $s_1 \neq s_2$, contradicting the condition.

To prove that C_1 and C_2 are disjoint, we separate the smooth and corner cases from the crease case.

For the smooth and corner rules, $C_2(s) = \lambda\, C_1(s)$. If these curves had a point in common, then there would be $s_1 \neq s_2$, such that $C_2(s_2) = C_1(s_1)$. However, that would require $\lambda\, C_1(s_2) = C_1(s_1)$, which implies that $C_2(s_2)$ and $C_1(s_1)$ lie along the same radial line, contradicting that $s_1 \neq s_2$.

The case for the crease rule is more complicated because $C_2(s) = (\lambda_1\, x(s), \lambda_2\, y(s))$, and we require the second condition. (By symmetry, we need only consider the curves in quadrant I.) The curves $h_s(t)$ are defined so that $h_s(0) = C_1(s)$ and $h_s(1) = C_2(s)$. Each $h_s(t)$ approaches the origin in the limit of t. Since C_2 is radially ordered about the origin, the second condition assures that the curve family $h_s(t)$ radiates about the origin with respect to s. In particular, if $s_1 \neq s_2$, then h_{s_1} and h_{s_2} are distinct curves.

Now suppose that the inner and outer boundary curves for the crease rule have a point in common. $C_2(s_2) = C_1(s_1)$ for some $s_1 \neq s_2$. Then for all t,

$$h_{s_1}(t) = \left(\lambda_1^t\, x(s_1), \lambda_2^t\, y(s_1)\right) = \left(\lambda_1^{t+1}\, x(s_2), \lambda_2^{t+1}\, y(s_2)\right) = h_{s_2}(t+1).$$

This implies that $h_{s_1}(t)$ and $h_{s_2}(t)$ represent the same curve, thus contradicting that they must be distinct for $s_1 \neq s_2$.

The regions for the crease and corner rules have boundary components, L_1 and L_2; they are each injective because they are portions of a B-spline curve whose control points are increasing along the x- and y-axes. This is easily verified for general valence n by examining the right eigenvectors R_1 and R_2 that define the control net. The fact that the curves C_1 and C_2 are radially ordered ensures that they have only their end points in common with these line segments.

Therefore, we have established the sufficiency of the given conditions for the injectivity of the characteristic map for the smooth, crease, and corner rules.

Evaluating the conditions for a particular map

Now we describe how we check the control net for a particular map for the conditions described above.

We show that the curve C_2 is radially ordered about the origin in two steps. First, we show that the boundary curve for each patch along this border is radially ordered. Then we show that the consecutive patch curves along this boundary lie in consecutive radial wedges about the origin.

To show that a single patch boundary curve $f_i(u)$ is radially ordered (and hence injective), we calculate its 5 Bézier control points $(b_0, b_1, b_2, b_3, b_4)$ and show that they are radially ordered about the origin. We also show that these points do not sweep out an angle $> 2\pi$. See Figure 3.12. We can prove that this is sufficient for injectivity and for the radially ordering of the curve as follows.

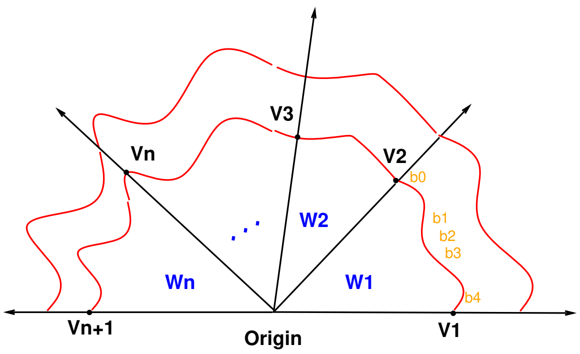

Figure 3.11: Annulus boundary curves.

The *variation diminishing property* of Bézier curves states that a line cannot intersect a Bézier curve more times than it intersects the curve's control polygon. (See Farin [10].) If the control points are radially ordered, then no line from the origin can intersect the control polygon more than once. If the curve doubled back, then a line from the origin through that point would intersect the curve and hence its control polygon more than once.

Next, we show that the control points for each of these curves are separated from the control points of the other curves by their containment in distinct wedges W_i, formed by rays from the origin to the endpoints of the patch boundary curves. See Figure 3.11 for an example for a crease rule. Thus, distinct patch curves along the inner boundary share at most their common end points, and the inner boundary curve as a whole is radially increasing about the origin.

For the crease, the second (transversality) condition can be checked as follows. The condition is that the determinant $h'_s(1) \times C'_2(s) > 0$ be positive for all s. We differentiate $h_s(t)$ with respect to t and differentiate $C_2(s)$ with respect to s. Expanding the determinant then gives us the equivalent condition

$$\ln \lambda_1 \, x(s) \, y'(s) - \ln \lambda_2 \, x'(s) \, y(s) > 0.$$

The left-hand side expression of this inequality can be represented as a piecewise polynomial calculated from the polynomial patch boundary curves $f_i(u)$ that compose

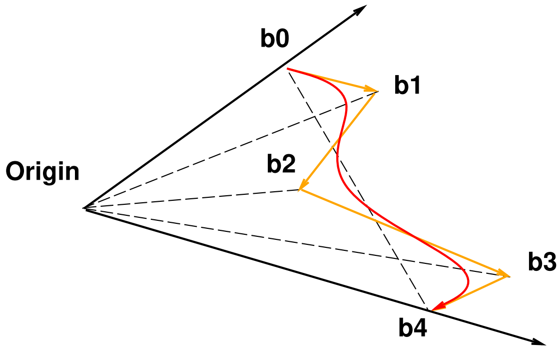

Figure 3.12: Patch boundary curve.

$C_2(s)$. We use the Bézier control points b_0, b_1, b_2, b_3, b_4 to calculate the Bernstein coefficients for the expression, and we test that each of these coefficients for each patch curve is positive to verify the condition.

The Procedure for Testing Injectivity

We can summarize the procedure for testing the injectivity of the characteristic map as follows.

1. We identify, in radial order, the patch boundary curves f_i that comprise the inner boundary C_2 of the annulus, and we calculate the 5 Bézier control points for each patch curve f_i.

2. For each curve f_i, we test that these 5 points are radially ordered as described above. (This proves that the patch boundary is injective.)

3. For the crease, we also test the transversality condition for each component f_i of C_2 by the method described above.

4. We take the consecutive end points $(V_1, V_2, \ldots, V_m)$ of these curves and verify that they are in radial order about the origin. For the smooth rule, we check that they do not circle the origin more than once. For the crease, we check

that they lie within quadrants I and II. For the corner, we check that they lie in quadrant I.

5. We then define distinct radial wedges W_i as the regions separated by the rays from the origin through V_i and V_{i+1} as in the figure. For each wedge W_i, we test that the Bézier control points for the corresponding boundary curve lie inside the wedge defined by their end points. Since the curve lies in the convex hull of the control points, it must lie within the convex wedge region.

Thus, we have specified an algorithm for testing the injectivity of the map for a particular type of vertex with a given valence.

3.4.4 Results

We now show our characteristic maps for the smooth, crease, and corner vertex types. Using a Mathematica implementation, we have generated these maps for consecutively increasing valences n and have obtained only positive results for the regularity and injectivity tests having taken n beyond 100. For the smooth rule, we took $n \geq 3$, for the crease rule $n \geq 4$, and for the corner rule, $n \geq 2$. As explained earlier, we leave for future work the crease case of $n = 3$. We separately investigated the crease case of $n = 2$, and although we do not show the results, we do conclude that its characteristic map is regular and injective.

It is tempting to believe that the surfaces are regular at the vertices of these types for all valences. They clearly are for most practical valences.

We show graphical evidence for our conclusion in the figures that follow. We plot the control nets using the same relative scale for the sub-dominant eigenvectors that we used to present them in Section 3.3. For the smooth case, they are orthonormal, for example. We obtain the image of the characteristic maps by performing 2 subdivisions and calculating the limit points. We plot the boundaries of the patches in this limit surface.

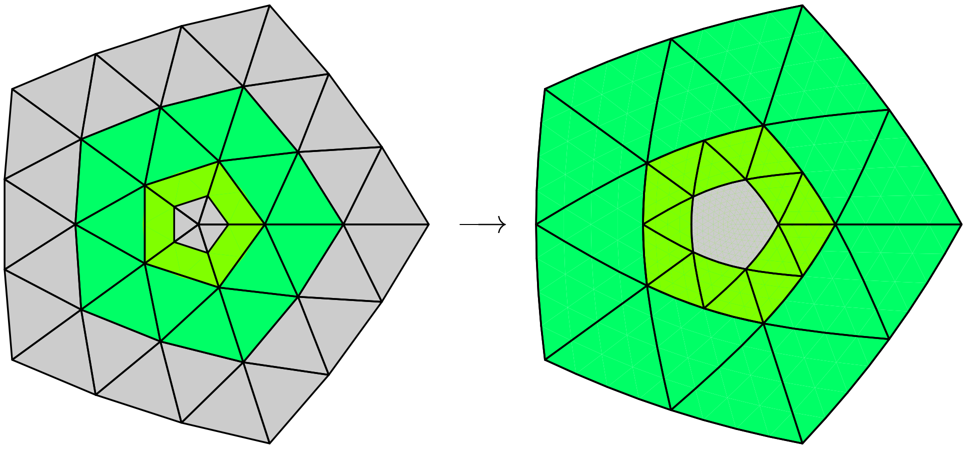

Figure 3.13: Characteristic map for smooth rule, valence 5.

Smooth Rule

Figures 3.13, 3.14, and 3.15 show the characteristic map of the smooth rule for valences 5, 8, and 13 respectively. The left-hand side of each figure shows the control net for the map in the sub-dominant eigenspace; the right-hand side shows its image. Since the sub-dominant eigenvectors are equal, we get n-fold symmetry. The faces in the dark colored control annulus map to the triangular patches that are outlined in the dark colored region in the image. We also show an inner annular region in a lighter shade of the same color. Note that its image connects smoothly with that of the outer annulus.

Crease Rule

Figures 3.16 and 3.17 show the characteristic map for the crease rule for valences 5 and 7, respectively. Since the sub-dominant eigenvectors are unequal, we lose the n-fold symmetry of the smooth case. The left-hand side of each figure shows the control net for the map in the sub-dominant eigenspace; the right-hand side shows its image. The control mesh includes the vertices that were added along the outside of

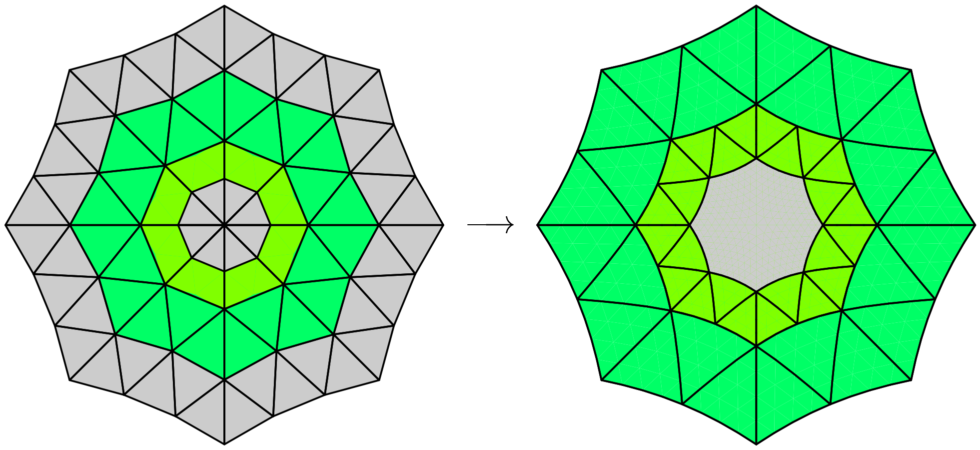

Figure 3.14: Characteristic map for smooth rule, valence 8.

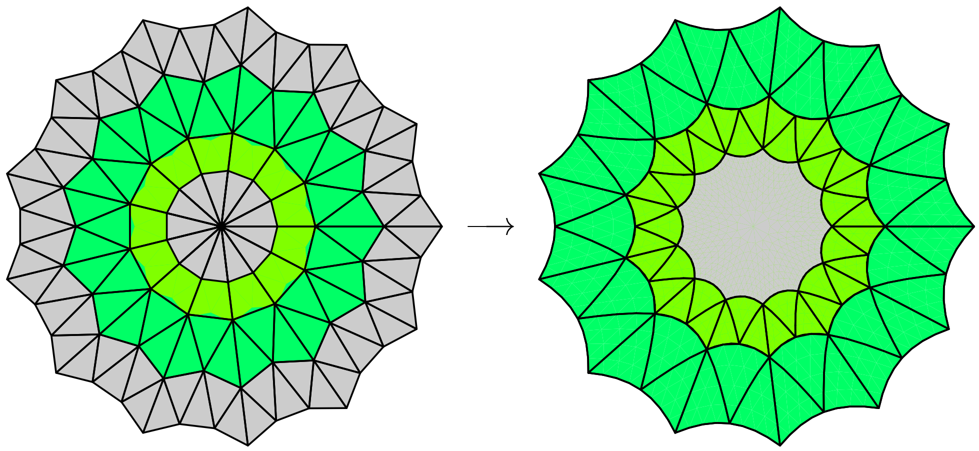

Figure 3.15: Characteristic map for smooth rule, valence 13.

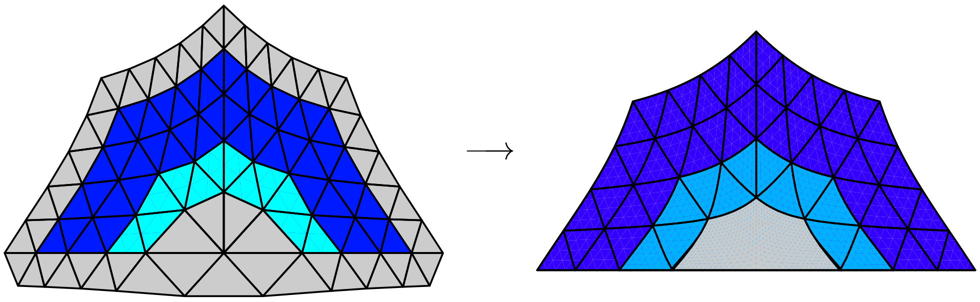

Figure 3.16: Characteristic map for crease rule, valence 5.

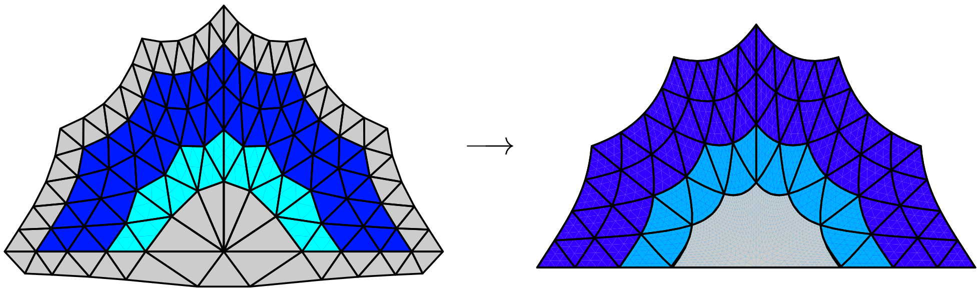

Figure 3.17: Characteristic map for crease rule, valence 7.

the crease to complete the regular control nets for the patches along the crease edge. The faces in the dark colored control annulus map to the triangular patches that are outlined in the dark colored region in the image. We also show an inner annular region in a lighter shade of the same color. Note that its image connects smoothly with that of the outer annulus.

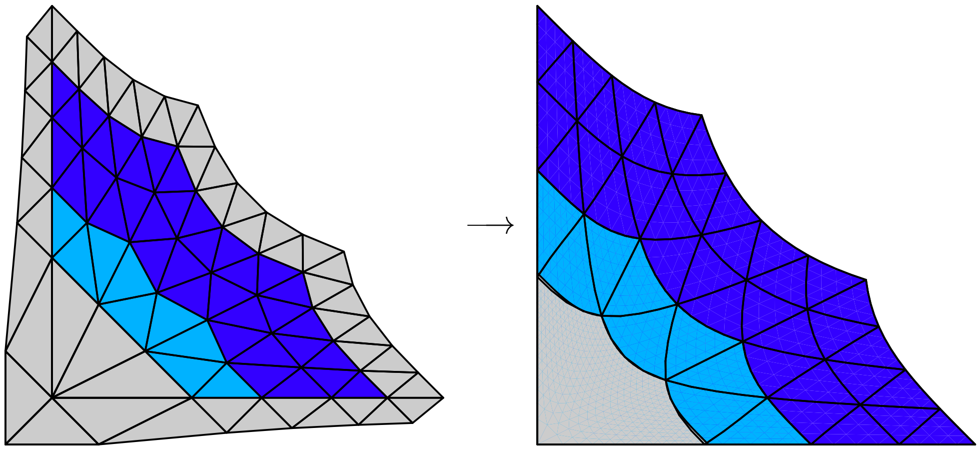

Figure 3.18: Characteristic map for corner rule, valence 4.

Corner Rule

Figure 3.18 shows the characteristic map for the corner rule for valence 4. Since the sub-dominant eigenvectors are equal, we get reflection symmetry in the line $y = x$. The left-hand side of each figure shows the control net for the map in the sub-dominant eigenspace; the right-hand side shows its image. The control mesh includes the vertices that were added along the outside of the crease to complete the regular control nets for the patches along the crease edge. The faces in the dark colored control annulus map to the triangular patches that are outlined in the dark colored region in the image. We also show an inner annular region in a lighter shade of the same color. Note that its image connects smoothly with that of the outer annulus.

3.5 Discussion

One goal of our analysis has been to show that the piecewise smooth subdivision rules define surfaces that are G^1 smooth at all smooth extraordinary vertices and up to creases and boundaries. We also wanted to show that the limit points and tangent vectors to the surface were as expected at all points including along creases, at corners, and at conical and cusp points.

For this purpose, we formulated the eigenanalysis of the local subdivision matrices for each of the rules for general valence. This allowed us to evaluate the conditions on the eigenvalues and eigenvectors for convergence to a G^1 smooth surface for all valences, n. The smooth, crease, and corner rules met the conditions for all n, except the crease rule for $n = 3$.

This general eigenanalysis also allowed us to define the control mesh for the characteristic maps for all n. However, our tests of the properties of the maps must be performed for a given valence. Except for the crease rule $n = 3$, which requires a different method, we succeeded in obtaining positive results for these tests for all valences up to at least 100.

Thus, for practical purposes, we can use the piecewise smooth subdivision rules with the confidence that piecewise tangent plane smooth surfaces result. However, a proof for general valence would be theoretically desirable.

As further motivation for analyzing the rules, we describe our experience with a rule that was originally designed intuitively, but had to be altered because it did not pass the test of its characteristic map. This is the rule for crease edges adjacent to one extraordinary vertex that resulted in what we now call the special crease edge rule.

The need for that special edge rule is counterintuitive. One would most naturally want to place all new crease vertices at the midpoints of the crease edges, that is, to use the ordinary crease edge rule for all crease edges. However, if we analyze the local subdivision matrix with this rule, we find that for valence 8 the control mesh for the characteristic map is badly formed and the map is not regular. Figure 3.19 shows this control net. We also find that, for some other valences, the local subdivision matrix is defective, and the analysis becomes more complicated. For these reasons, we chose to introduce the special crease edge rule.

