Modeling Surfaces of Arbitrary Topology Using Manifolds

by
Cindy Marie Grimm
B.A. University of California, Berkeley, 1990
M.S. Brown University, 1992

Thesis
Submitted in partial fulfillment of the requirements for the
Degree of Doctor of Philosophy in the Department of Computer Science
at Brown University

May, 1996

This dissertation by Cindy Marie Grimm
is accepted in its present form
by the Department of Computer Science
as satisfying the dissertation requirement
for the degree of Doctor of Philosophy.

Date 2/26/96 [illegible]
John Forbes Hughes

Recommended to the Graduate Council

Date 2/26/96 [illegible]
Andries van Dam

Date 3/22/96 [illegible]
Anthony D. DeRose
Pixar Animation Studios, Richmond, Ca

Approved by the Graduate Council

Date 3/20/96 [illegible]

Vita

I was born July 13, 1967 in Stanford, California. I attended the University of California at Berkeley and received a Bachelors of Art in both Art and Computer Science in 1990. I received my Master of Science in Computer Science from Brown University in 1992. While at Brown I published the following three papers:

"Smooth Isosurface Approximation", Cindy Grimm and John Hughes, Eurographics Workshop on Implicit Surfaces, April 1995

"Modeling Surfaces of Arbitrary Topology Using Manifolds", Cindy Grimm and John Hughes, Siggraph 1995

"Visual Interfaces for Solids Modelling", Cindy Grimm, David Pugmire, Mark Bloomenthal, John Hughes, and Elaine Cohen, UIST 1995

Acknowledgements

I am grateful to the Brown graphics group for all their help and support, and the Hapkido club for keeping me sane, especially my fellow instructor Mark Abbott. Thank you to Michael Kowalski for building Alexander's two-holed torus and Daniel Robbins for making the original teapot. My appreciation to the folks at Stanford for letting me use their laser-scanned bunny. Special thanks to Professor Marc Levoy at Stanford for his many insightful suggestions.

Thanks to the following people for reading various versions of the thesis, and for all their helpful comments: Katrina Avery, William Smart, Tony DeRose, Andy van Damm, and of course, John Hughes.

And finally, many thanks to William Smart for keeping me well-supplied with jelly beans and beer.

Contents

List of Figures

Chapter 1

Introduction

Modeling complicated surfaces is a difficult and time-consuming task. In this paper we present a technique for both analyzing and building surfaces whose topology and geometry are complicated. The aim of this research is to provide a constructive approach for creating and editing such surfaces using a minimum of data. Additionally, the final surface should resemble the data in a reasonable way. We approach the problem by defining techniques which are designed specifically for analyzing and building complicated surfaces.

The desire to model complicated surfaces is not new; the CAD/CAM world in particular has developed a variety of approaches to the problem. We briefly describe two of these approaches in order to provide some intuition about why they may not be appropriate for designing free-form shapes such as the flower in Figure 1.1 or topologically complicated models such as the exhaust manifold in Figure 1.2.

CAD/CAM has focused on building surfaces using techniques based on traditional mechanical design. There are two reasons for this: the language is well developed and familiar to designers and the types of objects that can be machined are limited by existing tools, such as lathes and drills. CAD/CAM systems generally take one of two approaches. Solids modeling begins with solid primitives, such as cubes and spheres, then adds and subtracts these primitives from one another to construct more complicated shapes [Man88][Hof89]. This technique is suitable for a wide class of objects, especially many man-made objects. The second approach is spline-based. Surfaces are built up from curves using operators such as the *sweep* operator [BR91], an example of which is shown in Figure 1.3. These surfaces are very free-form, although their topology is essentially rectangular.[1] This approach has many parallels with the traditional approach of describing a surface by drawing what it looks like from three or more sides.

[1] An object's topology is rectangular if it can be built from a stretchy, rectangular sheet without tearing holes in it, creasing it, or compressing an edge to a point. For example, cylinders and tori are essentially rectangular, but a two-holed torus is not.

Figure 1.1: A sculpture of a flower.

Figure 1.2: An exhaust manifold.