The particular 3-5 ratio was chosen for the special crease edge so that the largest sub-dominant eigenvalue λ_1 would correspond to the matrix block determined by the crease edge rules. It ensured that $\lambda_1 > \lambda_2$ for all n and that the matrix was non-defective. While other ratios would also meet these requirements, they would

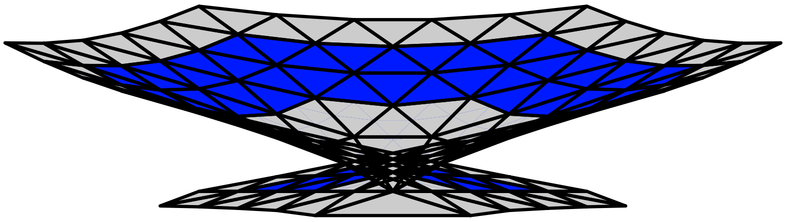

Figure 3.19: The control net for the valence 8 crease rule that does not use the special edge rule.

depend on the valence. Some experimentation with this rule would be interesting, however, and might lead to better shaped crease curves. The current crease curves tend to create a surface that may be flatter than desirable at the extraordinary crease vertices.

3.6 Future Work

We identify some areas where further analysis is needed or extensions to the piecewise smooth scheme could be considered. We also suggest that our algorithms that test the regularity and injectivity of characteristic maps for particular valence be replaced by a proof of that property for general valences.

- Investigate the smoothness of the surface at a valence 3 crease. Because the subdivision matrix for that case does not have a sub-dominant eigenspace of dimension 2, we were unable to apply the method of constructing and analyzing the characteristic map. However, by projecting the 3D control mesh in this eigenspace onto planes of pairs of the sub-dominant eigenvectors, we see promising evidence that the surface is actually smooth at such a vertex. Since this is a common valence in practical meshes, this is an important issue to address.

- Identify other singularity types that would fit into our piecewise smooth subdivision scheme, and analyze the rules for them. It is likely that other singularities would be useful, and input from the design community would be desirable for this. For example, we have not yet addressed the case where a crease ends not in a dart, but in a point that has a discontinuity along the crease edge to it. We expect this pinched surface feature might be of use as it resembles the hull of a ship.

- Investigate other possibilities besides the 3-5 rule for the special crease edge, perhaps one that depends on the valences on both side of the crease. One could examine a local matrix that takes edges on both sides of the crease, and set the edge ratio to get the largest sub-dominant eigenvalue to correspond to the left eigenvector that lies along the crease.

- Write a proof for general n that the smooth rules at extraordinary vertices lead to a G^1 surface. Use the characteristic map to:

  1. Show the regularity for a single patch in an annulus for general n.
  2. Use a symmetry argument to prove that regularity holds if the map is applied to any other patch in the annulus. Rotating commutes with the basis function.
  3. Show the injectivity of the map on the inner boundary of a single patch in an annulus for general n. Show that the map is bounded in a wedge-shaped region for that patch.
  4. Use the symmetry argument again and show that the wedge-shaped regions do not overlap each other.

- Write a proof for general n that the crease and corner rules lead to a G^1 surface because of the regularity and injectivity of the characteristic map. This is more difficult than for the smooth case since the rotational symmetry cannot be used.

Chapter 4

AN APPLICATION OF SUBDIVISION SURFACES

In the interest of widening the application of subdivision surfaces, we investigate a problem motivated by engineering practice. We address the reconstruction of an object from non-uniformly sampled 3D data.

Hoppe et al. [13] have created subdivision surfaces that approximate objects that have been sampled densely and uniformly. Costs associated with this data requirement include sampling time and expense, data processing, and the computer time to run the reconstruction algorithm [16]. The difficulty of obtaining uniform samples from complex objects poses another problem. For example, the physical dimensions of a scanner may make it impossible to reach certain regions such as in holes or pockets. There may also be regions that are occluded from the view of a scanner. For these reasons, we are motivated to develop an algorithm that reconstructs an object from non-uniformly sampled data.

The work of Hoppe et al. is most appropriate for data gathered by a laser range scanner. We are interested, on the other hand, in providing an algorithm that would accommodate less expensive hand held scanners. In this context, the user would be involved in selecting regions to sample and has full knowledge of the geometry and topology of the object. Typically, he would sample more densely in regions of complex geometry. However, knowing the application of the object, he might sample more densely in regions of higher importance where more accurate reconstruction is required or where more detail is required for manipulating the object. He could also judge an area to be of less importance regardless of its geometry (or topology) and sample sparsely.

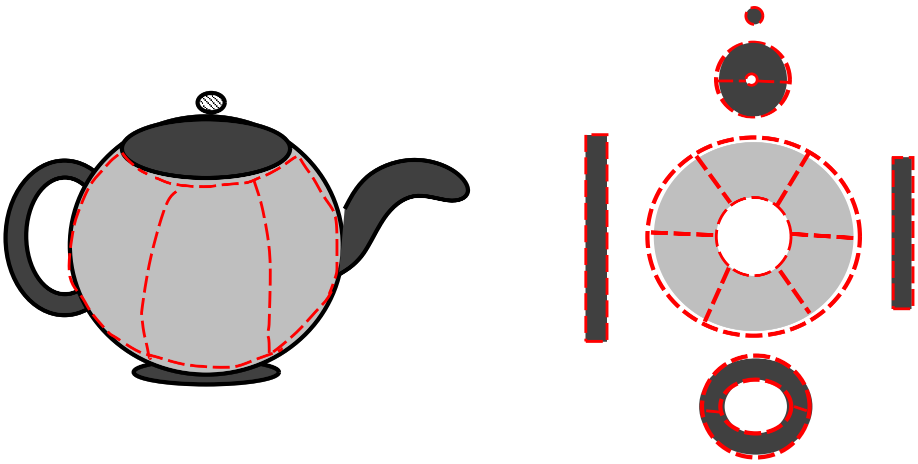

Figure 4.1: Sampling regions to reconstruct a teapot.

We propose the following plan for economically sampling an object to capture its key shape. First, identify smooth regions of planar topology and collect uniform samples of data along their boundaries. Second, for each region, collect enough data to capture the key topography. Areas of more detail or more importance would be sampled more densely. For example, regions on a teapot could be identified as in Figure 4.1, which includes a plan view of component regions.

To recreate the object as a single surface, we propose an algorithm that first creates curves that approximate the region boundaries and then creates a surface that interpolates these boundary curves and fits the input data for the region interiors. This proposed algorithm also addresses the related problem of constructing a surface that interpolates a curve network and fills the regions between curves. See Figure 4.2.

In this chapter, we first solve the more local sub-problem of constructing a surface that interpolates a closed boundary and approximates a set of non-uniformly placed data points interior to the region. We then suggest how to apply this sub-problem to the global problem.

The chapter is organized as follows. In Section 4.1, we give a specification of the global problem and outline a procedure for its solution. In Section 4.2, we define a sub-problem and present an overview of the algorithm we developed for it. Sections

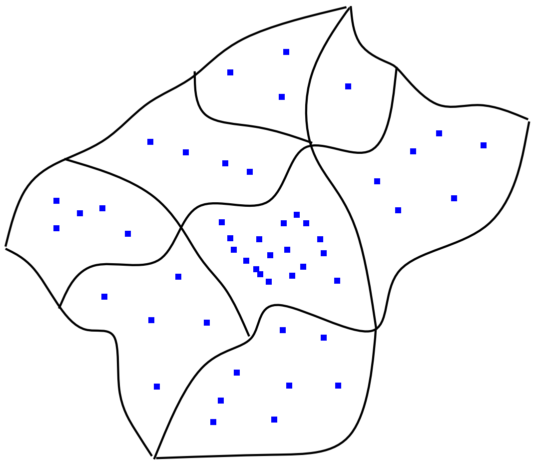

Figure 4.2: A curve network to be interpolated.

4.3 though 4.6 present details of the steps in the algorithm, and Sections 4.7 and 4.8 provide results and discussion of the algorithm. In Section 4.9, we describe how the sub-problem algorithm can be applied to the procedure for the global problem.

4.1 Global Problem Description

As described above, we are interested in solving the global problem of reconstructing an object of arbitrary topological type that has been sampled along curves to define regions of planar topology. We give a specification of this problem, discuss our strategy, and outline a procedure for solving it.

4.1.1 Specification

Input:

- *Curve Network Data*

  We obtain a sequence of densely and uniformly sampled points for each network curve. We assume that nodes in the network are shared curve end points. The curves can be sampled from a surface of arbitrary topological type, but regions between curves must have planar topology.

- *Interior Data*

  For each region bordered by the input curves, there is a randomly distributed set of interior data points.

Output:

- *Subdivision Curves*

  We create a subdivision curve (recall Section 2.2.1) for each network curve. That is, we find control points for each curve so that the resulting curve approximates the network curve data. End points are interpolated.

- *Subdivision Surface*

  We create a G^1 smooth subdivision surface that interpolates the subdivision curves and fits the interior data in each region. While the surface covering each region is topologically equivalent to a disk, the surface can be of arbitrary topological type.

  Optionally, we could designate some of the subdivision curves as creases to introduce sharp features into the surface.

4.1.2 Strategy

To solve this global problem, we follow a strategy that considers the surface as a composite of regions bounded by the network curves, such as in Figure 4.2. Each region will be filled with a fair surface that fits the interior data and interpolates the region boundary. The single control mesh that defines the final subdivision surface would comprise the meshes for all these regions.

To create a single surface that is smooth across the network curves, the surface control mesh will meet certain conditions placed on the vertices along and adjacent to the region boundaries. These conditions will constrain the placement of vertices meant

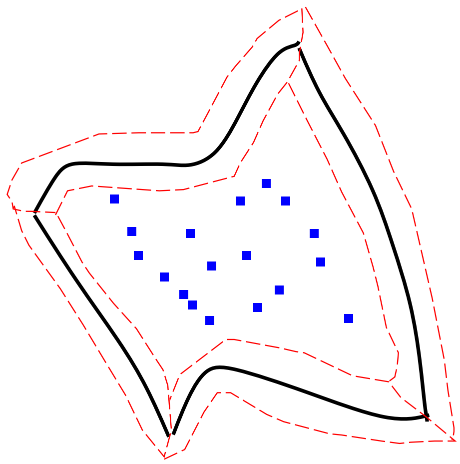

Figure 4.3: A bounded region for the sub-problem.

to optimize the fairness of the surface and its fit to the input data. These constraints are global in nature since they affect adjacent regions.

This strategy allows us to define a sub-problem for which we have implemented the algorithm described in detail in the following sections of this chapter. This sub-problem is a local version of the global problem and consists of filling a single region bounded by a closed curve with a surface that fits the interior data and interpolates the region boundary. Within our algorithm, we also provide a solution to the problem of defining the constraints across a boundary curve. Figure 4.3 shows such a region extracted from the curve network of Figure 4.2 to which the sub-problem could be applied. The dashed lines encircle the boundary curves along which meshes could be specified that guarantee smooth surfaces incorporating the curves.

4.1.3 Procedure

We present here a high level outline of a procedure to solve the global problem; it can be summarized in 4 steps as follows.

1. *Fit the network curves.*

Find control points for each network curve so that the subdivision curve they define comes within an error bound of the input data points.

2. *Create the surface mesh structure.*

   Define the topology of a triangular mesh that incorporates the control points for the network curves. We do this in two steps. First, we create strip meshes along the curves define constraints on the vertex positions in these meshes to guarantee that the surface will be smooth across these curves and at the network nodes. Second, for each region bounded by the curves, we create a mesh of planar topology. The boundary strip meshes and the region meshes combine (by construction) to form a single control mesh for the subdivision surface. Other than the control points for the network curves, the positions of the vertices in the mesh are as yet unassigned. In the next steps, these values are determined by optimizations that are constrained (globally) by the relations placed on the geometry of the boundary strip meshes.

3. *Create fair surface.*

   Apply a fairing procedure to our single large mesh with globally defined constraints. From this optimization, we obtain a surface that interpolates the network curve points and minimizes an energy norm that measures variation in curvature. This surface provides initial parameters for a least squares optimization used in the next step.

4. *Fit surface to interior data and fair.*

   Perform a combination of least square fitting and fairing (as in the last step) and subject the optimization to the global constraints as defined in step 2. The parameters for the least squares fitting are obtained by projecting the data points for the interior regions onto the surface created in the last step. This identifies which control points are to be the "closest points" to the data points.

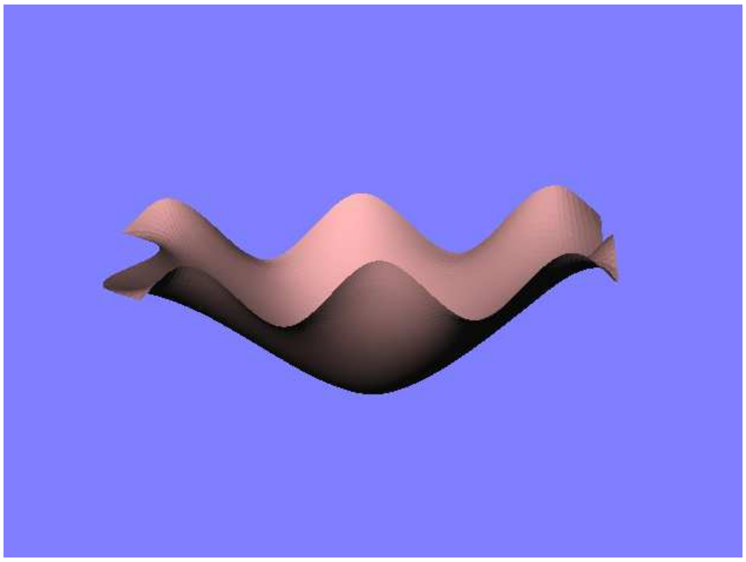

Figure 4.4: A surface fit to a curved boundary and one interior point.

Many of the details of this procedure will be explained when we present the algorithm for the sub-problem. In Section 4.9 we will describe how the sub-problem can be applied to this procedure.

4.2 Sub-Problem Overview

The sub-problem embodies most of the details for the solution to the global problem. Therefore, the main thrust of this chapter will be to describe the algorithm for solving it. In this section, we present the specification for the sub-problem algorithm and outline its procedure. Details will follow in succeeding sections. Figure 4.4 shows a surface created with this algorithm.

4.2.1 Sub-Problem Algorithm Specification

Input:

- *Boundary Data*

We assume that the boundary is closed and consists of a sequence of connected curve segments. For each segment, we are given a sequence of points that lie on the curve. We assume that the points are densely sampled and uniformly distributed.

- *Interior Data*

  We make no assumption about the order of points given in the interior. We expect these points to be randomly selected and non-uniformly distributed.

Output:

- *Boundary Curves*

  For each curve segment, we create a subdivision curve that approximates the given boundary data. Successive curves share end points.

- *Smooth Surface*

  We create a subdivision surface that interpolates these curves and fills the region they bound with a smooth fair surface fitted to the given interior data.

  Thus, the surface we create approximates the input boundary and interior points, has a well-defined tangent plane at each surface point, and is topologically equivalent to a disk.

4.2.2 Procedure

1. *Create curves that approximate the input boundary data.*

   For each set of boundary curve data, perform a least squares best fit by a smooth subdivision curve. That is, find control points for each boundary subdivision curve so that the given data points are within an error bound of the resulting curve. The end points are interpolated.

2. *Create the surface mesh structure.*

Construct a triangular mesh whose topology is equivalent to a disk, and whose boundary vertices are the control points of the boundary subdivision curves. The end points of the boundary subdivision curves become corner vertices in this mesh.

We create the structure of the mesh by defining the connectivity of the set of vertices. The interior vertex coordinates are as yet unassigned.

3. *Create a smooth fair surface that fits the boundary.*

   Find values for the interior vertex coordinates V to create an initial surface in preparation for step 4. We evaluate an energy norm for the mesh and minimize it over values for the interior control points. (The boundary control points are fixed.) That is, we solve the optimization problem:

$$\min_{V} \ \{Energy(mesh)\}.$$

   We use a thin plate energy functional to obtain a surface with low variation in curvature.

   The resulting mesh provides a surface on which the interior data points can be projected to obtain an initial parameterization for the least squares fitting in the next step.

4. *Create a smooth fair surface that fits the boundary and interior points.*

   Find values for the interior control points V, so that the subdivision surface passes within an error bound of the given interior data points P_i. (Again the boundary control points are fixed.) We use a least squares best fit algorithm combined with minimizing an energy norm as in step 3. That is, we solve the following optimization problem:

$$\min_{V} \left\{ Energy(mesh) + \kappa \sum_{i=1}^{n} \| S_i - P_i \|^2 \right\},$$

where κ is a factor that controls the balance of accuracy of fit versus the fairness of the surface. S_i is the closest point on the surface to the interior point P_i and is computed from the control points for the surface derived in step 3.

We describe each of these steps in more detail in the following sections. Then we will show the results of some example data sets.

4.3 Boundary Curve Fitting

We find a least squares best fit by a subdivision curve for each boundary curve. That is, we find control points for the boundary subdivision curves.

If $\{P_i\}_{i=1}^{n}$ are the given data points for one boundary curve, we find control points $\{X_j\}_{j=0}^{m}$ for a subdivision curve $C(u)$, for $u \in [0,1]$, so that the P_i are within an error bound of $C(u_i)$, for some parameter values u_i. In Section 2.2.1 we showed how $C(u)$ can be evaluated.

We wish to solve the non-linear least squares problem:

$$\min_{u_i,m,X_j} \left\{ \sum_{i=1}^{n} \|C(u_i) - P_i\|^2 \right\}.$$

In our solution, we use an alternate projection method: fix the u_i and choose the number of control points $m+1$, solve the minimization for the X_i, then recompute the u_i. We repeat the process until we meet our error bound or until we detect that we can do no better. The algorithm proceeds as follows.

Initially, we choose a normalized chord length parameterization. This is reasonable since we assume that the data points are uniformly and densely sampled.

We begin with 4 control points and perform the fit with this initial parameterization. To evaluate the result, we project the data points onto the curve and look at the parameter values of the projected points. If these projected parameter values are not in the same order as the data points, we increase the number of control points and fit again still using the initial parameters. We repeat this until the projected parameters are in order.

Now we examine the closeness of fit by calculating an error function that is based on the distance between the data points and their projected points. If our error bound is not met, we proceed with further fitting using the projected parameters.

Each time we fit, we examine the error function. If the fit is closer, we fit again using new parameters, those of the projected points on this new curve. If the fit is not improved, we increase the number of control points and fit again using the same parameters as in the last fit. We do this until we meet our error bound or until we meet our maximum number of control points (never more than the number of data points). We also stop if the projected parameters lose their ordering with respect to the data points or if both our last fittings with added control points and with new parameters fail to give improvement.

Some details follow:

- *Chord Length Parameterization*

  Let u_i be the normalized chord length parameter for point P_i, so that $u_1 = 0, u_n = 1$, and
  $$\mid u_i - u_{i-1} \mid = \frac{\|P_i - P_{i-1}\|}{\sum_{j=2}^{n} \|P_j - P_{j-1}\|}.$$

- *Subdivision Curve Control Points*

  The X_j correspond to the parameter values $\frac{j}{m}$, for $j = 0, \ldots, m$. Thus $C\left(\frac{j}{m}\right)$ is the limit point of X_j.

- *Projected Parameterization*

  To simplify the projection, we use the nearest curve point that corresponds, after a certain number of subdivisions, to a control point of the curve. We make this more precise.

For each data point P_i, we find a parameter value u_i as follows. We set $u_1 = 0$ and $u_n = 1$. We subdivide the curve control net k times and find the limit point $C\left((\frac{j}{m})(\frac{1}{2})^k\right)$ that is closest to P_i. Then we take $u_i = (\frac{j}{m})(\frac{1}{2})^k$. To find j, we minimize the following difference over j from 0 to $2^k m$:

$$\|C\left((\frac{j}{m})(\frac{1}{2})^k\right) - P_i\|.$$

$C\left((\frac{j}{m})(\frac{1}{2})^k\right)$ is evaluated from the subdivided control points using the limit mask 1-4-1, where the weight 4 is applied to the control point that corresponds to the parameter value $(\frac{j}{m})(\frac{1}{2})^k$. (Recall Section 2.2.1.)

If we find that 2 or more points P_i get assigned the same parameter value, then we must increase k or disregard one of the points. We choose the latter and typically take $k = 3$.

- *Least Squares Fit*

We perform a least squares fit of the curve $C(u)$ to the points $\{P_i\}_{i=1}^n$ to find the subdivision curve control points $\{X_j\}_{j=0}^m$.

We solve the following optimization for a given parameterization $\{u_i\}$ and a fixed number of control points $m+1$:

$$\min_{X_j}\left\{\sum_{i=1}^{n}\|C(u_i) - P_i\|^2\right\},$$

where $C(u_i)$ is evaluated from the control points X_j as described above.

See Numerical Recipes [24] or Golub and Van Loan [11] for procedures for solving least squares problems.

- *Error Functions*

To evaluate the result, we use the root mean square error

$$MeanErr(C, U, P) = \sqrt{\sum_i \|C(u_i) - P_i\|^2/n},$$

where n is the number of data points P_i.

Another useful error function is the discrete maximum error

$$MaxErr(C, U, P) = \max_i \|C(u_i) - P_i\|,$$

which can also be used to evaluate the results.

4.4 Mesh Construction

We create a mesh with the topology of a disk to fill the region enclosed by the boundary curves. The subdivision surface defined by this mesh will interpolate the boundary curves. We treat each curve on this boundary as a separate crease curve, hence the end points of the curves become corners. We proceed to construct the mesh by first creating a strip mesh around the boundary then creating the interior portion of the mesh.

We color-code the layers in this construction in Figure 4.5. The original boundary is shown in black and has 3 corners, shown with black dots. The boundary strip adds an outer red layer and an inner blue layer. The interior mesh adds the green layers.

We describe the mesh construction in the following steps:

1. *Create mesh strip along the boundary.*

   We begin with the control vertices found for the boundary curves. We create a mesh along this boundary by adding a layer of (temporary) outer vertices and a layer of (permanent) inner vertices.

   We will see later that creating this ring-shaped mesh simplifies the fairing algorithm by creating a control mesh for the surface along this boundary that is regular away from the corner points. The boundary mesh is also useful in extending the algorithm to the more general problem that we described in Section 4.1 because it provides for a smooth surface across the boundary curves. In the

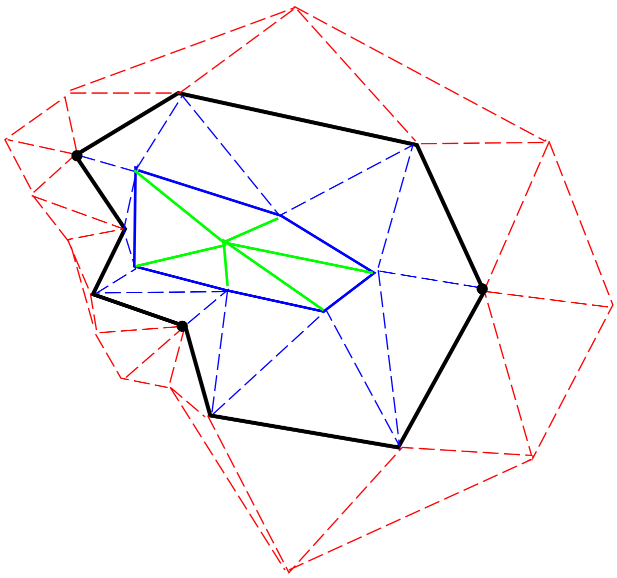

Figure 4.5: Example mesh construction.

final surface, the outer vertices can be dropped because fairing will have been completed.

Since the mesh now has a temporary artificial boundary, we designate the sets of original boundary vertices as creases. This will also distinguish the original boundary vertices from the interior vertices. This designation is appropriate because the subdivision rules treat creases and boundaries alike.

Now each crease corresponds to a separate boundary curve, and, as crease intersections, the curve end points become corner vertices. This allows discontinuities in the boundary where the curves join.

The boundary strip mesh construction is shown in Figure 4.6. For each boundary edge (V_i, V_{i+1}), we add an outer vertex O_i and inner vertex I_i. At a corner C, the inner vertex I is shared by two edges, and we add an extra outer vertex J.

All vertices on the original boundary now have valence 6. (There is really no good reason for requiring that the corner vertices have valence 6, except that it

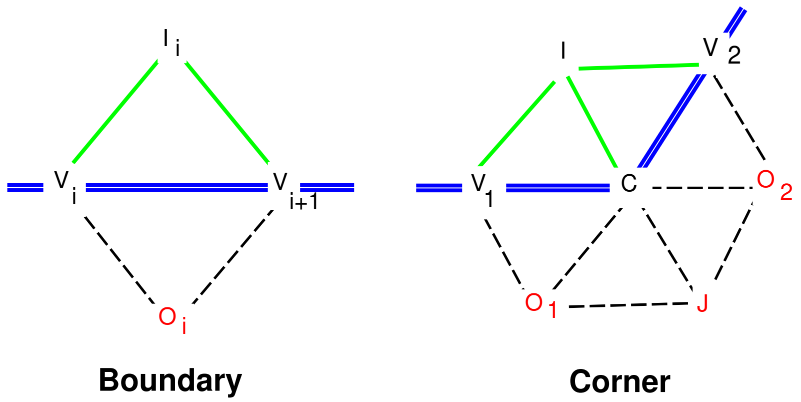

Figure 4.6: Details in the construction of the boundary strip mesh.

is simplifying for all corners to be the same.)

If N is the number of original boundary vertices and R the number of boundary curves (and, hence, the number of corners), then there are $N + R$ outer vertices and $N - R$ inner vertices. In the next section, we start with the inner vertices and create a mesh to fill in the region they bound.

2. *Create a mesh to fill the inner region.*

   We present a simple, straightforward scheme for filling the region bounded by the inner vertices of the boundary mesh of the previous step. The scheme is not meant to be optimal in the number of extraordinary vertices or in their distribution. However, we find that this scheme creates a very reasonable mesh. Extraordinary vertices are generally of valences 5 and 7 and evenly distributed. Any adjacent extraordinary vertices are separated in the next step when we subdivide.

   The mesh scheme proceeds as follows. We treat the region to be filled as hexagonal regardless of the number of vertices on its boundary or the placement of corners in the original boundary. We assign the vertices to the edges of the hexagon so that extra vertices are distributed evenly. Figure 4.7 shows filling regions bounded by 14 and 9 vertices respectively.

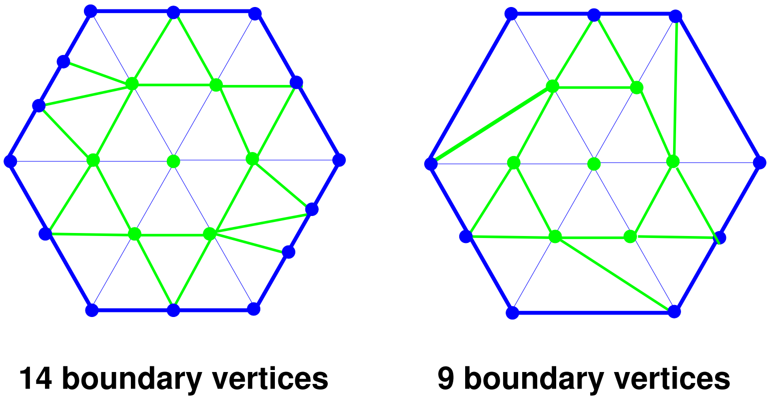

Figure 4.7: The inner region treated as a hexagon.

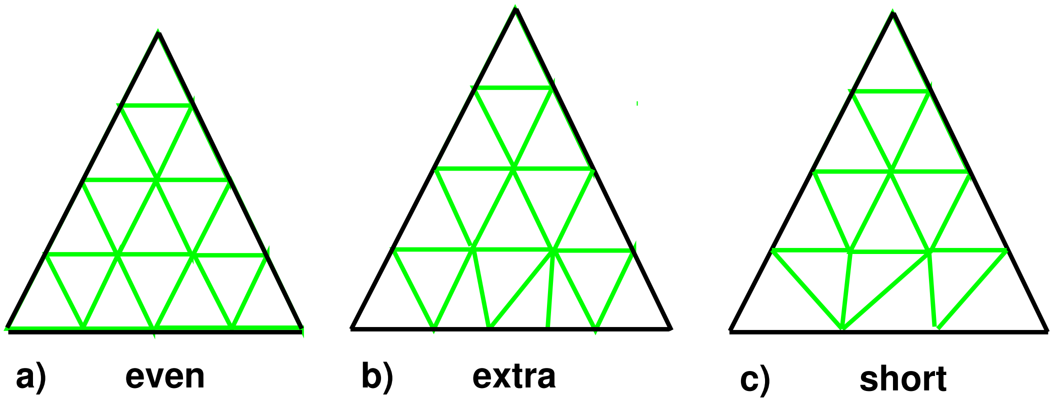

Figure 4.8: The 3 types of meshed triangles created in the inner region.

We fill the hexagonal region with 6 meshed triangles sharing a central point in the hexagon. All triangles have the same number of vertices per edge except perhaps at the region boundary. Edge removals or insertions make up for the irregularities. In other words, we collapse or split faces. Figure 4.8 illustrates the three types of meshed triangles that can result.

More precisely, let n be the number of vertices on the boundary of the region. (This will be the inner boundary from the last step, so $n = N - R$.) The inner edges of the triangles will contain $m + 1$ vertices where $m = Round[n/6]$.

If n is evenly divisible by 6, then the hexagon would have $m+1$ vertices on each side, and each triangle is filled with a mesh having $(m+1)(m+2)/2$ vertices. See Figure 4.8a.

If $n \bmod 6 = 1$ or 2, then we take the first (and possibly 4th) sides of the hexagon to have $m+2$ vertices and so the corresponding 1 (or 2) triangles will have one extra face. See Figure 4.8b..

If instead, n mod $6 = 3, 4$, or 5, then we take the first, third, and fifth sides of the hexagon (as appropriate) to have m vertices and the corresponding triangles have one less face. See Figure 4.8c.

Any extraordinary vertex in this inner mesh will, therefore, have valence 5 or 7. We note, however, that if the inner boundary has $n = 8$ or fewer vertices, we simply add one interior point and connect it with each of the n vertices. This gives us a valence n vertex. (The above algorithm would actually give the same results for these cases, but it's clearer to think of these low valences as a simpler case.) See Figure 4.14 for an example where the central vertex is valence 8.

When we combine this mesh with the boundary mesh of the previous step, we see that there are possibly extraordinary vertices of valence 5 along their common border. The placement of these vertices depends on how the hexagonally-shaped inner mesh is situated with respect to the corners of the original boundary. Examples of these extraordinary vertices appear near the edges in most of the meshes shown in Section 4.7. There are 3 of them in the mesh in Figure 4.12.

3. *Subdivide the mesh.*

Subdividing the mesh simplifies the remaining steps of the algorithm by accomplishing the following goals.

(a) Any adjacent extraordinary vertices introduced in the last step will be separated.

(b) The boundary strip will be isolated from the effects of the extraordinary vertices, and vice versa. Thus, the surface along the boundary away from corners will be composed of ordinary (Box-spline) patches since they will have regular control meshes. In addition, the control net for any interior extraordinary patch will be unaffected by the crease or corner rules.

(c) More vertices are introduced to aid the least squares surface fitting.

After the mesh is subdivided, we drop the outermost boundary layer so that we still have a single layer of vertices outside the original boundary (crease) edge. This single outer layer, plus the boundary crease, plus the first inner layer now compose what we call the boundary strip mesh. It is for the positions of these vertices that the following constraints are defined.

4. *Define constraints on the mesh.*

We define constraints on the positions of the vertices in the boundary strip mesh. These constraints will be used in the fairing optimization to ensure that the surface interpolates the boundary subdivision curves. These constraints also guarantee that away from corners the surface is a G^2 quartic Box-spline along the boundary. This also proves useful for setting up the fairing problem.

The constraints are:

(a) The boundary (crease) vertex positions are fixed.

(b) The outer and inner vertices of the boundary mesh must be placed so that applying the smooth subdivision rules will give the same results as applying the crease rules. This allows us to use a regular control net to represent the non-corner patches along this boundary. The requirement is for $O_i + I_i = V_i + V_{i+1}$. (See the Figure 4.6.) We set this same requirement for all boundary edges, including those adjacent to the corner.

We are able to express these constraints as linear combinations of the vertex positions V. Let M be the total number of vertices in the mesh and N be the number of vertices in the fixed boundary. We can then define a $2N$ by M matrix C and an array D of $2N$ vertex positions so that

$$C \cdot V = D.$$

Thus, if there are N original boundary vertices, the first N rows of C will have exactly one entry of 1 and all the others 0, and the first N rows of D are exactly the boundary vertices. This specifies the boundary interpolation constraints. From the second constraint, there will be N more rows of C each with two entries of 1, two of -1, and all the others 0. The corresponding rows of D are all set to $(0, 0, 0)$.

Methods [11, 24] for solving the optimization problems that we define for fairing and fitting require an initial set of data that satisfies the constraints. Therefore, we assign values to the mesh vertices so that the initial mesh satisfies these constraints. For the first constraint, we simply assign the values of the original boundary curve control points to the corresponding boundary vertices in the mesh. (These values were adjusted by the subdivision of the previous step.) To satisfy the second constraint, we assign the midpoint of the relevant boundary edge to both the inner and outer vertices. Thus, we set $O_i = I_i = (V_i + V_{i+1})/2$.

We must take some extra care at the corners since the inner vertex is shared by two edges. Recall Figure 4.6. We desire $O_1 + I = C + V_1$ and $O_2 + I = C + V_2$. By taking $I = (V_1 + V_2)/2$, we can now solve for the values of O_1 and O_2.

4.5 Fair Surface Evaluation

Our goal now is to create an initial surface that fits the boundary curves. We desire a fair surface because it will be used to obtain parameters for the least squares fitting done in the next step.

We minimize a thin plate energy norm defined on the mesh in hopes of finding values for the control vertices that will yield a surface with minimal variation in curvature.

This optimization is constrained by fixing the boundary (crease) vertex positions to those of the original boundary and by enforcing the relationship among the vertex positions of the boundary strip mesh. These constraints were described in the last section. Here we describe the energy norm, explain how it is calculated for the mesh, and present the optimization to be performed.

4.5.1 Defining Energy Norm

The energy norm is a function of the shape of the surface. If we consider the surface as a composite of patches, then the energy will be summed over all patches:

$$Energy(Surface) = \sum_j E(patch_j).$$

For our subdivision surface, we can consider a patch to be the portion of the surface that corresponds to a (non-boundary) face in the mesh.

As described in Halstead et al. [12], the thin plate energy norm for a surface patch S can be written as

$$E(S) = \int_0^1 \int_0^1 (||S_{xx}||^2 + 2||S_{xy}||^2 + ||S_{yy}||^2) dxdy,$$

where, for example, S_{xx} is the second partial derivative of S with respect to x.

Since S is vector-valued, (S_1, S_2, S_3), we more precisely have $E(S) = E(S_1) + E(S_2) + E(S_3)$ with E defined as above for a single valued S.

Since our patches correspond to triangles, it is more convenient to express S in Barycentric coordinates. To that end, we convert the above equation using a change of variables with $x = u\cos a + v\sin a$, and $y = u\sin a + v\cos a$ as shown in Figure 4.9 where $w = 1 - u - v$. This gives us $dxdy = \sin\theta dudv$, where $\theta = \pi/2 - 2a$.

The resulting energy expressed in variables u and v is:

$$\begin{aligned} E_{uv}(S) = \int_{v=0}^{1} \int_{u=0}^{1-v} & (\quad ||S_{uu}||^2 + (2\cos^2\theta + 2)||S_{uv}||^2 + ||S_{vv}||^2 \\ & + \quad 4\cos\theta < S_{uu} + S_{vv}, S_{uv} > \\ & + \quad 2\cos^2\theta < S_{uu}, S_{vv} >) \sin\theta dudv, \end{aligned}$$

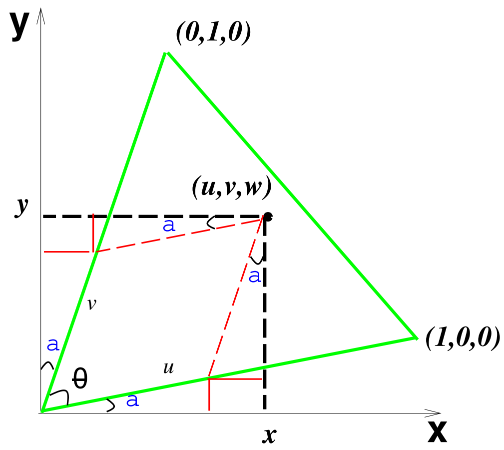

Figure 4.9: Calculating (u, v, w) from (x, y) coordinates.

where $< *, * >$ means the inner product. We make the natural choice of an equilateral domain triangle, and so take $\theta = \pi/3$. We note that this norm is dependent on the parameterization of the patch S.

To obtain a symmetric expression for the energy using barycentric coordinates, we average $E = (1/3)(E_{uv} + E_{vw} + E_{wu})$.

4.5.2 Evaluating the Energy Norm

In order to use the energy norm in an optimization for the control mesh, we need to be able to evaluate $E(S)$ as a function of positions of the vertices in the mesh.

We recall from Chapters 2 and 3 that, for regular portions of the mesh, the corresponding patches are quartic Box-splines. For these *ordinary patches*, therefore, we have a closed-form polynomial expression for S, and we can calculate $E(S)$ directly. To calculate the energy of *extraordinary patches*, we will use repeated subdivision to obtain a sequence of ordinary sub-patches whose energies we can evaluate and combine.

Energy of ordinary patches

For ordinary patches, the calculation is straightforward because we have a polynomial representation for the patch. The quartic Box-spline surface patch can be written as

$$S(u, v, w) = B_q(u, v, w) \cdot \mathcal{P},$$

where $B_q(u, v, w)$ is an array of the quartic Box-spline basis function values, and $\mathcal{P}$ is an array of the (12) quartic Box-spline patch control points. See Section 2.2.2 where $B_q(u, v, w) = Bern^{(4)}(u, v, w) \cdot Q$.