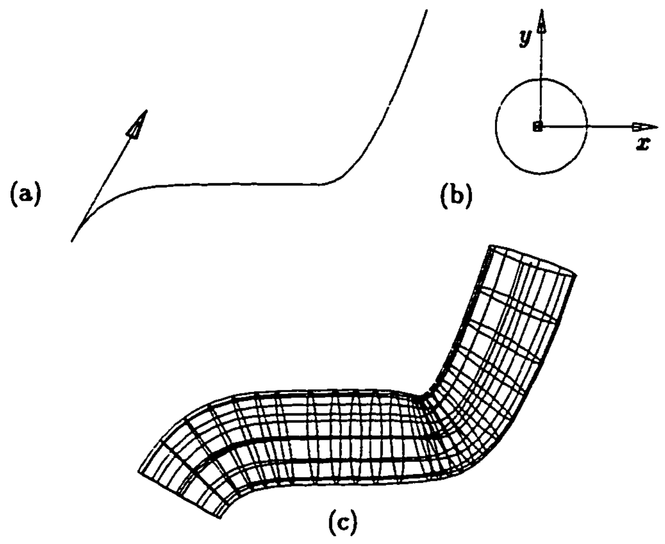


Figure 1.3: (a) An axis curve and tangent vector. (b) The cross-section curve with x and y axes shown; the z axis is out of the paper. (c) The resulting sweep surface: the z axis of the cross-section is always oriented in the direction of the axis curve's tangent vector.

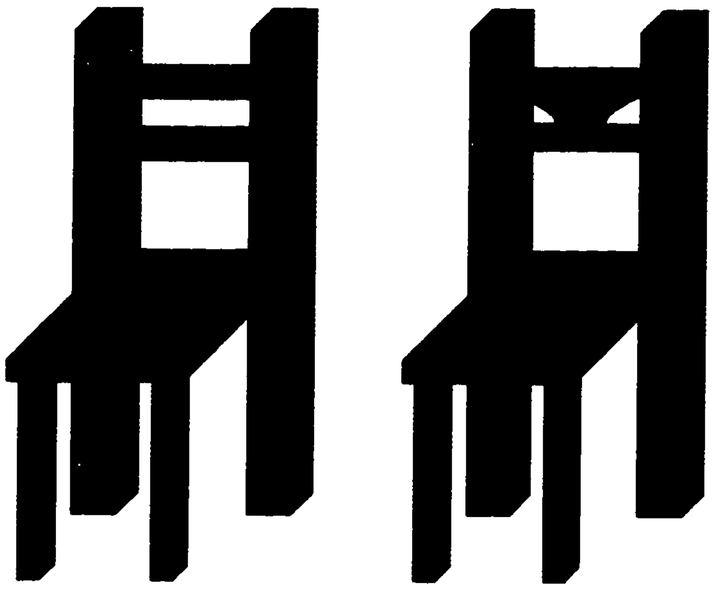


Figure 1.4: Two different chairs.

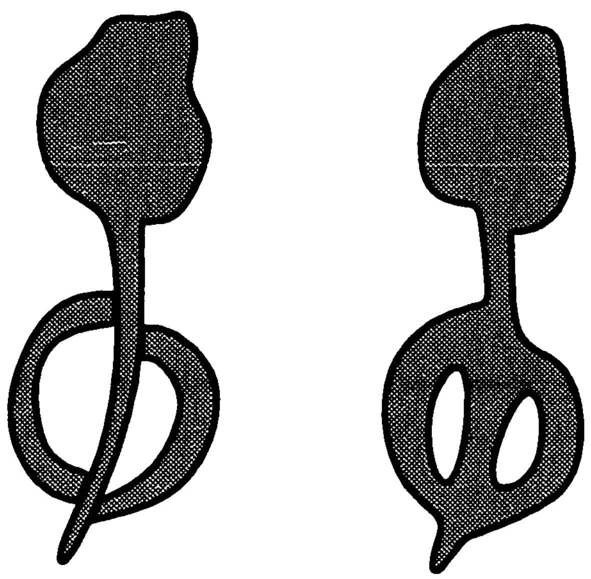


Figure 1.5: Two ways to make the body of the flower. Left: A sweep with two sweeps attatched. Right: A sweep with two holes cut in it.

Both of these approaches rely on a fairly good understanding of how basic operations, such as subtracting a cylinder from a cube, can be combined to produce the desired shape. The model is generally broken down into subparts that each consist of a sequence of operations. Deciding which subpart hierarchy to use and the order of operations within those subparts can be difficult. A major part of a good design is structuring the model with as much flexibility as possible, which often requires fairly specific knowledge of the desired surface. This is because a small change in topology can often require restructuring an entire model. For example, the two chairs in Figure 1.4 look very similar but changing from one to the other is difficult and much of the geometric detail cannot be carried over. There are, of course, benefits to organizing a model using CAD/CAM techniques; in the chair example, changing certain aspects of design, such as the height of the chair, the curve of the seat, etc., requires the change of only a single value or curve. Parameter and feature-based design systems exploit this by allowing designers to create explicitly parameterized models in which changes can be propagated through the model automatically.