We can use this equation for S in the expression for $E(S)$ derived in the last section. We find that the patch energy $E(S)$ is bilinear in $\mathcal{P}$ because the surface patch S is a linear sum of the points $\mathcal{P}$. Thus, for ordinary patches, we can write the energy in matrix form as

$$E_{ord}(patch) = Trace(\mathcal{P}^T \cdot K_{ord} \cdot \mathcal{P}),$$

where K_{ord} is a 12×12 real symmetric matrix. We show our values for this matrix in Appendix E.

Energy of extraordinary patches

We now consider the patches that are not quartic Box-splines. We consider two cases: patches at a smooth extraordinary point and patches near a corner as shown in Figure 4.10. We can exclude all other crease patches from this discussion because they are ordinary patches by our construction of the boundary meshes.

First, let us suppose that the control net for an extraordinary patch contains an extraordinary point that is not on a crease or boundary. We note that patches with an extraordinary point on the border of its control net are still ordinary because only smooth subdivision rules affect points in the patch; the extraordinary point must be in the interior of the control net to have an effect.

We also assume that there is at most one such interior extraordinary point in each patch control net. This is a valid assumption because any adjacent extraordinary

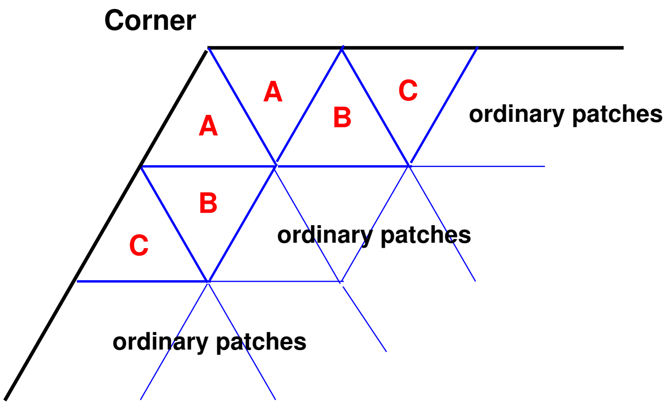

Figure 4.10: Types of patches at a corner.

points created during the mesh construction were separated in the subdivision that was the last step in that construction. (See Section 4.4, step 3.)

Recall Figure 2.10 from Section 2.2.2, which shows control nets for two patches with extraordinary points of valence 5 and 7 respectively. In Section 2.2.2, we discussed how, by repeated subdivision, we could evaluate an extraordinary patch as a sequence of ordinary sub-patches. See also Figure 2.11. We can use this idea to evaluate the energy on a patch because the thin plate energy is calculated from the energies of the sub-patches as

$$E(patch) = 4\sum_{i=1}^{4} E(subpatch_i).$$

(See Halstead et al. [12].) Thus, we can calculate the energy norm for these extraordinary patches by repeated subdivision and summing the energies of their ordinary sub-patches.

We find, however, that for extraordinary patches this sum diverges in the limit. Halstead et al. [12] discuss a solution to this problem and show how to derive an expression for the energy for an extraordinary patch. A simplification is to approximate the energy by taking only a few terms in this series.

Because subdivision is a linear operation on the control net $\mathcal{P}$ for the patch, and because component energies are summed, we can write the resulting energy for the

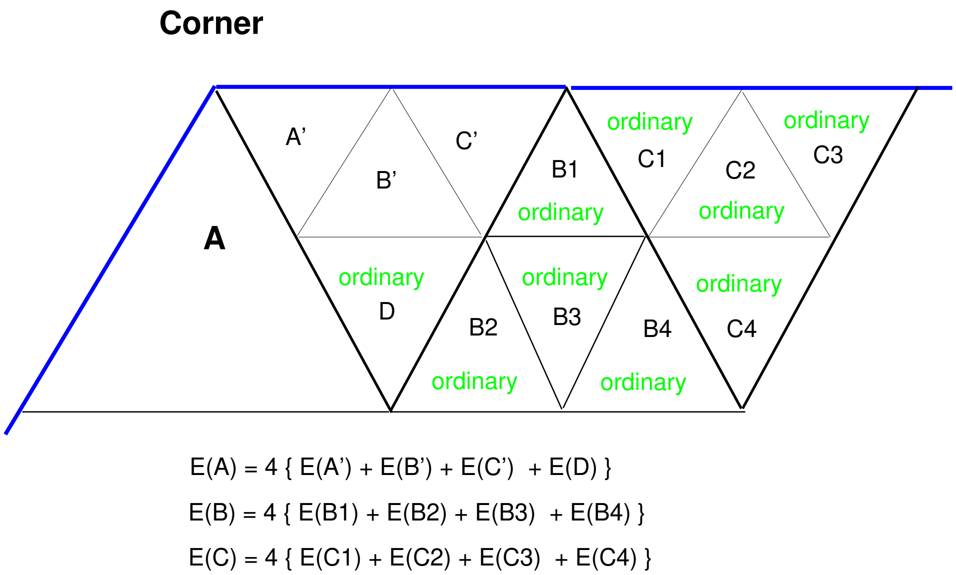

Figure 4.11: Energy of patch type A is summed from its variously typed sub-patches, while the energy of patch types B and C are summed from 4 ordinary sub-patches.

extraordinary patch with a valence n vertex using a matrix K_n as

$$E_n(patch) = Trace(\mathcal{P}^T \cdot K_n \cdot \mathcal{P}),$$

where K_n is a $n+6 \times n+6$ real symmetric matrix. We show values for K_5 and K_7 in Appendix E.

Now we consider patches that are affected by the corner vertex subdivision rule and the special crease edge rule (3-5). (That rule was defined in Section 3.1.2.) Figure 4.10 shows the patches in our design that are so affected. For patches of type B or C, after one subdivision of their control net, each of their 4 component patches is ordinary. (Recall that, in this mesh, we only have ordinary vertices on the crease except at corners.) For these patches, the energy is 4 times the sum of the energies computed from the subdivided control mesh, as shown for the subdivided patch B in Figure 4.11. With this method, we calculate matrices K_B and K_C. (Actually, for each of these, we have 2 mirror image matrices to capture the left- and right-handedness of the corner arrangement.)

Patches of type A, however, require a different scheme because they do not decompose into four ordinary patches. As we do for the patches with extraordinary vertices, we adopt the technique of repeatedly subdividing and summing the energies of the sub-patches. Each subdivision of a patch of type A yields 4 component patches, and they are one each of type A, B, C, and one ordinary patch. Thus, for a patch of type A, the energy would be summed as shown in Figure 4.11. Again, we can calculate a limiting energy matrix K_A.

In the end, we will sum the patch energies for the entire mesh and express the resulting energy using a single matrix K. Thus, we have an expression for the energy of the surface in terms of the vertices V of the control mesh for the entire surface:

$$E(Surface) = Trace(V^T \cdot K \cdot V).$$

4.5.3 Mesh Optimization

We solve the optimization problem,

$$\min_V \{Energy(mesh))\},$$

subject to $C \cdot V = D$, the constraints on the boundary meshes.

For our problem, the energy and constraints decouple into 3 independent optimizations. That is, we minimize the mesh energy for each component of $V = (V_x, V_y, V_z)$.

We find V_x to minimize

$$V_x^T \cdot K \cdot V_x,$$

subject to $C \cdot V_x = D$.

Either a null-space method or Lagrange multipliers can be used to solve for V_x. See Golub and Van Loan, “Matrix Computations” [11] or Numerical Recipes [24].

4.6 Fair Surface Fitting

In this section, we describe how we take the faired mesh of the last section plus the interior data points and create a fair surface that approximates those points and still interpolates the input boundary curves. To do this, we perform a combination of least squares fitting and surface fairing. We weight this combination with our choice of fitness versus fairness.

We note that because we are starting with a surface with low variation in curvature that fits the boundary, we cannot expect a surface with shape more complicated than applying the interior data to this surface might indicate. This is probably exactly what we desire. To say this more concisely, the surface that we now create will be a bijective mapping of points from the initial faired surface.

4.6.1 Parameterizing the Interior Data

We use the faired surface mesh of the last section to obtain an initial "parameterization" for the interior data points to be used in the least squares fitting. For this, we seek the closest points on the surface to the data points. We approximate this projection by finding the limit points of the mesh that are closest to the data points. Thus, for each data point P_i, we find the S_i that among all the limit points is the closest. If S_i is found to be a boundary vertex, we ignore the corresponding data point P_i in our least squares fit because the boundary vertices are fixed.

Using the subdivision formulas for the limit masks, we obtain arrays L_i so that we can write

$$S_i = L_i \cdot V,$$

where V is an array of the vertex positions of the mesh. Combining all the limit positions in matrix form, we can write, $S = L \cdot V$.

4.6.2 Least Squares and Fairing Optimization

To obtain a surface that fits the interior data, interpolates the boundary curves, and is fair, we use an optimization that combines a least squares term with an energy norm term. The optimization problem we solve is:

$$\min_{V}\left\{E(Surface) + \kappa \sum_{i=1}^{n} \|S_i - P_i\|^2\right\},$$

subject to the same constraints, $C \cdot V = D$, as were applied to the previous fairing optimization. κ is a positive real number that weights the fitness versus fairness.

Using the expressions we have for E and S_i, we formulate the problem in terms of the mesh vertex positions V:

$$Trace(V^T \cdot K \cdot V) + \kappa \sum_{i=1}^{n} \|L_i \cdot V - P_i\|^2.$$

This becomes

$$Trace(V^T \cdot M \cdot V) - 2\,\kappa Trace(V^T \cdot L \cdot P) + Trace(P^T \cdot P),$$

where M is calculated as

$$M = K + \kappa\,(L^T \cdot L).$$

Again, either a null-space method or Lagrange multipliers can be used to solve for V.

4.7 Examples and Results

4.7.1 Implementation

We developed a prototype implementation of the algorithm using Mathematica. We take advantage of Mathematica's high-level interface and numerical tools, and we use its graphics capabilities to display meshes. A disadvantage in using Mathematica is slow performance for large meshes. The pictures of limit surfaces are displayed using a subdivision surface viewer written by Kari Pulli [25]. They are simply shown with flat shading.

4.7.2 Simple Examples Showing Mesh Construction and Fairing

We illustrate the mesh construction by showing the results for some simple sets of boundary data.

Our first example uses the sides of an equilateral triangle for three boundary curves. The result after fairing is shown in Figure 4.12. Note the 3 vertices of valence 5.

We use a square for a second example and see the result in Figure 4.14. The central vertex has valence 8 in this case because before subdivision the inner region to be filled has a border of only 8 vertices. We continue this sequence with a pentagon and obtain the less symmetrical result in Figure 4.15. We show one more planar mesh using an octagonal boundary in Figure 4.16.

These examples also illustrate the sharp corners on the boundary. This is an extended capability to the boundaries of Hoppe et al. [13]. In that work, all boundaries of smooth regions are treated as a single smooth crease curve.

4.7.3 Examples Showing Fitting Plus Fairing

We now add a central interior point to the first example and show the resulting surface that is fit to it. See Figure 4.13. The surface has been subdivided once after the fitting and fairing and limit points are taken.

In Figure 4.17, we show the mesh fitted to a single point interior to the octagonal boundary shown in Figure 4.16. Figure 4.25 shows surfaces resulting from adding some more interior points around the rim of this object.

Figure 4.18 shows the mesh for a surface that interpolates a curved boundary. Opposite boundary curves are translations of each other. There are no internal points fitted. Figure 4.19 shows the mesh for a surface with the same boundary now fit to one internal point. Figure 4.4 showed the resulting smooth surface.

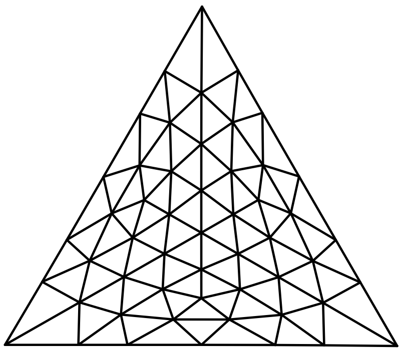

Figure 4.12: A faired mesh with a triangular boundary.

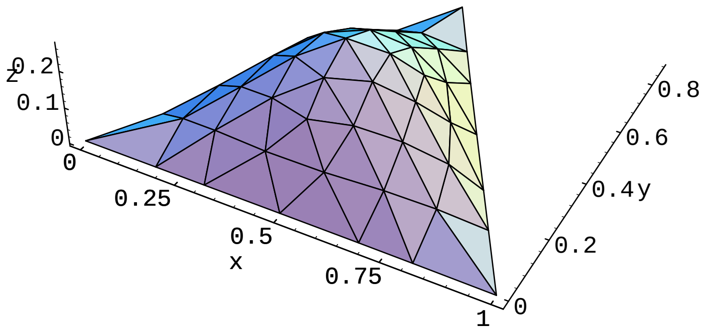

Figure 4.13: A faired surface fit to one interior point.

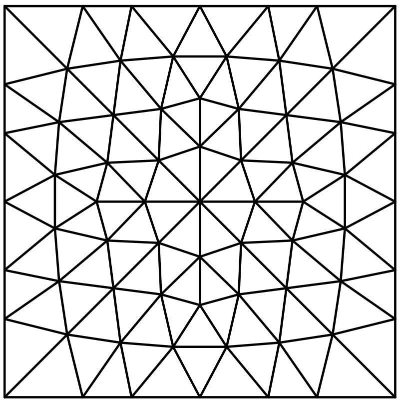

Figure 4.14: A faired planar mesh with a square boundary.

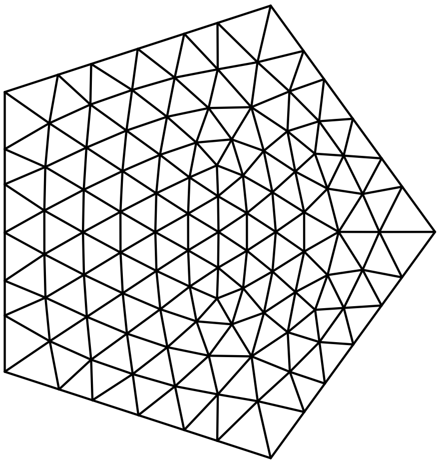

Figure 4.15: A faired planar mesh with a pentagonal boundary.

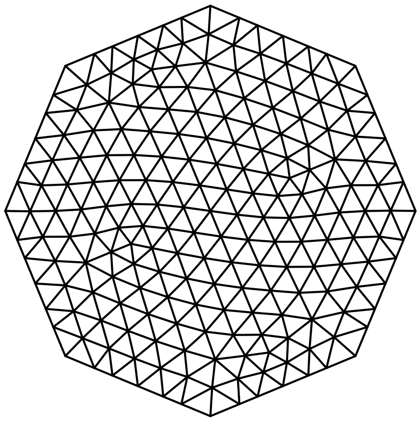

Figure 4.16: A faired planar mesh with an octagonal boundary.

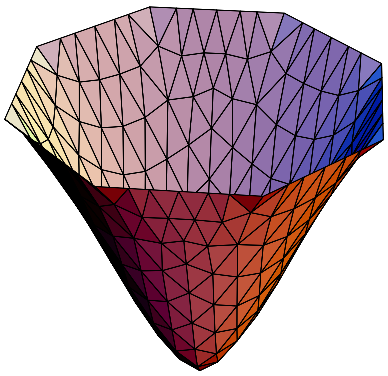

Figure 4.17: A faired surface fit to one interior point.

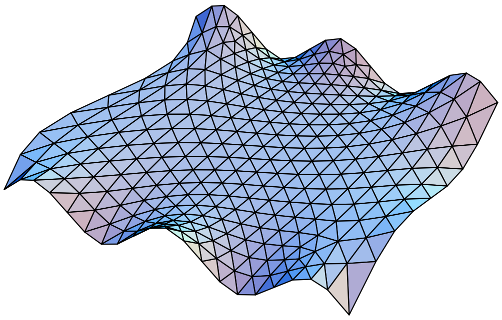

Figure 4.18: A faired surface.

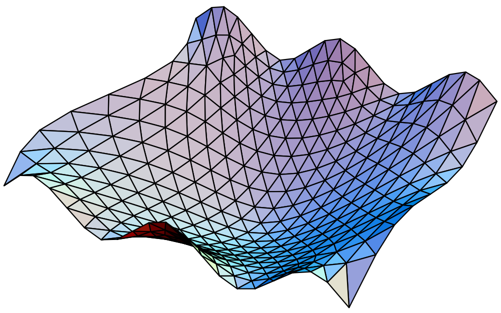

Figure 4.19: A faired surface fit to one interior point.

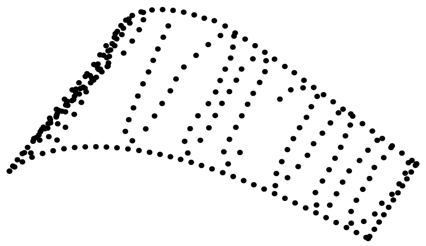

Figure 4.20: A view of the turbine blade as points.

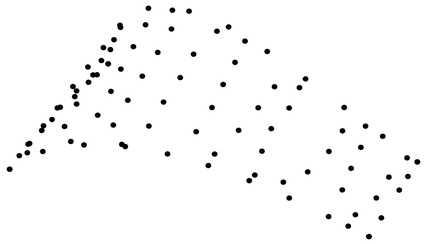

Figure 4.21: Actual points used for reconstruction.

4.7.4 A Mechanical Part

We obtained some data for this work from the Spatial MetriX Corporation. SMX manufactures a 3D touch probe scanner. The data was measured from the surface of a hydro-turbine blade. Boundaries were sampled uniformly and densely, and various interior points were obtained.

Figure 4.20 shows the turbine blade shape from a collection of sampled points. A subset of those points was used in our reconstruction. Figure 4.21 shows the actual points used in the figures that follow. There are 34 boundary points and 61 interior points.

Figure 4.22 shows two views of the mesh that was created and faired to fit the boundary curves. There are 114 vertices in this mesh with 34 (coincidence) boundary points.

Figure 4.23 shows two views of the same mesh after it has been fitted and faired to the interior points. The relatively large number of interior points may be the cause of the rather distorted looking mesh.

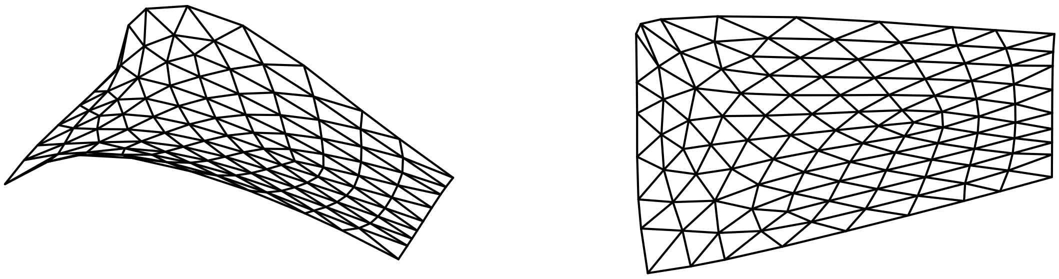

Figure 4.22: Mesh fit to boundary of turbine blade and faired.