Sometimes, however, the designer or artist wants the freedom to change the model quickly and drastically and sometimes the desired surface does not break down easily into a series of spline or solids modeling operations. For example, the flower in Figure 1.1 can be thought of as a sweep with two sweeps attached to it, or as a single sweep with two holes cut out of it (see Figure 1.5). While a good user interface may simplify the specification task, it is still restricted by the underlying modeling techniques. To address these problems we present a general method for blending simple surfaces together into complicated topologies. A complicated surface is

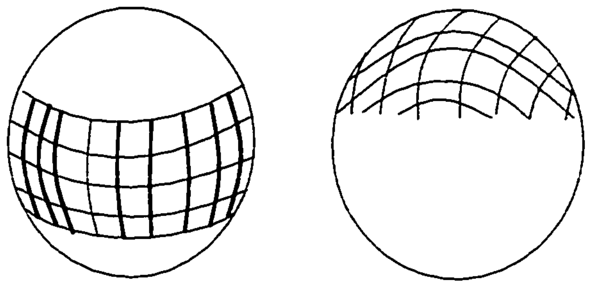


Figure 1.6: Left: Parameterizing the sphere with latitude and longitude lines. Right: Using a different parameterization for the north pole.

built by saying "it looks like this over here and that over there...;" the blending between these requirements takes place automatically. This is essentially what splines do for rectangular topologies, and makes it very simple to alter the surface without redesigning or rethinking the entire surface.

Another problem addressed in this paper is parameterization. It is always possible to find a parameterization for a complicated surface but it is not always an ideal one. For example, a sphere parameterized by the standard latitude and longitude method behaves well around most of the equator but near the poles the latitude lines converge to a point and along the date line there is a "seam." Any technique, such as parameter-space based texture mapping, that uses this parameterization will have difficulty at the poles and at the date line. The solution presented here provides a local parameterization at every point that "behaves well" and a method for moving from one local parameterization to the next.

Although in this paper we focus on building interesting models for computer graphics, these techniques are also applicable to problems arising from the representation of data whose domain is locally planar. For example, a bidirectional reflectance function [CW93] can be represented as a function on a sphere. This task is simplified by representing the function in "bits," each of which represents what the function looks like over part of the sphere. The bits do not have to be the same size; a specular highlight might be built from lots of small bits while the remaining function is represented with a few, larger bits.

In addition to the general issues mentioned above, the following issues (inspired, in part, by the success of splines as a modeling tool) were also considered when designing this surface model.

- User-defined arbitrary continuity. Although C^2 continuity is sufficient for most modeling situations, the ability to model smoother surfaces is sometimes necessary (for example, car bodies and airplane wings).

- A smooth parametric domain. We do not attempt to define a single parameterization of the entire surface; instead, we provide a local parameterization at any point of the surface and a method for smoothly changing between local parameterizations. Let us return to parameterizing the sphere. Around the equator we can use the standard longitude and latitude lines (see the left side of Figure 1.6) but at the north and south poles use a different parameterization (the right side of Figure 1.6). These parameterizations do not match up (and hence do not form a global parameterization), so we also need to define a way to blend between them. A smooth parametric domain is useful for calculating a variety of surface attributes (e.g., geodesics, derivatives) and for operations such as texture mapping.

- A compact representation. A complicated spline curve can be represented by a set of control points and a knot vector. For a uniform B-spline, just the control polygon suffices (and also provides a fairly simple way to manipulate the shape of the curve at a high level). By analogy, the surface model described here starts with a control polyhedron that describes and controls the surface shape and topology.
- Computational tractability. The surfaces are simple enough to compute that they can be constructed interactively.

One way to view our technique is as an extension of B-splines to arbitrary topologies (indeed, for rectangular surfaces our technique reduces to B-splines). A different view is that we are constructing a smooth structure from the structure of a polyhedron that can then be used as a parameterization of that polyhedron. If the points of the polyhedron are in a one-to-one correspondence with another object, for example, a subdivision surface or an implicit surface, then we can relate the smooth structure to that object as well. This has the potential to simplify such tasks as texture mapping implicit surfaces.

Chapter 2

Previous work

The problem of modeling free-form surfaces has been approached from a variety of directions. Most techniques fall into one of four classes: solids models, implicit models (including algebraic surfaces), geometric models (e.g., polygonal models), and parametric models (e.g., splines).

Solids models [Man88][Hof89] build complicated models by combining basic 3D geometric primitives such as cylinders, cones, and cubes. They can model any oriented topology but are limited in geometric detail. The individual geometric primitives have their own parameterization but there is no relationship between these parameterizations.

Implicit surfaces [Mur91] [BS91] [WMW86] have found a limited use in modeling "soft" objects. They naturally model smooth shapes of arbitrary topology but have difficulty modeling objects with intricate surface geometry (for example, the face on the object in Figure 2.1) because there is no small-scale control over the surface. Its geometry is determined by the underlying skeleton and the offset function, both of which are fairly large-scale. Also, they have no parameterization, making techniques such as parameter-space texture mapping [Ped95] and direct manipulation more difficult [WH94]. Surfaces with boundary are formed by clipping an implicit surface to a volume, say a tetrahedron [War89] [BI92a] [Sed85]. These surfaces (called *algebraic patches*) have two advantages: positional and derivative constraints are easily expressed and solved for, and they typically have a lower degree than their parametric counterparts. Complicated surfaces can be made by stitching together these algebraic patches [BBX95]. Unfortunately, these surfaces are difficult to use and expensive to render (rendering requires either marching cubes [LC87] or a ray tracer). Additionally, it is difficult to guarantee that the patch is single-sheeted, i.e., that the isosurface passes through the clip volume exactly once.

Polyhedral models can approximate nearly any surface but they have only C^0 continuity, and accuracy is achieved at the cost of model size. One method for producing smooth models from polyhedra is *subdivision* [CC78] [Nas87] [BS88] [Loo87]. This process begins with an arbitrary polyhedron and through a process of "chipping off" the corners produces, in the limit,

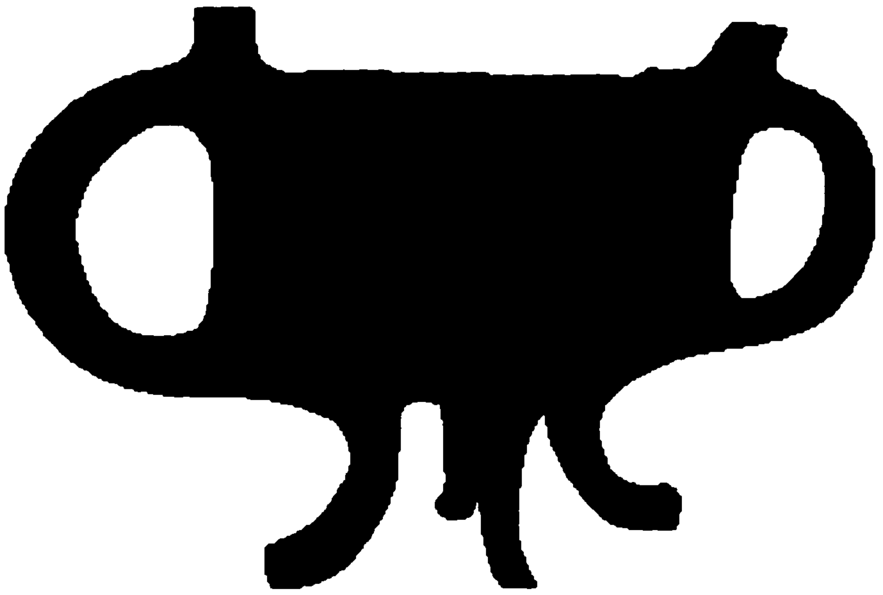


Figure 2.1: A model of a ding (an ancient Chinese bronze ritual vessel).

a dense set of points describing a smooth surface. The surface is defined not analytically but as the limit of a series of polyhedra; this makes any analysis of it very difficult, since there is no simple way to determine even the continuity of it [DS78]. Recent work [DK93] has made these surfaces more tractable but they still lack a smooth parameterization. One further problem is that the surface produced is at best C^2 smooth (and in some cases not even C^1 [HB94]) and there is no obvious way to produce smoother surfaces.

The most common parametric model is the spline, in particular, NURBS and their simpler cousins, B-spline and Bézier surfaces [BBB87] [Far88]. Unfortunately, these surfaces are based on a rectangular domain and therefore do not naturally model arbitrary topologies. The different approaches to making splines model non-rectangular topologies have all proceeded by "stitching" together various types of spline patches in different ways. These techniques fall into three broad classes: filling in n-sided holes in otherwise regular meshes, allowing patches to meet n at a vertex, and more global methods involving either constraints and minimization (variational modeling) or a systematic conversion of an irregular mesh into regular patches.