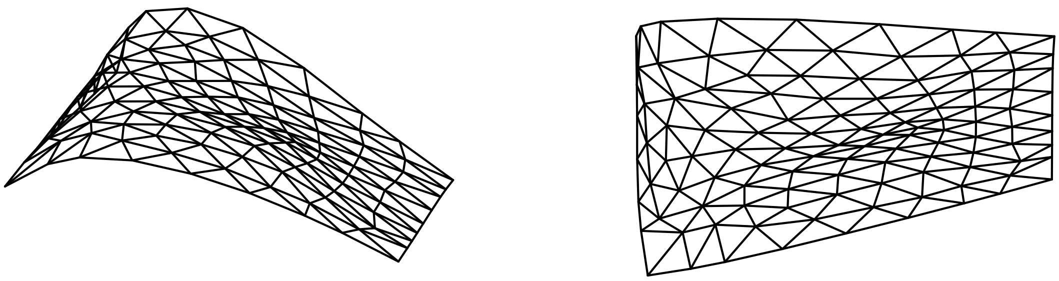

Figure 4.23: Mesh fit to boundary and interior of turbine blade and faired.

Figure 4.24 shows the limit surface for the turbine blade in views before and after fitting the interior points. The views on the left show the surface fit only to the boundary data. The views on the right include fitting to the interior data points.

4.8 Discussion

In this section, we describe some possible improvements and design alternatives that could be applied to the algorithm.

4.8.1 Implementation

We used Mathematica for our prototype because it provides a high-level interface with numerical and graphical tools. It could be rewritten in C++ for better performance and to be able to handle larger data sets. It would also enable better interfaces with other applications.

A real-time version of this algorithm would make its incorporation into an interactive modeling tool feasible. A designer could interactively adjust the shape of his model by moving interior data points, for example.

We should extend our implementation to calculate normals at the vertices of the resulting surface meshes so that better shading could be applied in the viewing program.

4.8.2 Mesh Creation

Our mesh construction is meant to be simple and efficient, and we think our results are reasonable. A more optimal scheme might minimize extraordinary points and find optimal placement for them, but would probably take more complex processing and require handling many special cases. An improvement might be to consider the shape of the boundary when constructing the inner mesh. For example, to fill a region with long skinny shape, a line of meshed triangles would be more appropriate than

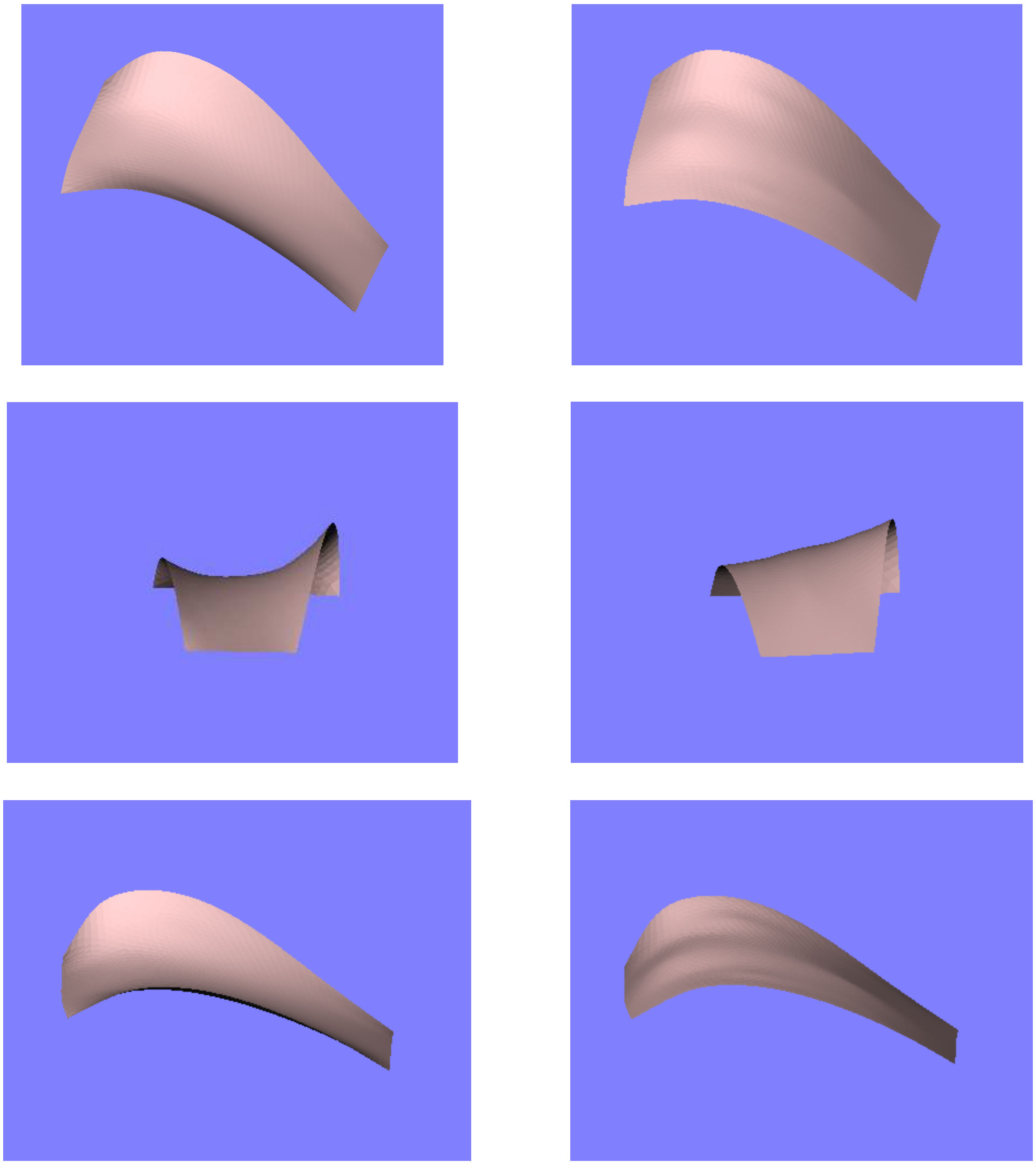

Figure 4.24: The turbine blade.

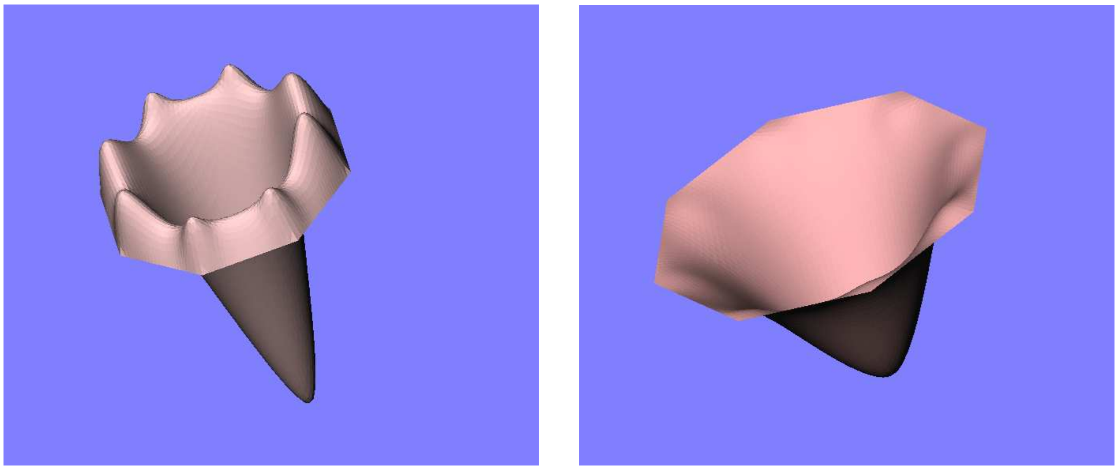

Figure 4.25: The effect of moving the interior points.

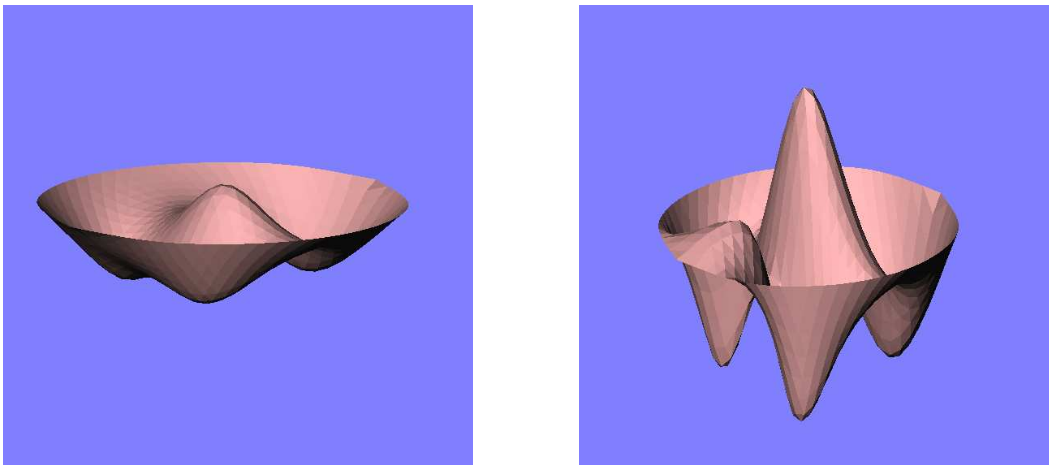

Figure 4.26: The effect of changing the ratio of fit:fair.

the hexagonal configuration that we used. We might also consider the placement of corner vertices along the boundary in designing the inner mesh. These improvements might result in symmetry of the mesh to matching the symmetry of the boundary.

Another concern is that our current meshing scheme favors convex boundaries at corners. Recall Figure 4.6. To better handle concave corners, the placement of vertices in the boundary strip mesh should be inside-out from that for convex corners.

We might consider avoiding the use of the temporary outer boundary vertices. However, these vertices have a fundamental use in the global algorithm where the boundary meshes are the connection between smooth regions across a subdivision curve.

4.8.3 Fairing

We employ a fairing algorithm that reduces the change in curvature. However, that may not be what is always wanted. For example, when used with fitting, many data points are required to create large gradients. The example in Figure 4.26 shows that a high ratio of fitting vs. fairing is needed to obtain a surface resembling the expected shape when only a few internal points are used. Hence, choosing values for this ratio (the constant κ) depends on the object and amount of data. Further experimentation is probably required. Alternatively, some heuristics could be employed to find a good value for κ.

We also question the effect of extraordinary points in the faired surface. It is possible that the energy norm could have a better balance with respect to vertex valence. We would like to investigate weighting the energy as a function of the valence. Preliminary experiments did not show much effect, however.

We mentioned that our method for calculating the energy norm for extraordinary patches results in a divergent series. Our current implementation takes a few terms in th sum of energies from sub-patches. We abandoned the alternative implementation that is based on the scheme in Halstead et al. [12] because the values we obtained were large enough to give numerical difficulties in our software environment. Perhaps a different implementation would iron out this difficulty.

Our mesh construction made it possible to avoid the case where extraordinary vertices lie on the boundary (other than at the corners). This might not be possible if this algorithm is used in a larger context. In that case, we would need to calculate the energy for crease patches directly.

Another consideration would be to try other energy norms, such as the intrinsic (but non-linear) MVS norm developed by Moreton and Séquin [20].

4.8.4 Fitting

Both curve and surface fitting could be improved by using a better parameterization for the least squares sum. Rather than taking the closest control point for each data point, we could use an orthogonal projection onto the boundary curves or surface.

One must also decide how to handle the case of multiple data points having the same nearest control point. We currently don't prevent this and probably have some mesh distortion because of it. We might expect an improvement in the least squares fit by discarding all but the closest data point. An alternative would be to perform another subdivision to obtain more control points. Ideally, the algorithm would subdivide adaptively, that is, only in regions where more points are needed.

Figure 4.25 illustrates the difficulty of creating an object with a ridge, or in this case a vase with a lip. The few interior data points placed along the ridge result in bumps. The proper way to create the lip would be to sample it as another boundary curve thereby creating an object with two regions. This is now a case of the global problem described at the beginning of this chapter.

In general, if the surface being approximated is complex, it should be divided into separate regions. The procedure outlined in the next section suggests how to extend the current algorithm to approximate a more complex object defined by a network of curves.

4.9 Procedure for the Global Problem

As we discussed at the beginning of this chapter, we would like to be able to apply the algorithm to the more global problem of creating subdivision surfaces that interpolate a network of curves and fit random data in the regions bordered by the curves. We would like the surface to be G^1 smooth, but to also allow curves to be creases. Further, we would like the surface to be of arbitrary topology.

We believe that the algorithm we have presented in this chapter provides a basis for solving that problem. A detailed specification of the problem was given in Section 4.1. In this section, we outline a procedure for solving the global problem and indicate how the algorithm can be applied.

The procedure follows a strategy that is similar to that of our simpler problem. We can now think of the surface as a patchwork of regions bounded by the network curves and perform almost the same operations of meshing, fairing, and fitting. However, now we also have constraints that guarantee smoothness across boundary curves and at nodes as well as the constraints that preserve the curves. Further, these constraints are global in nature because they affect adjacent regions. Thus fairing and fitting must be applied globally.

We summarize the steps of the procedure we propose.

1. *Fit the network curves.*

   Using the same procedure as in the basic algorithm, we find control points for each network curve so that the subdivision curve comes within an error bound of the input data points, and the end points are interpolated.

2. *Create the surface mesh structure.*

   Again, we use the strategy of creating strip meshes along the curves and then mesh the interior of the regions. This time the strip meshes provide more than a convenience; they allow us to define constraints that guarantee that the surface will be smooth across these curves. An additional constraint at the network nodes will ensure G^1 smoothness over the entire surface. We describe the steps for creating the mesh as follows.

- Create mesh strips along curves.

  We create a network of strip meshes to cover the control points for the network curves. We do this so that the curve control points away from the network nodes have a regular control net. This means that the subdivision scheme will give us a smooth G^2 surface along the curves except at the nodes. To that end, we constrain the mesh so that the smooth subdivision rules will give us the same results along the curve as would the crease rules. This ensures that the network subdivision curve is interpolated.

  At the nodes in the network, we will get a G^1 surface only if the control point neighborhood of the node is planar. Thus, we define a constraint on those control points to ensure it. Note that this constraint is non-linear.

  As in the previous algorithm, we assign initial values to the non-curve control points so that the constraints are satisfied.

- Create interior region meshes.

  We can now create meshes for each of the regions bounded by the network curves. The first layer of control points in each region has already been created in the strips. We can now employ our same interior meshing strategy for each region independently.

  Note that although the meshes for individual regions are topologically equivalent to a disk, the topology of the surface is not restricted.

- Combine strip and region meshes to get the surface mesh.

  We then collect all the mesh components into one large mesh and set up the constraints. However, this time some of the constraints are non-linear so that we cannot express them as simply as before. Further, our problem does not decouple to treat the x, y, and z coordinates independently.

3. *Create fair surface.*

   We apply our same fairing procedure to our single large mesh with globally defined constraints. We get a resulting surface that interpolates the network curve points and minimizes our energy norm.

4. *Fit surface to interior data and fair.*

   We use the surface just created to obtain parameters to fit the interior data. We project the data points onto the surface to find the closest points. (Again, this can be approximated by using the nearest limit point for some level of subdivision.)

   We do the same combination of least square fitting and fairing, and we subject our optimization to the global constraints that preserve the network subdivision curves and guarantee smoothness across curves and nodes.

Chapter 5

SUMMARY

In this dissertation, we presented our work in two areas of subdivision surfaces. We gave a theoretical analysis of the piecewise smooth subdivision scheme, and we described a practical algorithm that uses the scheme to reconstruct an actual object from non-uniformly sampled data.

5.1 Analysis

The primary goal of the analysis was to show that the piecewise smooth subdivision rules define surfaces that are G^1 at all smooth extraordinary points and up to creases and boundaries. It was also important to show that the limit points and tangent vectors to the surface were as expected at all points including along sharp edges and at sharp points.

To meet these goals, we used techniques that involve the eigenanalysis of matrices representing the subdivision rules, but also rely on the smoothness properties of the underlying (Box-spline) basis functions for the regular (non-extraordinary) regions of the surface. Although these techniques had been developed in the context of the symmetrical subdivision rules for smooth surfaces [1, 12, 27], we were able to extend them for use on our asymmetrical rules for creases and corners. In particular, we extended the sufficient conditions for a regular G^1 surface at an extraordinary (smooth, crease, or corner) point by allowing unequal sub-dominant eigenvalues and by adapting the characteristic maps of Reif [27] to the asymmetrical crease and corner neighborhoods.

To verify that these conditions are fulfilled for the smooth, crease, and corner rules, we derived formulas (for general valence n) for the eigenvalues and eigenvectors of their local subdivision matrices, and devised an algorithm that generates their characteristic map (for a particular n) and tests its properties.

The conclusions of this analysis were presented in Section 3.1 along with the rules for the various features of the piecewise smooth subdivision scheme. We can summarize the results by saying that the eigenvalue and eigenvector conditions were found to hold for all values of n for the smooth, crease, and corner rules, except for the $n = 3$ crease rule, which needs further investigation. Our tests of the characteristic map gave only positive results for consecutive values of n beyond practical valences for these rules.

From the eigenanalysis, we derived exact formulas for the limiting values of the mesh control vertices and for the surface normal vectors at those points where they exist. For the sharp vertices, we also used the eigenanalysis to find their limiting values and to verify the limiting tangent behavior. These formulas were also shown in Section 3.1.

While our analysis dealt with a specific scheme for piecewise smooth surfaces, we believe that the techniques we used could be applied to other subdivision schemes that introduce sharp features with rules that may be asymmetrical.

5.2 Application

The goal of the application was to show how piecewise smooth subdivision surfaces can be used in the reconstruction of an object from non-uniformly sampled data. We wished to accommodate the use of hand held scanners where the data samples would reflect the user's knowledge of the topology, geometry, and application of the object. In this context, we described the specific problem of creating a surface from data sampled uniformly along closed curves and non-uniformly within the regions they enclose. The collection of curves determine the topology of the surface.

Our strategy for solving this global problem entailed first designing an algorithm for creating a fair smooth surface of planar topology that interpolates a given closed boundary and fits a set of data on its interior. Many details in the solution of this sub-problem apply directly to the global problem. These include procedures to:

- Find control points for a subdivision curve that fits uniformly sampled data;
- Create a mesh structure to fill a region defined by its boundary of vertices;
- Optimize the placement of vertices in the mesh to create a smooth fair subdivision surface that interpolates the boundary vertices;
- Further optimize the vertex positions to fit the surface to the given interior data points while keeping a degree of fairness.

Our algorithm also addresses the problem that the surface will be smooth across the boundary of adjoining regions by creating a mesh structure with constraints that guarantee it. This structure is key to using the algorithm in a procedure that we have outlined to solve the global problem.

We believe our algorithm demonstrates the usefulness of subdivision surfaces for optimized surface fitting in a lost cost approach to object reconstruction. It also shows the advantage of the piecewise smooth scheme in fitting boundaries, including sharp corners, and in modeling smooth curves on the surface. By fitting non-uniformly sampled data points, it also provides a method for adjusting the surface shape within a closed boundary region.

BIBLIOGRAPHY

[1] A. A. Ball and J. T. Storry. A matrix approach to the analysis of recursively generated b-spline surfaces. *Computer-Aided Design*, 18:437–442, 1986.

[2] A. A. Ball and J. T. Storry. Conditions for tangent plane continuity over recursively defined b-spline surfaces. *ACM Transactions on Graphics*, 7(2):83–102, April 1988.