The most complete method involving n-sided patches is S-patches [LD89] [LD90]. These patches have the advantage that the derivative information along an edge is independent of the other edges (given a high enough degree). Joining them together smoothly, however, requires Bézier-style constraints on the control points; this reduces the available degrees of freedom and complicates tasks such as data fitting. Other methods [Sab83] [War92] [Gre83] [DM84] are computationally expensive or non-general. Mann *et al.* conclude that these and other polynomial interpolants are unsatisfactory [MLL+92].

Hollig and Mogerle [HM90] explore a general method for determining continuity constraints on patches meeting n at a vertex. Van Wijk [Wij86] discusses the specific case of three and four, or an odd number of, patches meeting at a point. Chiyokura and Kimura [CK83] define a surface on which any number of patches can meet at a point, although continuity across this point is not discussed. Neamtu and Pfluger [NP94] use global geometric information to constrain patches locally; parametric continuity is not discussed.

In variational modeling, the shape of the patches is determined by a set of sketch curves (containing positional and derivative information) and the minimization of an energy functional [MS92] [WW92] [CG91]. These techniques produce pleasing surfaces (close to C^2 or higher, although this is not guaranteed) at considerable expense. Witkin and Welch [WW94] extend this technique to an interactive system by replacing the surface patches with triangular elements (essentially a finite element solution). Sketching with curves is natural for many objects that are fairly regular, but it is unclear if this technique is suitable for more free-form objects such as that in Figure 1.1.

The most comprehensive approach to date are Loop's patches [Loo94b] [Loo94a], work building on Peters' techniques [Pet92] [Pet93]. These techniques begin with an arbitrary mesh, produce a more regular mesh with particular properties, then approximate the mesh using three- and four- sided patches. One advantage is that the patches are constructed using the geometric information in the original mesh instead of calculating continuity constraints on a per-edge basis (the conditions for continuity are established in general and can be satisfied by assigning linear combinations of the vertices of the mesh to the control points).

In general, all of these spline techniques build a surface in a quilt-like fashion, with continuity between pieces of the quilt maintained by constraints on the individual patches. This quilting has two drawbacks: changing the topology or geometry of the surface requires recalculation of the constraints between the affected patches and there is no global parameterization. The former makes data-fitting or direct manipulation techniques difficult because these internal constraints must be maintained on top of the external constraints. The latter makes texture mapping and other parameter-space operations difficult.

Operation	Solids Modeling	Subdivision Surfaces	Implicit Surfaces
Rendering	Triangulation	Subdivide until small enough	Ray tracing or marching cubes
Continuity	C^{-1} to C^{∞}	C^2	C^0 to C^{∞}
Parameter space	Local to geometrical primitives, discontinuous between primitives	Initial polyhedron (not smooth)	None
Geometrical control	Boolean combinations of primitives	Location of polyhedron	Skeleton and offset function
Local control	Yes	Yes	No
Hierarchical	Tree structure of primitives and operations	Yes	No

Operation	Patch Gluing	Manifolds
Rendering	Triangulate domain or ray trace	Triangulate domain
Continuity	C^2	C^k for a given k
Parameter space	Local to patches, discontinuous between patches	C^k
Geometrical control	Unconstrained control points	Control points
Local control	Yes	Yes
Hierarchical	Yes (Oslo algorithm)	Unknown

Table 2.1: A summary of the different techniques discussed in this section. Although not discussed in this section, we include the technique developed in this paper (listed under the heading "Manifolds") for completeness.

Chapter 3

A different approach

It is easier to build a complicated surface by stitching together simpler ones than by describing the entire surface at once. Many of the approaches mentioned in the previous chapter do precisely that; they begin with n-sided patches and "glue" them together. Although the glue used varies somewhat, these approaches all have one thing in common: the patches are glued together by abutting their edges, as shown in Figure 3.1. This approach has several disadvantages:

- The gluing relies on constraints on the control points of the patches. Enforcing continuity in this manner presents two problems; whenever the control points are moved the constraints must be re-established, and there is generally a reduction in the number of degrees of freedom of the surface. The latter makes operations such as data fitting more difficult because maintaining the constraints while reducing the approximation error is in general more difficult than just reducing the error.

- The domain has discontinuities at the patch boundaries. These discontinuities cause difficulties when defining operations whose domain crosses a patch boundary, for example, texture mapping. They also present problems when defining derivatives across the boundaries – much of the patch literature [Sab83] [LD90] [HM90] is concerned with defining what continuity means across such boundaries.

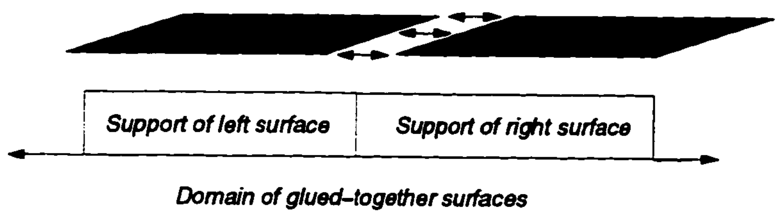


Figure 3.1: Two surface patches glued together by abutting along their edge, making them positionally continuous. The supports of the patches do not overlap.

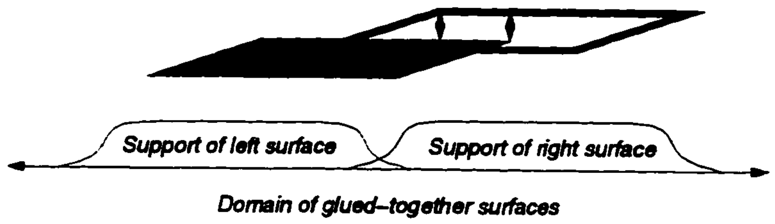


Figure 3.2: Two surface patches glued together by overlapping; the supports of the patches overlap substantially.

- Increasing the continuity across a patch boundary is difficult, both deriving the necessary constraints and maintaining them (increasing the continuity increases the number of control points needed to maintaining the constraints).

As a different approach, consider Figure 3.2. Here the domains of the patches overlap substantially instead of abutting. There is room to move from the domain of one patch to the domain of the next, which eliminates the problem of domain discontinuities. As we demonstrate later, the patches can be glued together *before* geometry is assigned to them, meaning the glue does not rely on geometric constraints.

> **Aside 1** An analogy to splines: *The difference between these two approaches, abutting versus overlapping, is similar to the difference between extending a Bézier curve and extending a B-spline curve. To extend a Bézier curve in a smooth fashion, another curve is added and some constraints are placed on the first few control points of the new curve to ensure a continuous join (see Figure 3.3). In this case the curves are abutted, with the locations of the control points chosen to maintain continuity.*
>
> *In contrast, to extend a B-spline curve, another basis function-control point pair is added (see Figure 3.4). This "adds" the section of curve defined by the four basis functions $b_1, \ldots, b_4$ to the existing section of curve, already defined by the four basis functions $b_0, \ldots, b_3$. The curve sections are overlapped along the portion $[1, 7]$ of the real line.*

The idea of describing a complicated surface with many overlapping bits of simpler surfaces is not a new one – mathematicians call such a structure a *manifold* [Spi70][ST67]. A world atlas can be viewed as an informal example of a manifold (see Figure 3.5): each page of the atlas is rectangular (i.e., a simple bit of surface) but the collection of pages describes a spherical object, the world; the pages of the atlas overlap enough to show how to get from one page to the next. For example, the page for France contains part of Spain, and the page for Spain contains part of France. When traveling from France to Spain there is a time when one is located on both pages simultaneously; the two maps may not be identical where they overlap but they contain enough information to establish a correspondence between the two pages.

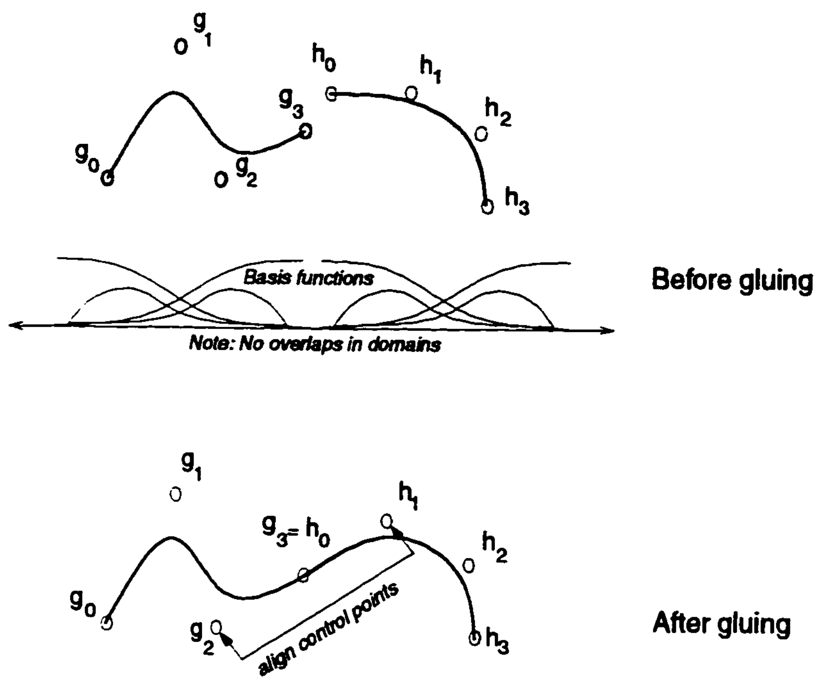


Figure 3.3: Extending a Bézier curve by joining another Bézier curve to it. Continuity is established by constraining the first two control points of the second curve.