[3] R. H. Bartels, J. C. Beatty, and B. A. Barsky. *An Introduction to Splines for Use in Computer Graphics and Geometric Modeling*. Morgan Kaurmann, Los Altos, California, 1989.

[4] E. Catmull and J. Clark. Recursively generated B-spline surfaces on arbitrary topological meshes. *Computer-Aided Design*, 10:350–355, September 1978.

[5] A. S. Cavaretta, W. Dahmen, and C. A. Micchelli. Stationary subdivision. In *Memoirs of the AMS*, number 435 in 93. American Mathematical Society, september 1991.

[6] Philip J. Davis. *Circulant Matrices*. John Wiley and Sons, New York, 1979.

[7] C. DeBoor. *A Practical Guide to Splines*. Springer Verlag, New York, 1978.

[8] D. Doo and M. Sabin. Behaviour of recursive division surfaces near extraordinary points. *Computer-Aided Design*, 10(6):356–360, September 1978.

[9] Nira Dyn, David Levin, and John A. Gregory. A butterfly subdivision scheme for surface interpolation with tension control. *ACM Transactions on Graphics*, 9(2):160–169, April 1990.

[10] Gerald Farin. *Curves and Surfaces for Computer Aided Geometric Design*. Academic Press, 3rd edition, 1992.

[11] Gene Golub and Charles Van Loan. *Matrix Computations*. John Hopkins University Press, 2nd edition, 1989.

[12] Mark Halstead, Michael Kass, and Tony DeRose. Efficient, fair interpolation using Catmull-Clark surfaces. *Computer Graphics (SIGGRAPH '93 Proceedings)*, pages 35–44, August 1993.

[13] H. Hoppe, T. DeRose, T. Duchamp, M. Halstead, H. Jin, J. McDonald, J. Schweitzer, and W. Stuetzle. Piecewise smooth surface reconstruction. *Computer Graphics (SIGGRAPH '94 Proceedings)*, July 1994.

[14] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Surface reconstruction from unorganized points. *Computer Graphics (SIGGRAPH '92 Proceedings)*, 26(2):71–78, July 1992.

[15] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Mesh optimization. *Computer Graphics (SIGGRAPH '93 Proceedings)*, pages 19–26, August 1993.

[16] Hugues Hoppe. *Surface reconstruction from unorganized points*. PhD thesis, Department of Computer Science and Engineering, University of Washington, June 1994. (TR 94-06-01).

[17] Charles Loop. Smooth subdivision surfaces based on triangles. Master's thesis, Department of Mathematics, University of Utah, August 1987.

[18] J. Michael Lounsbery. *Multiresolution Analysis for Surfaces of Arbitrary Topological Type*. PhD thesis, Department of Computer Science and Engineering, University of Washington, 1994.

[19] Michael Lounsbery, Tony DeRose, and Joe Warren. Multiresolution surfaces of arbitrary topological type. *ACM Transactions on Graphics*, 1996. (To appear).

[20] Henry P. Moreton and Carlo Séquin. Functional optimization for fair surface design. *Computer Graphics (SIGGRAPH '92 Proceedings)*, 26(2):167–176, July 1992.

[21] Ahmad H. Nasri. Polyhedral subdivision methods for free-form surfaces. *ACM Transactions on Graphics*, 6(1):29–73, January 1987.

[22] Ahmad H. Nasri. Surface interpolation on irregular networks with normal conditions. *Computer Aided Geometric Design*, 8(1):89–96, 1991.

[23] Hartmut Prautzsch. Analysis of c^k-subdivision surfaces at extraordinary points. Presented at Oberwolfach, June, 1995.

[24] William H. Press, Brian P. Flannery, Saul A. Teukolsky, and William T. Fetterling. *Numerical Recipes.* Cambridge University Press, Cambridge, 1986.

[25] K. Pulli and M. Segal. Fast rendering of subdivision surfaces. In *Proceedings of 7th Eurographics Workshop on Rendering*, June 1996.

[26] Ulrich Reif. A degree estimate for subdivision surfaces of higher regularity. Mathematisches Institute A 94-12, Universität Stuttgart, 1994.

[27] Ulrich Reif. A unified approach to subdivision algorithms. *Computer Aided Geometric Design*, 12(2):153–174, 1995.

Appendix A

CREASE EDGE RULES AND LIMIT FORMULAS

In this appendix, we clarify the application of the crease edge rules and show formulas for the limit points and tangents for crease vertices with varying types of neighbors.

In Section 3.1.2, we presented the subdivision rules for vertices and edges on a crease. There are two types of vertices on a crease: ordinary, extraordinary (recall Figure 3.2). The end point of a crease is a corner, which we include in the category extraordinary crease vertex, or it is a dart, which is a smooth vertex.

The smooth edge rule is always applied to edges incident to a dart regardless of whether it is a crease edge or not. Figure 2.6 showed the smooth masks. The rules applied to all other crease edges only involve the crease vertices. Recall that there are two types of crease edge rules, *ordinary* and *special*. Their masks are 1-1 and 3-5 respectively. Table A.1 shows how the masks for these rules are applied.

In Section 3.1.2, we showed the limit masks for the ordinary and extraordinary crease vertices when their neighbors along the crease are both ordinary vertices. When one or both neighbors are extraordinary, a different mask applies. Table A.2 shows the six cases of these maps. They are calculated by first calculating the vertices after one subdivision and then applying the formulas given for ordinary neighbors.

Tangent vectors need adjustments as well. Table A.3 shows the masks for calculating the tangent vector along the crease at V_0. The mask depends on the type of neighbor crease vertices, V_1 and V_n. When the neighbors are of the same type, the tangent direction is always $V_1 - V_n$.

We note that a crease vertex that is a neighbor to a dart will have a different formula for its limit point and crease tangent vector from any that are shown in the table.

Table A.1: Applying subdivision masks to crease edge $V_1 - V_2$.

	V_1	–	V_2
Case 1	ord	–	ord
	1	–	1
Case 2	extra	–	extra
	1	–	1
Case 3	extra	–	ord
	3	–	5

Table A.2: Applying limit position masks on a crease at V_0 with crease neighbors V_1 and V_2.

	V_1	–	V_0	–	V_2
Case1	ord	–	ord	–	ord
	1	–	4	–	1
Case2	ord	–	extra	–	ord
	1	–	3	–	1
Case3	extra	–	ord	–	ord
	7	–	33	–	8
Case4	extra	–	ord	–	extra
	7	–	34	–	7
Case5	extra	–	extra	–	ord
	7	–	25	–	8
Case6	extra	–	extra	–	extra
	7	–	26	–	7

Table A.3: Limit tangent masks on a crease at V_0 with crease neighbors V_1 and V_2.

	V_1	–	V_0	–	V_2
Case1	ord	–	ord	–	ord
	1	–	0	–	-1
Case2	ord	–	extra	–	ord
	1	–	0	–	-1
Case3	extra	–	ord	–	ord
	3	–	1	–	-4
Case4	extra	–	ord	–	extra
	1	–	0	–	-1
Case5	extra	–	extra	–	ord
	4	–	1	–	-5
Case6	extra	–	extra	–	extra
	1	–	0	–	-1

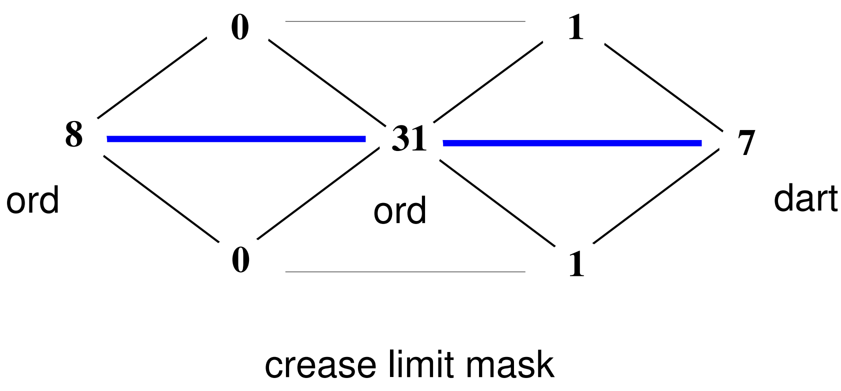

Figure A.1: The limit mask for an ordinary crease vertex with a dart neighbor and an ordinary crease vertex neighbor.

Figure A.1 shows the mask for the case where the crease vertex and its crease neighbor are both ordinary. There are three more cases that we do not show.

The cross crease tangent vectors also depend on the type of neighbor vertices on a crease. The formulas for these can be calculated by calculating the positions after one subdivision and then applying the formulas given in Section 3.1.2.

Appendix B

FORMULAS USED IN EIGENANALYSIS

In this appendix, we give derivations for some of the formulas used in Chapter 3.

B.1 Formulas for matrix A_n

We begin with a derivation of the characteristic polynomial for the symmetric banded n by n matrices A_n, where

$$A_n = (1/8)\begin{pmatrix} 3 & 1 & 0 & 0 & \cdots & \cdots & 0 \\ 1 & 3 & 1 & 0 & \cdots & \cdots & 0 \\ 0 & 1 & 3 & 1 & 0 & \cdots & 0 \\ . & . & . & . & . & . & . \\ 0 & 0 & \cdots & 0 & 1 & 3 & 1 \\ 0 & 0 & \cdots & \cdots & 0 & 1 & 3 \end{pmatrix}.$$

Expansion of the determinant, $det(A_n - \lambda I_n)$, by its top row leads to the recursion formula

$$P_{A_n}(\lambda) = (3/8 - \lambda)P_{A_{n-1}}(\lambda) - (1/64)P_{A_{n-2}}(\lambda). \tag{1}$$

Defining $P_{A_0}(\lambda) = 1$ and $P_{A_1}(\lambda) = (3/8 - \lambda)$ gives us the correct start.

We can solve this recursion to obtain

$$P_{A_n}(\lambda) = (-1/8)^n \sin((n+1)\theta)/\sin\theta, \tag{2}$$

for $0 \leq \theta < \pi$ and where $\lambda = (3 + 2\cos\theta)/8$. This can be verified by induction on n using well-known trigonometric identities.

We have the values of λ for which $P_{A_n}(\lambda) = 0$, by taking $\theta = (\pi j)/(n+1)$, since $\sin(\pi j) = 0$. This tells us that the eigenvalues of A_n are

$$\alpha_j = (3 + 2\cos\frac{\pi j}{n+1})/8, \tag{3}$$

for $j = 1, \ldots, n$.

A more general recursion formula is

$$P_{A_n}(\lambda) = P_{A_k}(\lambda)P_{A_{n-k}}(\lambda) - (1/64)P_{A_{k-1}}(\lambda)P_{A_{n-k-1}}(\lambda).$$

This formula becomes Equation (1) by taking $k = 1$. We can prove this formula by using induction on k. Alternatively, it can be verified using Equation (2) and trigonometric identities.

In the next section, we will use this formula by substituting $2n$ for n and n for k to get

$$P_{A_{2n}}(\lambda) = P^2_{A_n}(\lambda) - (1/8)^2 P^2_{A_{n-1}}(\lambda). \tag{4}$$

B.2 *Formulas for Matrix* C_n

We now derive formulas that relate the characteristic polynomials of C_n and A_n.

For $n \geq 3$, we will prove,

$$P_{C_n}(\lambda) \;=\; (3/8 - \lambda)P_{A_{n-1}}(\lambda) - 2(1/8)^2 P_{A_{n-2}}(\lambda) - 2(-1/8)^n, \tag{5}$$

or, equivalently, using Equation (1),

$$P_{C_n}(\lambda) \;=\; P_{A_n}(\lambda) - (1/8)^2 P_{A_{n-2}}(\lambda) - 2(-1/8)^n, \tag{6}$$

and for $n \geq 2$,

$$P_{C_{2n}}(\lambda) = (5/8 - \lambda)(1/8 - \lambda)P^2_{A_{n-1}}(\lambda) \tag{7}$$

and

$$P_{C_{2n-1}}(\lambda) = (5/8 - \lambda)P^2_{A_{n-1}}(\lambda). \tag{8}$$

We recall that

$$C_n = (1/8)\begin{pmatrix} 3 & 1 & 0 & \cdots & \cdots & 0 & 1 \\ 1 & 3 & 1 & 0 & \cdots & \cdots & 0 \\ 0 & 1 & 3 & 1 & 0 & \cdots & 0 \\ . & . & . & . & . & . & . \\ 0 & 0 & \cdots & 0 & 1 & 3 & 1 \\ 1 & 0 & \cdots & \cdots & 0 & 1 & 3 \end{pmatrix}.$$

The formula (A.5) is obtained by expanding the determinant $det(C_n - \lambda I_n)$ by its top row to get

$$P_{C_n}(\lambda) = (3/8 - \lambda)P_{A_{n-1}}(\lambda) - 2(1/8)det(D_{n-1}),$$

where

$$D_{n-1} = (1/8)\begin{pmatrix} 1 & 1 & 0 & \cdots\cdots & 0 \\ 0 & & & & \\ : & 8(A_{n-2} & - & \lambda I_{n-2}) & \\ 0 & & & & \\ 1 & & & & \end{pmatrix}.$$

Then, expanding further, we get

$$det(D_{n-1}) = (1/8)P_{A_{n-2}}(\lambda) - (-1/8)^{n-1}.$$

Combining these last equations gives us the required formula (A.5).

To derive the Equations (7) and (8), we will use induction on n. We show the proof for Equation (7) only; Equation (8) is similar.

It is easy to verify Equation (7) for $n = 2$. Then we note that from Equation (6) we have

$$P_{C_{2n}}(\lambda) = P_{A_{2n}}(\lambda) - (1/8)^2 P_{A_{2n-2}}(\lambda) - 2(1/8)^{2n}$$

and

$$(1/8)^2 P_{C_{2n-2}}(\lambda) = (1/8)^2 P_{A_{2n-2}}(\lambda) - (1/8)^4 P_{A_{2n-4}}(\lambda) - 2(1/8)^{2n}.$$

By subtracting these equations, we get

$$P_{C_{2n}}(\lambda) = (1/8)^2 P_{C_{2n-2}}(\lambda) + P_{A_{2n}}(\lambda) - 2(1/8)^2 P_{A_{2n-2}}(\lambda) + (1/8)^4 P_{A_{2n-4}(\lambda)}. \quad (9)$$

Now we use the induction hypothesis to replace $P_{C_{2n-2}}$ in this equation, and we employ Equation (4) to make substitutions for $P_{A_{2n}}, P_{A_{2n-2}}$ and $P_{A_{2n-4}}$ in Equation (9). By collecting the resulting terms and little use of Equation (1), we get Equation (7).

Since we know the eigenvalues for matrices A_n (Equation (3)), we can use Equations (7) and (8) to verify that the eigenvalues of C_n are given by

$$\gamma_j = (3 + 2\cos\frac{2\pi(j-1)}{n})/8,$$

for $j = 1, \ldots, n$.

B.3 Formula for Matrix $Smooth_n$

Finally, we can derive the formula for the characteristic polynomial for $Smooth_n$ in terms of the characteristic polynomial for the circulant matrix C_n. The resulting formula is:

$$(5/8 - \lambda) P_{Smooth_n}(\lambda) = (1-\lambda)((a-3)/8 - \lambda) P_{C_n}(\lambda).$$

We recall,

$$Smooth_n = (1/8)\begin{pmatrix} a & b & b & b & \cdots & \cdots & b & b \\ 3 & 3 & 1 & 0 & \cdots & \cdots & 0 & 1 \\ 3 & 1 & 3 & 1 & 0 & \cdots & \cdots & 0 \\ 3 & 0 & 1 & 3 & 1 & 0 & \cdots & 0 \\ . & . & . & . & . & . & . & . \\ 3 & 0 & \cdots & \cdots & 0 & 1 & 3 & 1 \\ 3 & 1 & 0 & \cdots & \cdots & 0 & 1 & 3 \end{pmatrix},$$

where $a = 8 - nb$.

We first expand the determinant $det(Smooth_n - \lambda I_{n+1})$ by its top row. I_{n+1} is the $n+1$ by $n+1$ identity matrix. We get

$$P_{Smooth_n} = (a/8 - \lambda)P_{C_n}(\lambda) - (3/8)(nb/8)(P_{A_{n-1}}(\lambda) - 2(1/8)det(B_{n-1}),$$

where

$$B_n = (1/8)\begin{pmatrix} 1 & 1 & 0 & \cdots\cdots & 0 \\ 1 & & & & \\ : & 8(A_{n-1} & - & \lambda I_{n-1}) & \\ 1 & & & & \end{pmatrix}.$$

We can expand $det(B_n)$ to get the recursion

$$det(B_n) = (1/8)(P_{A_{n-1}}(\lambda) - det(B_{n-1})).$$

We claim that for $n \geq 3$,

$$(5/8 - \lambda)det(B_n) = P_{A_n}(\lambda) + (1/8)P_{A_{n-1}}(\lambda) - (-1/8)^n. \qquad (10)$$

To prove the claim, we expand the recursion for $det(B_n)$ in the left-hand side of Equation (10), which we can rewrite as

$$det(B_n) + (3/8 - \lambda)det(B_n) + det(B_n).$$

In this expansion, we recall Equation (1) to associate terms

$$P_{A_{j+1}}(\lambda) - (3/8 - \lambda)P_{A_j}(\lambda) + (1/8)^2 P_{Aj-1}(\lambda) = 0,$$

for $j = 1, \ldots, n-2$. The remaining terms give us the right-hand side of Equation (10) to prove the claim.

Now we recall Equation (5),

$$P_{C_n}(\lambda) = (3/8 - \lambda)P_{A_{n-1}}(\lambda) - 2(1/8)^2 P_{A_{n-2}}(\lambda) - 2(-1/8)^n.$$

By applying Equation (10), we get

$$P_{C_n}(\lambda) = (5/8 - \lambda)(P_{A_{n-1}}(\lambda) - 2det(B_{n-1})).$$

We can now write the formula for $P_{Smoothn}(\lambda)$:

$$\begin{aligned}
(5/8-\lambda)P_{Smoothn} &= (5/8-\lambda)(a/8-\lambda)P_{Cn}(\lambda) \\
&\quad +(3/8)((a/8)-1)(5/8-\lambda)(P_{A_{n-1}}(\lambda)-2det(B_{n-1})) \\
&= ((5/8-\lambda)(a/8-\lambda)P_{Cn}(\lambda)+(3/8)((a/8)-1)P_{Cn}(\lambda) \\
&= (1-\lambda)((a-3)/8-\lambda)P_{Cn}(\lambda).
\end{aligned}$$

Appendix C

2-NEIGHBORHOOD SUBDIVISION MATRICES

In this appendix, we show some examples of the 2-neighborhood subdivision matrices derived and used in the analysis of the piecewise smooth subdivision rules.

In Section 3.4.1, we used the 2-neighborhood subdivision matrices in the construction of the characteristic maps for the smooth, crease, and corner rules. These matrices are extensions of the 1-neighborhood matrices that are presented in Section 3.3. Here, we show the 2-neighborhood matrices for selected valences. Where convenient, we give formulas with the valence represented by the variable n. We also give formulas for the right eigenvectors that correspond to the two sub-dominant eigenvalues.

C.1 Smooth Rule, valence 5

The 2-neighborhood for a smooth vertex has $3n+1$ vertices. Thus, for $n = 5$, the local subdivision matrix will have dimensions 16 by 16. Figure C.1 shows the numbering we use for the vertices in this neighborhood.