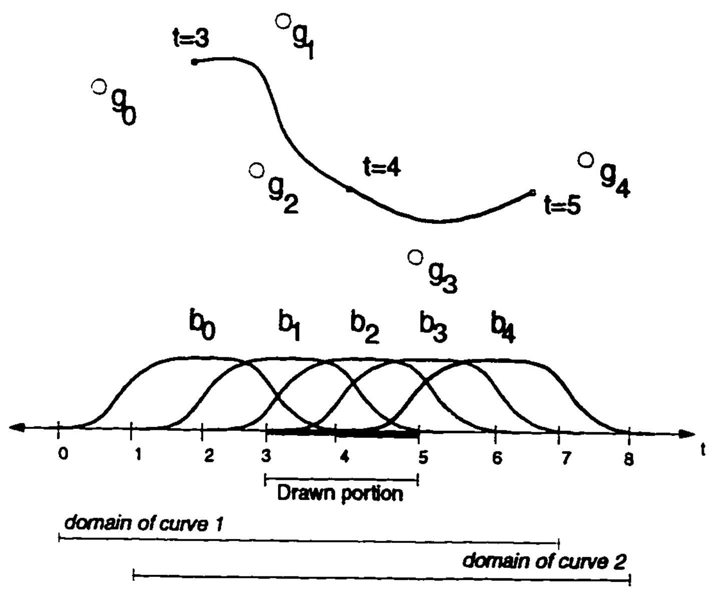


Figure 3.4: Extending a B-spline curve by adding another segment. Continuity is automatically maintained.

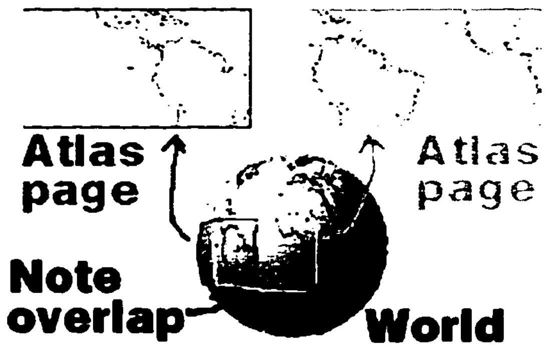


Figure 3.5: Two overlapping pages from a world atlas.

Of course, map makers already have their complicated surface (the world) and are simply covering that surface with simpler surfaces (the atlas pages). Since we are trying to build a surface we need to reverse this process. Suppose you wanted to describe the earth to someone: you could hand them an atlas and they could reconstruct what the earth looks like by gluing the pages together. In fact, you could describe an imaginary world by "making up" an atlas of that world. We propose to do just that: "make up" an atlas for the surface we want, then build that surface by gluing the pages of the atlas together.

As a simpler version of this, consider "making up" an atlas of a park. Each page of the atlas is some part of the park, for example, the swings, the pond, the grassy field, the boat dock, etc. Each page is labeled with the part of the park it shows and contains a bit of the neighboring area, also labeled. For example, the page containing the boat dock also shows a bit of the pond and the grassy field. We now have an object that allows us to navigate around the park; i.e., we can figure out how to get from the boat dock to the swings by tracing paths through the pages (see Figure 3.6).

Although this information is sufficient for navigating from one part of the park to another, we still need one other piece of information to build the park — we need to know what the park *looks like* — how big the pond is, how much sand is around the swing set, etc. One method for adding this information is to make a geometrical model of each page. We may need to do some blending between the models — the bit of the model for the boat dock may not match up exactly with the model for the pond — but with a little care we can now build a model of the park.

The remainder of this section describes how to build a curve in this manner. We do not create a new "type" of curve in this discussion — the curve is nearly identical to a B-spline curve — but we are expressing the material in a different language. Later, when we repeat this construction for a surface, this different language lets us build surfaces that are more general than B-spline surfaces.