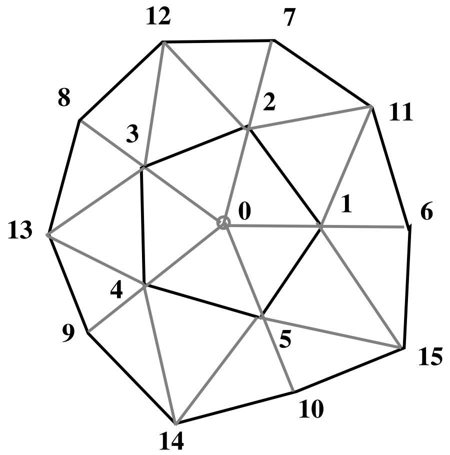

Figure C.1: 2-neighborhood for a smooth vertex of valence 5.

$$Smooth_m = (1/8)\begin{pmatrix}
a & b & b & b & b & b & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
3 & 3 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
3 & 1 & 3 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
3 & 0 & 1 & 3 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
3 & 0 & 0 & 1 & 3 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
3 & 1 & 0 & 0 & 1 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
x & 5 & x & 0 & 0 & x & x & 0 & 0 & 0 & 0 & x & 0 & 0 & 0 & x \\
x & x & 5 & x & 0 & 0 & 0 & x & 0 & 0 & 0 & x & x & 0 & 0 & 0 \\
x & 0 & x & 5 & x & 0 & 0 & 0 & x & 0 & 0 & 0 & x & x & 0 & 0 \\
x & 0 & 0 & x & 5 & x & 0 & 0 & 0 & x & 0 & 0 & 0 & x & x & 0 \\
x & x & 0 & 0 & x & 5 & 0 & 0 & 0 & 0 & x & 0 & 0 & 0 & x & x \\
1 & 3 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
1 & 0 & 3 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
1 & 0 & 0 & 3 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 3 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
1 & 0 & 0 & 0 & 0 & 3 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix},$$

where $a = 8 - 5b$, $b = (1/n)(5 - (3 + 2\cos(\frac{2\pi}{n}))^2/8)$, and $x = 1/2$.

The sub-dominant eigenvalues are $\lambda_1 = \lambda_2 = (3 + 2\cos(\frac{2\pi}{n}))/8$.

The corresponding right eigenvectors are:

$$\begin{aligned} R_1 &= (0, c_1, c_2, \ldots, c_n, yc_1, yc_2, \ldots, yc_n, \\ &\quad z(c_1 + c_2), z(c_2, +c_3), \ldots, z(c_n + c_1)), \\ R_2 &= (0, s_1, s_2, \ldots, s_n, ys_1, ys_2, \ldots, ys_n, \\ &\quad z(s_1 + s_2), z(s_2, +s_3), \ldots, z(s_n + s_1)), \end{aligned}$$

where, for $i = 1, \ldots, n$,

$$\begin{aligned} c_i &= \cos(\frac{2\pi(i-1)}{n}), \\ s_i &= \sin(\frac{2\pi(i-1)}{n}), \\ y &= (13 + 2\cos(\frac{2\pi}{n}))/(5 + 4\cos(\frac{2\pi}{n})), \\ z &= (3)/(2 + 2\cos(\frac{2\pi}{n})). \end{aligned}$$

For $n = 5$, these vectors are:

$$\begin{aligned} R_1 &= (0, 1., 0.3090, -0.8090, -0.8090, 0.3090, 2.1837, 0.6748, \\ &\quad -1.7666, -1.7666, 0.6748, 1.5, -0.5729, -1.8541, -0.5729, 1.5), \\ R_2 &= (0, 0, 0.9510, 0.5877, -0.5877, -0.9510, 0, 2.0768, \\ &\quad 1.2835, -1.2835, -2.0768, 1.0898, 1.7633, 0, -1.7633, -1.0898). \end{aligned}$$

C.2 Crease Rule, valence 5

The 2-neighborhood for a smooth vertex has $3n$ vertices. Thus, for $n = 5$, the local subdivision matrix will have dimensions 15 by 15. Figure C.2 shows the numbering we use for the vertices in this neighborhood.

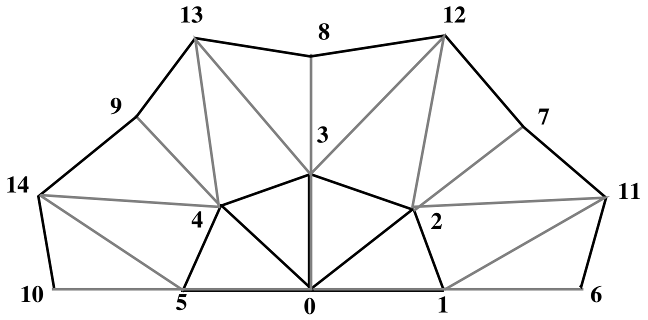

Figure C.2: 2-neighborhood for a crease vertex of valence 5.

$$
Crease_m = (1/8)\begin{pmatrix}
6 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
3 & 5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
3 & 1 & 3 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
3 & 0 & 1 & 3 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
3 & 0 & 0 & 1 & 3 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
3 & 0 & 0 & 0 & 0 & 5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 6 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
x & x & 5 & x & 0 & 0 & 0 & x & 0 & 0 & 0 & x & x & 0 & 0 \\
x & 0 & x & 5 & x & 0 & 0 & 0 & x & 0 & 0 & 0 & x & x & 0 \\
x & 0 & 0 & x & 5 & x & 0 & 0 & 0 & x & 0 & 0 & 0 & x & x \\
1 & 0 & 0 & 0 & 0 & 6 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
1 & 3 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
1 & 0 & 3 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 3 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
1 & 0 & 0 & 0 & 3 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix},
$$

where $x = 1/2$.

The sub-dominant eigenvalues are $\lambda_1 = 5/8$ and $\lambda_2 = (3 + 2\cos(\frac{\pi}{n-1}))/8$.

The corresponding right eigenvectors are:

$$R_1 = (0, x_1, x_2, \ldots, x_n, (3/2)x_1, (5/3)x_2, \ldots, (5/3)x_{n-1}, (3/2)x_n,$$

$$
\begin{aligned}
& (3/4)(x_1 + x_2), (3/4)(x_2, +x_3), \ldots, (3/4)(x_n + x_1)), \\
R_2 \;\; = \;\; & (0, s_1, s_2, \ldots, s_n, ys_1, ys_2, \ldots, ys_n, \\
& z(s_1 + s_2), z(s_2, +s_3), \ldots, z(s_n + s_1)),
\end{aligned}
$$

where, for $i = 1, \ldots, n$,

$$
\begin{aligned}
x_i &= (n + 1 - 2i)/(n - 1), \\
s_i &= \sin(\frac{\pi(i-1)}{n-1}), \\
y &= (13 + 2\cos(\frac{\pi}{n-1}))/(5 + 4\cos(\frac{\pi}{n-1})), \\
z &= (3)/(2 + 2\cos(\frac{\pi}{n-1})).
\end{aligned}
$$

For valence $n = 5$, we obtain:

$$
\begin{aligned}
R_1 &= (0, 1., 0.5, 0, -0.5, -1., 1.5, 0.8333, 0, -0.8333, -1.5, \\
&\quad 1.125, 0.375, -0.375, -1.125), \\
R_2 &= (0, 0, 0.7071, 1., 0.7071, 0, 0, 1.3019, 1.8412, 1.3019, 0, \\
&\quad 0.6213, 1.5, 1.5, 0.6213).
\end{aligned}
$$

C.3 Corner Rule, valence 4

The 2-neighborhood for a smooth vertex has $3n$ vertices. Thus, for $n = 4$, the local subdivision matrix will have dimensions 12 by 12. Figure C.3 shows the numbering we use for the vertices in this neighborhood.

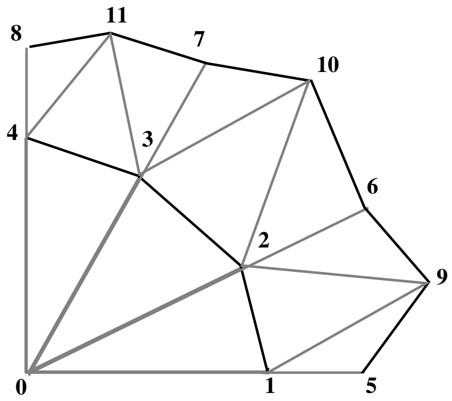

Figure C.3: 2-neighborhood for a corner vertex of valence 4.

$$Corner_m = (1/8)\begin{pmatrix} 8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 3 & 5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 3 & 1 & 3 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 3 & 0 & 1 & 3 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 3 & 0 & 0 & 0 & 5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 6 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ x & x & 5 & x & 0 & 0 & x & 0 & 0 & x & x & 0 \\ x & 0 & x & 5 & x & 0 & 0 & x & 0 & 0 & x & x \\ 1 & 0 & 0 & 0 & 6 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 3 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 3 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 3 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix},$$

where $x = 1/2$.

The sub-dominant eigenvalues are $\lambda_1 = \lambda_2 = 5/8$

The corresponding right eigenvectors are:

$$\begin{aligned} R_1 \;&=\; (0, n-1, n-2, \ldots, 1, 0, (3/2)(n-1), (5/3)(n-2), \ldots, (5/3), 0, \\ &\quad (3/4)(2n-3), (3/4)(2n-5), \ldots, (3/4)), \end{aligned}$$

$$
\begin{aligned}
R_2 &= (0, 0, 1, 2, \ldots, n-1, 0, (5/3), (5/3)(2), \ldots, (5/3)(n-2), (3/2)(n-1), \\
&\quad (3/4), (3/4)(3), (3/4)(5), \ldots, (3/4)(2n-3)).
\end{aligned}
$$

Thus, for $n = 4$, they are:

$$
\begin{aligned}
R_1 &= (0, 3, 2, 1, 0, (9/2), (10/3), (5/3), 0, (15/4), (9/4), (3/4)), \\
R_2 &= (0, 0, 1, 2, 3, 0, (5/3), (10/3), (9/2), (3/4), (9/4), (15/4)).
\end{aligned}
$$

Appendix D

THE JACOBIAN OF THE CHARACTERISTIC MAP

In Section 3.4.2, we discussed an algorithm for testing that the characteristic map is regular on each of the patches in the surface it defines. That algorithm proceeded by calculating the Bernstein coefficients $Coef(\mathcal{P})$ of the Jacobian of the map on each patch with control points $\mathcal{P} = (\mathcal{P}_\infty, \mathcal{P}_\in, \ldots, \mathcal{P}_{\infty\in})^\mathcal{T}$ and by checking that those coefficients are all positive in value. In this appendix, we show our derivation for the $Coef(\mathcal{P})$ and show how they might be evaluated.

The characteristic map can be represented piecewise as a quartic Box-spline function. We recall that the quartic Box-spline patch can be written in matrix form:

$$Patch(u, v, w) = Bern^{(4)}(u, v, w) \cdot Q \cdot \mathcal{P},$$

where $Bern^{(4)}(u, v, w)$ and Q are given in Section 2.2.2.

We wish to examine the Jacobian of $Patch(u, v, w)$ for each patch that the characteristic map defines. Since the map has values in $\mathbb{R}^2$, we can write it as $Patch(u, v, w) = (x(u, v, w), y(u, v, x))$.

Since $w = 1 - u - v$, we can think of $Patch(u, v, w)$ as a function of u and v. Then we can write the Jacobian as

$$J(u, v, w) = det \begin{pmatrix} \left(\frac{\partial x}{\partial u} - \frac{\partial x}{\partial w}\right) & \left(\frac{\partial y}{\partial u} - \frac{\partial y}{\partial w}\right) \\ \left(\frac{\partial x}{\partial v} - \frac{\partial x}{\partial w}\right) & \left(\frac{\partial y}{\partial v} - \frac{\partial y}{\partial w}\right) \end{pmatrix}.$$

We can write $\mathcal{P} = (X, Y)$, where X and Y are column vectors of the x and y coordinate of the control points in $\mathcal{P}$ for the patch.

Using the matrix notation, we have

$$\begin{aligned} J(u,v,w) = &\ 16\left\{(Bern^{(3)}\cdot Q_u\cdot X)(Bern^{(3)}\cdot Q_v\cdot Y)\right\} \\ - &\ 16\left\{(Bern^{(3)}\cdot Q_v\cdot X)(Bern^{(3)}\cdot Q_u\cdot Y)\right\}, \end{aligned}$$

where

$$4\ Bern^{(3)}\cdot Q_u = (\frac{\partial Bern^{(4)}}{\partial u} - \frac{\partial Bern^{(4)}}{\partial w})\cdot Q,$$
$$4\ Bern^{(3)}\cdot Q_v = (\frac{\partial Bern^{(4)}}{\partial v} - \frac{\partial Bern^{(4)}}{\partial w})\cdot Q,$$

and $Bern^{(3)}(u,v,w)$ is the array of 10 Bernstein coefficients of degree 3,

$$Bern^{(3)}(u,v,w) = (u^3, 3u^2v, 3u^2w, 3uv^2, 6uvw, 3uw^2, v^3, 3v^2w, 3vw^2, w^3).$$

This can be rewritten in the form

$$J(u,v,w) = 16\sum_{k=1}^{28} Bern_k^{(6)}(u,v,w)\cdot X^T\cdot M^{(k)}\cdot Y,$$

where $Bern^{(6)}(u,v,w)$ is the array of Bernstein polynomials of degree 6,

$$\begin{aligned} (u^6, 6u^5v, 6u^5w, 15u^4v^2, 30u^4vw, 15u^4w^2, 20u^3v^3, 60u^3v^2w, 60u^3vw^2, 20u^3w^3, \\ 15u^2v^4, 60u^2v^3w, 90u^2v^2w^2, 60u^2vw, 15u^2w^4, 6uv^5, 30uv^4w, 60uv^3w^2, \\ 60uv^2w^3, 30uvw^4, 6uw^5, v^6, 6v^5w, 15v^4w^2, 20v^3w^3, 15v^2w^4, 6vw^5, w^6) \end{aligned}$$

and each $M^{(k)}$ is a real 12×12 matrix.

Thus, we have written the Jacobian in Bernstein form where the coefficient of the k-th Bernstein monomial is given by the formula

$$Coef_k(X,Y) = 16\ X^T\cdot M^{(k)}\cdot Y.$$

Because each of the $M^{(k)}$ is an antisymmetric 12×12 matrix, we may compute $Coef_k(X,Y)$ as follows

$$\begin{aligned} X^T\cdot M^{(k)}\cdot Y &= \sum_j\left(\sum_i X_i M_{ij}^k\right)Y_j \\ &= \sum_{i>j} M_{ij}^k\,(X_iY_j - X_jY_i) \\ &= \sum_{i>j} M_{ij}^k\,\mathrm{d}_{i,j}, \end{aligned}$$

where $\mathrm{d}_{i,j}$ is the determinant of the two control points P_i and P_j.

We now proceed to show the matrices $M^{(k)}$ and the resulting expressions for the coefficients $Coef_k(X,Y)$. We will present 7 of them as representative cases because, under the symmetry operations of interchanging variables u, v and w, there are only 7 unique Bernstein monomials:

$$u^6,\ 6u^5v,\ 15u^4v^2,\ 30u^4vw,\ 20u^3v^3,\ 60u^3v^2w,\ 90u^2v^2w^2\,.$$

We thus present only the corresponding 7 matrices $M^{(k)}$ and coefficients $Coef_k(X,Y)$. In the following, we will show only the nonzero rows and columns of each $M^{(k)}$ and list the numbers of the points in $\mathcal{P}$ to which it is applied. The ordering of the control points for a Box-spline patch was shown in Figure 2.9 in Chapter 2.

Case 1

The coefficient for u^6 is

$$Coef_1(X,Y) = X^T \cdot M^{(1)} \cdot Y,$$

where

$$M^{(1)} = \begin{pmatrix} 0 & -1 & 1 & -1 & 1 & 0 \\ 1 & 0 & 1 & -1 & 0 & -1 \\ -1 & -1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & -1 & -1 \\ -1 & 0 & -1 & 1 & 0 & 1 \\ 0 & 1 & -1 & 1 & -1 & 0 \end{pmatrix}$$

is applied to the control points numbered: 1,2,4,6,8,9.

We can write $Coef_1(X,Y)$ as the following sum of determinants of these control points:

$$\begin{aligned} &\mathrm{d}_{2,1} + \mathrm{d}_{1,4} + \mathrm{d}_{6,1} + \mathrm{d}_{1,8} + \mathrm{d}_{2,4} + \mathrm{d}_{6,2} \\ &+ \mathrm{d}_{9,2} + \mathrm{d}_{4,8} + \mathrm{d}_{4,9} + \mathrm{d}_{8,6} + \mathrm{d}_{9,6} + \mathrm{d}_{8,9}. \end{aligned}$$

Case 2

The coefficient for $6u^5v$ is

$$Coef_2(X,Y) = X^T \cdot M^{(2)} \cdot Y,$$

where

$$M^{(2)} = \begin{pmatrix} 0 & -1 & 1 & 0 & -5 & 5 & 0 \\ 1 & 0 & 4 & 6 & -6 & 2 & -7 \\ -1 & -4 & 0 & -6 & -2 & 6 & 7 \\ 0 & -6 & 6 & 0 & -6 & 6 & 0 \\ 5 & 6 & 2 & 6 & 0 & -8 & -11 \\ -5 & -2 & -6 & -6 & 8 & 0 & 11 \\ 0 & 7 & -7 & 0 & 11 & -11 & 0 \end{pmatrix}$$

is applied to the control points numbered: 1,2,4,5,6,8,9.

We can write $Coef_2(X,Y)$ as the following sum of determinants of these control points:

$$\begin{gathered} 11\,\{\,\mathrm{d}_{9,6} + \,\mathrm{d}_{8,9}\} + 8\,\mathrm{d}_{8,6} + 7\,\{\,\mathrm{d}_{4,9} + \,\mathrm{d}_{9,2}\} \\ +6\,\{\,\mathrm{d}_{6,2} + \,\mathrm{d}_{2,5} + \,\mathrm{d}_{5,4} + \,\mathrm{d}_{4,8} + \,\mathrm{d}_{6,5} + \,\mathrm{d}_{5,8}\} + 5\,\{\,\mathrm{d}_{6,1} + \,\mathrm{d}_{1,8}\} \\ +4\,\mathrm{d}_{2,4} + 2\,\{\,\mathrm{d}_{2,8} + \,\mathrm{d}_{6,4}\} + \,\mathrm{d}_{1,4} + \,\mathrm{d}_{2,1} \end{gathered}$$

Case 3

The coefficient for $15u^4v^2$ is

$$Coef_4(X,Y) = X^T \cdot M^{(4)} \cdot Y,$$

where

$$M^{(4)} = \begin{pmatrix} 0 & 0 & 0 & 0 & -5 & 5 & 0 & -1 & 1 \\ 0 & 0 & 3 & 12 & -10 & 10 & -12 & -2 & -1 \\ 0 & -3 & 0 & -12 & -10 & 10 & 12 & 1 & 2 \\ 0 & -12 & 12 & 0 & -36 & 36 & 0 & 0 & 0 \\ 5 & 10 & 10 & 36 & 0 & -17 & -41 & -1 & -2 \\ -5 & -10 & -10 & -36 & 17 & 0 & 41 & 2 & 1 \\ 0 & 12 & -12 & 0 & 41 & -41 & 0 & 1 & -1 \\ 1 & 2 & -1 & 0 & 1 & -2 & -1 & 0 & 0 \\ -1 & 1 & -2 & 0 & 2 & -1 & 1 & 0 & 0 \end{pmatrix}$$

is applied to the control points numbered: 1,2,4,5,6,8,9,10,11.

We can write $Coef_4(X,Y)$ as the following sum of determinants of these control points:

$$\begin{gathered} 41\,\{\,\mathrm{d}_{8,9} + \mathrm{d}_{9,6}\} + 36\,\{\,\mathrm{d}_{6,5} + \mathrm{d}_{5,8}\} + 17\,\mathrm{d}_{8,6} \\ +12\,\{\,\mathrm{d}_{9,2} + \mathrm{d}_{2,5} + \mathrm{d}_{5,4} + \mathrm{d}_{4,9}\} + 10\,\{\,\mathrm{d}_{6,2} + \mathrm{d}_{2,8} + \mathrm{d}_{6,4} + \mathrm{d}_{4,8}\} \\ +5\,\{\,\mathrm{d}_{6,1} + \mathrm{d}_{1,8}\} + 3\,\mathrm{d}_{2,4} + 2\,\{\,\mathrm{d}_{4,11} + \mathrm{d}_{11,6} + \mathrm{d}_{8,10} + \mathrm{d}_{10,2}\} \\ +\,\mathrm{d}_{4,10} + \mathrm{d}_{10,1} + \mathrm{d}_{1,11} + \mathrm{d}_{11,2} + \mathrm{d}_{8,11} + \mathrm{d}_{11,9} + \mathrm{d}_{9,10} + \mathrm{d}_{10,6} \end{gathered}$$

Case 4

The coefficient for $30u^4vw$ is

$$Coef_5(X,Y) = X^T \cdot M^{(5)} \cdot Y,$$

where

$$M^{(5)} = \begin{pmatrix} 0 & -5 & 0 & 2 & -22 & 13 & 14 & -2 \\ 5 & 0 & 13 & 52 & -45 & 16 & -37 & -4 \\ 0 & -13 & 0 & -2 & -14 & 5 & 22 & 2 \\ -2 & -52 & 2 & 0 & -50 & 52 & 50 & 0 \\ 22 & 45 & 14 & 50 & 0 & -37 & -92 & -2 \\ -13 & -16 & -5 & -52 & 37 & 0 & 45 & 4 \\ -14 & 37 & -22 & -50 & 92 & -45 & 0 & 2 \\ 2 & 4 & -2 & 0 & 2 & -4 & -2 & 0 \end{pmatrix}$$

is applied to the control points numbered: 1,2,4,5,6,8,9,10.

We can write $Coef_5(X,Y)$ as the following sum of determinants of these control points:

$$\begin{gathered} 92\,\mathrm{d}_{9,6} + 52\,\{\,\mathrm{d}_{2,5} + \,\mathrm{d}_{5,8}\} + 50\,\{\,\mathrm{d}_{6,5} + \,\mathrm{d}_{5,9}\} \\ +45\,\{\,\mathrm{d}_{8,9} + \,\mathrm{d}_{6,2}\} + 37\,\{\,\mathrm{d}_{8,6} + \,\mathrm{d}_{9,2}\} \\ +22\,\{\,\mathrm{d}_{6,1} + \,\mathrm{d}_{4,9}\} + 16\,\mathrm{d}_{2,8} + 14\,\{\,\mathrm{d}_{1,9} + \,\mathrm{d}_{6,4}\} \\ +13\,\{\,\mathrm{d}_{1,8} + \,\mathrm{d}_{2,4}\} + 5\,\{\,\mathrm{d}_{4,8} + \,\mathrm{d}_{2,1}\} \\ +4\,\{\,\mathrm{d}_{8,10} + \,\mathrm{d}_{10,2}\} + 2\,\{\,\mathrm{d}_{9,10} + \,\mathrm{d}_{10,1} + \,\mathrm{d}_{1,5} + \,\mathrm{d}_{5,4} + \,\mathrm{d}_{4,10} + \,\mathrm{d}_{10,6}\} \end{gathered}$$

Case 5

The coefficient for $20u^3v^3$ is

$$Coef_7(X,Y) = X^T \cdot M^{(7)} \cdot Y,$$

where

$$M^{(7)} = \begin{pmatrix} 0 & 0 & 0 & 0 & -1 & 1 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & 7 & -6 & 10 & -6 & -4 & 0 & -1 \\ 0 & 0 & 0 & -7 & -10 & 6 & 6 & 0 & 4 & 1 \\ 0 & -7 & 7 & 0 & -61 & 61 & 0 & -6 & 6 & 0 \\ 1 & 6 & 10 & 61 & 0 & 0 & -61 & -6 & -10 & -1 \\ -1 & -10 & -6 & -61 & 0 & 0 & 61 & 10 & 6 & 1 \\ 0 & 6 & -6 & 0 & 61 & -61 & 0 & 7 & -7 & 0 \\ 1 & 4 & 0 & 6 & 6 & -10 & -7 & 0 & 0 & 0 \\ -1 & 0 & -4 & -6 & 10 & -6 & 7 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

is applied to the control points numbered: 1,2,4,5,6,8,9,10,11,12.

We can write $Coef_7(X, Y)$ as the following sum of determinants of these control points:

$$\begin{gathered} 61\{\mathrm{d}_{6,5} + \mathrm{d}_{5,8} + \mathrm{d}_{9,6} + \mathrm{d}_{8,9}\} + 10\{\mathrm{d}_{6,4} + \mathrm{d}_{2,8} + \mathrm{d}_{8,10} + \mathrm{d}_{11,6}\} \\ +7\{\mathrm{d}_{5,4} + \mathrm{d}_{2,5} + \mathrm{d}_{11,9} + \mathrm{d}_{9,10}\} + 4\{\mathrm{d}_{10,2} + \mathrm{d}_{4,11}\} \\ +6\{\mathrm{d}_{10,5} + \mathrm{d}_{5,11} + \mathrm{d}_{8,11} + \mathrm{d}_{10,6} + \mathrm{d}_{6,2} + \mathrm{d}_{4,8} + \mathrm{d}_{9,2} + \mathrm{d}_{4,9}\} \\ +\mathrm{d}_{10,1} + \mathrm{d}_{1,11} + \mathrm{d}_{6,1} + \mathrm{d}_{1,8} + \mathrm{d}_{12,2} + \mathrm{d}_{4,12} + \mathrm{d}_{12,6} + \mathrm{d}_{8,12} \end{gathered}$$

Case 6

The coefficient for $60u^3v^2w$ is

$$Coef_8(X, Y) = X^T \cdot M^{(8)} \cdot Y,$$

where

$$M^{(8)} = \begin{pmatrix} 0 & 0 & 0 & -5 & -15 & 8 & 12 & -4 & 3 & 1 \\ 0 & 0 & 6 & 65 & -51 & 40 & -36 & -20 & -3 & -1 \\ 0 & -6 & 0 & -4 & -30 & 7 & 30 & 1 & 0 & 2 \\ 5 & -65 & 4 & 0 & -208 & 119 & 139 & -6 & 12 & 0 \\ 15 & 51 & 3 & 208 & 0 & -34 & -237 & -19 & -12 & -2 \\ -8 & -40 & -7 & -119 & 34 & 0 & 110 & 26 & 3 & 1 \\ -12 & 36 & -30 & -139 & 237 & -110 & 0 & 22 & -3 & -1 \\ 4 & 20 & -1 & 6 & 19 & -26 & -22 & 0 & 0 & 0 \\ -3 & 3 & 0 & -12 & 12 & -3 & 3 & 0 & 0 & 0 \\ -1 & 1 & -2 & 0 & 2 & -1 & 1 & 0 & 0 & 0 \end{pmatrix}$$

is applied to the control points numbered: 1,2,4,5,6,8,9,10,11,12.

We can write $Coef_8(X,Y)$ as the following sum of determinants of these control points:

$$\begin{gathered} 237\,\mathrm{d}_{9,6} + 208\,\mathrm{d}_{6,5} + 139\,\mathrm{d}_{5,9} + 119\,\mathrm{d}_{5,8} + 110\,\mathrm{d}_{8,9} \\ +65\,\mathrm{d}_{2,5} + 51\,\mathrm{d}_{6,2} + 40\,\mathrm{d}_{2,8} + 36\,\mathrm{d}_{9,2} + 34\,\mathrm{d}_{8,6} \\ +30\,\{\,\mathrm{d}_{6,4} + \,\mathrm{d}_{4,9}\} + 26\,\mathrm{d}_{8,10} + 22\,\mathrm{d}_{9,10} + 20\,\mathrm{d}_{10,2} + 19\,\mathrm{d}_{10,6} \\ +15\,\mathrm{d}_{6,1} + 12\,\mathrm{d}_{1,9} + 12\,\mathrm{d}_{5,11} + 12\,\mathrm{d}_{11,6} + 8\,\mathrm{d}_{1,8} + 7\,\mathrm{d}_{4,8} + 6\,\mathrm{d}_{2,4} \\ +6\,\mathrm{d}_{10,5} + 5\,\mathrm{d}_{5,1} + 4\,\mathrm{d}_{5,4} + 4\,\mathrm{d}_{10,1} + 3\,\{\,\mathrm{d}_{8,11} + \,\mathrm{d}_{11,9} + \,\mathrm{d}_{1,11} + \,\mathrm{d}_{11,2}\} \\ +2\,\{\,\mathrm{d}_{4,12} + \,\mathrm{d}_{12,6}\} + \,\mathrm{d}_{4,10} + \,\mathrm{d}_{1,12} + \,\mathrm{d}_{12,2} + \,\mathrm{d}_{8,12} + \,\mathrm{d}_{12,9} \end{gathered}$$

Case 7

The coefficient for $90u^2v^2w^2$ is

$$Coef_{13}(X,Y) = X^T \cdot M^{(13)} \cdot Y,$$

where

$$M^{(13)} = \begin{pmatrix}
0 & 0 & 0 & 0 & -1 & -4 & 0 & 1 & 4 & -1 & 0 & 1 \\
0 & 0 & 0 & 1 & 30 & -30 & -1 & 16 & 0 & -16 & 1 & -1 \\
0 & 0 & 0 & 0 & 4 & 1 & 0 & 1 & -4 & -1 & -1 & 0 \\
0 & -1 & 0 & 0 & 1 & -4 & -1 & 0 & 4 & 1 & 0 & 0 \\
1 & -30 & -4 & -1 & 0 & -119 & -4 & 30 & 119 & 0 & 4 & 4 \\
4 & 30 & -1 & 4 & 119 & 0 & -1 & 0 & -119 & -30 & -4 & -4 \\
0 & 1 & 0 & 1 & 4 & -1 & 0 & -1 & -4 & 0 & 0 & 0 \\
-1 & -16 & -1 & 0 & -30 & 0 & 1 & 0 & 30 & 16 & 0 & 1 \\
-4 & 0 & 4 & -4 & -119 & 119 & 4 & -30 & 0 & 30 & 1 & -1 \\
1 & 16 & 1 & -1 & 0 & 30 & 0 & -16 & -30 & 0 & -1 & 0 \\
0 & -1 & 1 & 0 & -4 & 4 & 0 & 0 & -1 & 1 & 0 & 0 \\
-1 & 1 & 0 & 0 & -4 & 4 & 0 & -1 & 1 & 0 & 0 & 0
\end{pmatrix}$$

is applied to the control points numbered: 1,2,4,5,6,8,9,10,11,12.

We can write $Coef_{13}(X,Y)$ as the following sum of determinants of these control points:

$$\begin{gathered}
119\,\{\,\mathrm{d}_{6,5} + \mathrm{d}_{5,9} + \mathrm{d}_{9,6}\} \\
+30\,\{\,\mathrm{d}_{9,10} + \mathrm{d}_{8,9} + \mathrm{d}_{10,6} + \mathrm{d}_{6,2} + \mathrm{d}_{2,5} + \mathrm{d}_{5,8}\} \\
+16\,\{\,\mathrm{d}_{2,8} + \mathrm{d}_{10,2} + \mathrm{d}_{8,10}\} \\
+4\,\{\,\mathrm{d}_{6,1} + \mathrm{d}_{1,9} + \mathrm{d}_{3,5} + \mathrm{d}_{9,3} + \mathrm{d}_{6,4} + \mathrm{d}_{4,9}\} \\
+4\,\{\,\mathrm{d}_{7,5} + \mathrm{d}_{5,11} + \mathrm{d}_{11,6} + \mathrm{d}_{9,7} + \mathrm{d}_{12,6} + \mathrm{d}_{5,12}\} \\
+\,\mathrm{d}_{5,1} + \mathrm{d}_{6,7} + \mathrm{d}_{9,11} + \mathrm{d}_{4,5} + \mathrm{d}_{3,6} + \mathrm{d}_{12,9} + \mathrm{d}_{1,8} + \mathrm{d}_{2,4} \\
+\,\mathrm{d}_{7,2} + \mathrm{d}_{10,3} + \mathrm{d}_{11,10} + \mathrm{d}_{8,12} + \mathrm{d}_{8,7} + \mathrm{d}_{3,8} + \mathrm{d}_{7,4} \\
+\,\mathrm{d}_{11,3} + \mathrm{d}_{10,1} + \mathrm{d}_{12,2} + \mathrm{d}_{1,12} + \mathrm{d}_{2,11} + \mathrm{d}_{4,10}
\end{gathered}$$

Appendix E

ENERGY MATRICES

E.1 Energy Matrix for an Ordinary Patch

The symmetric 12 by 12 matrix K_{ord}, described in Section 4.5.2, is shown here. The elements are given up to a common factor.

$K_{ord} =$

$$\begin{pmatrix}
502 & 237 & 17 & 158 & -759 & -801 & 79 & -138 & 45 & 471 & 79 & 110 \\
237 & 1974 & 237 & -138 & -1839 & -1839 & -138 & 333 & -102 & 333 & 471 & 471 \\
17 & 237 & 502 & 79 & -801 & -759 & 158 & 471 & 45 & -138 & 110 & 79 \\
158 & -138 & 79 & 502 & -759 & 45 & 110 & 237 & -801 & 471 & 17 & 79 \\
-759 & -1839 & -801 & -759 & 6702 & 54 & 45 & -1839 & 54 & -102 & -801 & 45 \\
-801 & -1839 & -759 & 45 & 54 & 6702 & -759 & -102 & 54 & -1839 & 45 & -801 \\
79 & -138 & 158 & 110 & 45 & -759 & 502 & 471 & -801 & 237 & 79 & 17 \\
-138 & 333 & 471 & 237 & -1839 & -102 & 471 & 1974 & -1839 & 333 & 237 & -138 \\
45 & -102 & 45 & -801 & 54 & 54 & -801 & -1839 & 6702 & -1839 & -759 & -759 \\
471 & 333 & -138 & 471 & -102 & -1839 & 237 & 333 & -1839 & 1974 & -138 & 237 \\
79 & 471 & 110 & 17 & -801 & 45 & 79 & 237 & -759 & -138 & 502 & 158 \\
110 & 471 & 79 & 79 & 45 & -801 & 17 & -138 & -759 & 237 & 158 & 502
\end{pmatrix}$$

E.2 Energy Matrix for a Valence 5 Extraordinary Patch

The symmetric 11 by 11 matrix K_5, described in Section 4.5.2, is shown here. The elements are given up to a common factor and rounded to the nearest integer.

$K_5 =$

$$\begin{pmatrix}
507 & 237 & 17 & 158 & -739 & -781 & 79 & -116 & 5 & 492 & 146 \\
237 & 1974 & 237 & -138 & -1761 & -1761 & -138 & 427 & -233 & 427 & 729 \\
17 & 237 & 502 & 79 & -781 & -739 & 158 & 492 & 5 & -117 & 146 \\
158 & -138 & 79 & 502 & -762 & 41 & 110 & 241 & -790 & 469 & 91 \\
-739 & -1761 & -781 & -762 & 6809 & 43 & 41 & -1897 & 326 & -449 & -831 \\
-781 & -1761 & -739 & 41 & 43 & 6809 & -762 & -449 & 326 & -1897 & -831 \\
79 & -138 & 158 & 110 & 41 & -762 & 502 & 469 & -790 & 241 & 91 \\
-116 & 427 & 492 & 241 & -1897 & -449 & 469 & 1986 & -1719 & 136 & 432 \\
5 & -233 & 5 & -790 & 326 & 326 & -790 & -1719 & 5927 & -1719 & -1340 \\
492 & 427 & -116 & 469 & -449 & -1897 & 241 & 135 & -1719 & 1986 & 433 \\
145 & 729 & 146 & 91 & -831 & -831 & 91 & 433 & -1340 & 433 & 934
\end{pmatrix}$$

E.3 Energy Matrix for a Valence 7 Extraordinary Patch

The symmetric 13 by 13 matrix K_7, described in Section 4.5.2, is shown here. The elements are given up to a common factor and rounded to the nearest integer.

$K_7 =$

$$\begin{pmatrix}
502 & 237 & 17 & 158 & -772 & -814 & 79 & -151 & 73 & 458 & 65 & 97 & 51 \\
237 & 1974 & 237 & -138 & -1889 & -1889 & -138 & 281 & -5 & 281 & 404 & 404 & 240 \\
17 & 237 & 502 & 79 & -814 & -772 & 158 & 458 & 73 & -151 & 97 & 65 & 51 \\
158 & -138 & 79 & 502 & -757 & 47 & 110 & 239 & -808 & 473 & 11 & 79 & 4 \\
-772 & -1889 & -814 & -756 & 6669 & 69 & 47 & -1721 & -294 & 119 & -797 & 250 & -111 \\
-814 & -1889 & -772 & 47 & 69 & 6669 & -757 & 119 & -294 & -1721 & 250 & -797 & -111 \\
79 & -138 & 158 & 110 & 47 & -757 & 502 & 473 & -808 & 239 & 79 & 11 & 4 \\
-151 & 281 & 458 & 239 & -1721 & 119 & 473 & 2133 & -2127 & 605 & 243 & -164 & -388 \\
73 & -5 & 73 & -808 & -294 & -294 & -808 & -2127 & 7464 & -2127 & -613 & -613 & 77 \\
458 & 281 & -151 & 473 & 119 & -1721 & 239 & 605 & -2125 & 2133 & -164 & 243 & -388 \\
65 & 404 & 97 & 11 & -797 & 250 & 79 & 243 & -613 & -164 & 501 & -188 & 111 \\
97 & 404 & 65 & 79 & 250 & -797 & 11 & -164 & -613 & 243 & -188 & 501 & 111 \\
51 & 240 & 51 & 4 & -111 & -111 & 4 & -388 & 77 & -388 & 111 & 111 & 351
\end{pmatrix}$$

VITA

Jean E. Schweitzer was born in Evanston, Illinois. In 1968, she received her B.A. from Northwestern University, majoring in Mathematics. In 1971, she received an M.A. in Mathematics from the University of Illinois at Urbana, and in 1978 an M.S. in Computer Science from Washington State University at Pullman. She received an M.S. in Computer Science from the University of Washington in 1992, and completed her Ph.D. there in 1996.