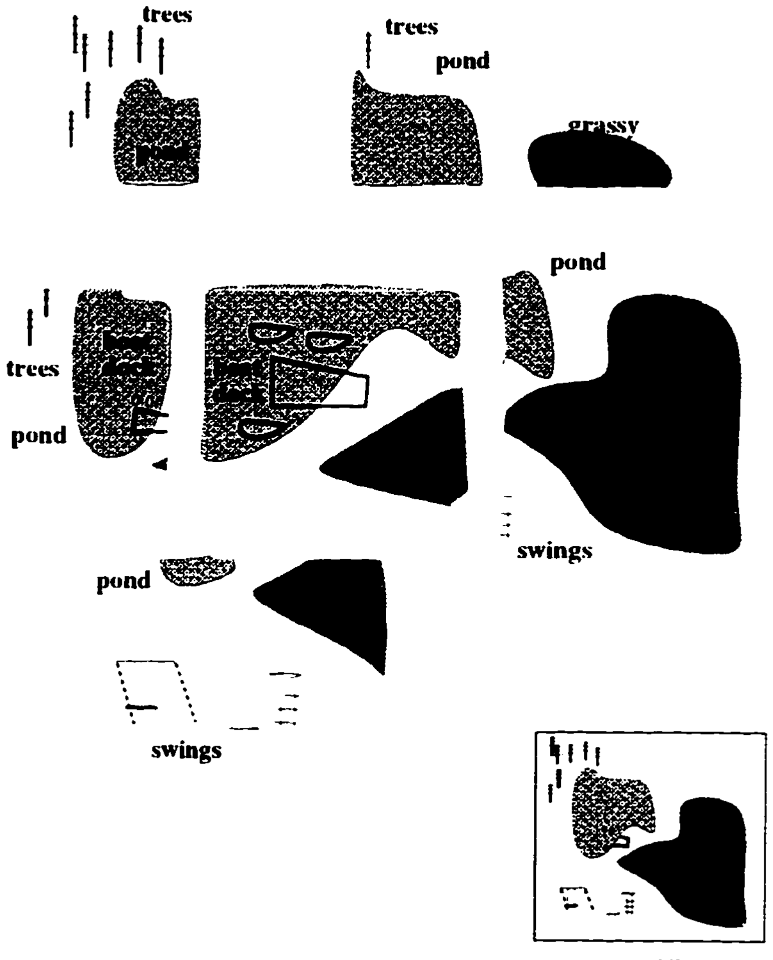


Figure 3.6: An atlas describing a park.

In the following sections we describe the steps involved in building a curve using an atlas. The sequence of steps is outlined in Figure 3.7; the individual steps are expanded as follows. To get started, we need a "sketch" of the desired curve – this sketch is a polygon, the details of which are given in Section 3.1. From this sketch we build an atlas, i.e., the parts of the park and how one gets from one part to another. This atlas is then stitched together into a manifold by gluing the pages together (Section 3.2). This manifold has the *topology* of our goal curve but not the *geometry*. We assign geometry to the manifold to produce the goal curve, i.e., to build a model of our park (Section 3.3).

In the following discussion we use the terms "page" and "world" for simplicity's sake; note that these correspond to the more formal terms "chart" and "manifold" used in later sections and traditional manifold literature.

3.1 The sketch, or polygon

To sketch our desired curve we need something that looks a lot like a curve but is quick and easy to create and involves only a small amount of data. We use a polygon, a collection of vertices and edges. Each edge consists of two vertices, and each vertex has exactly one or two edges adjacent to it.[1] The sketch for this example is the polygon in Figure 3.8, which has four vertices, $v_0, \ldots, v_3$, and four edges, $\{v_0, v_1\}, \ldots, \{v_3, v_0\}$.

3.2 The atlas

The atlas consists of a set of pages and information on how those pages overlap (i.e., the "map" of the park). Note that the set of pages chosen here is only one of many possible choices.

We put one page in the atlas for each element (each vertex and each edge) in the polygon. This may seem excessive – we could, for example, get by with just one page per vertex. The redundancy will prove useful, however, for blending between the pages, especially when we extend this technique to surfaces. Our example polygon has four vertices and four edges, so our atlas has eight pages. Each page must have some sort of topology – in a world atlas, the pages are rectangles. Since we are modeling a curve, the pages are a subset of the real line; we chose the unit interval $(-0.5, 0.5)$ for all the pages. The atlas constructed from the example polygon is shown in Figure 3.9.

Next we decide how each page overlaps with other pages in the atlas – which bit of the pond on this page is the same as which bit of pond on that page. This requires some care to ensure that the maps are consistent (Section 4.2). If one walks from the grassy field to the pond and then turns around and goes back, one expects to end up back on the grassy field. Fortunately,

[1] The sketch must look, locally, like a curve or line segment, not a T-joint; hence a vertex can not have more than two adjacent edges.

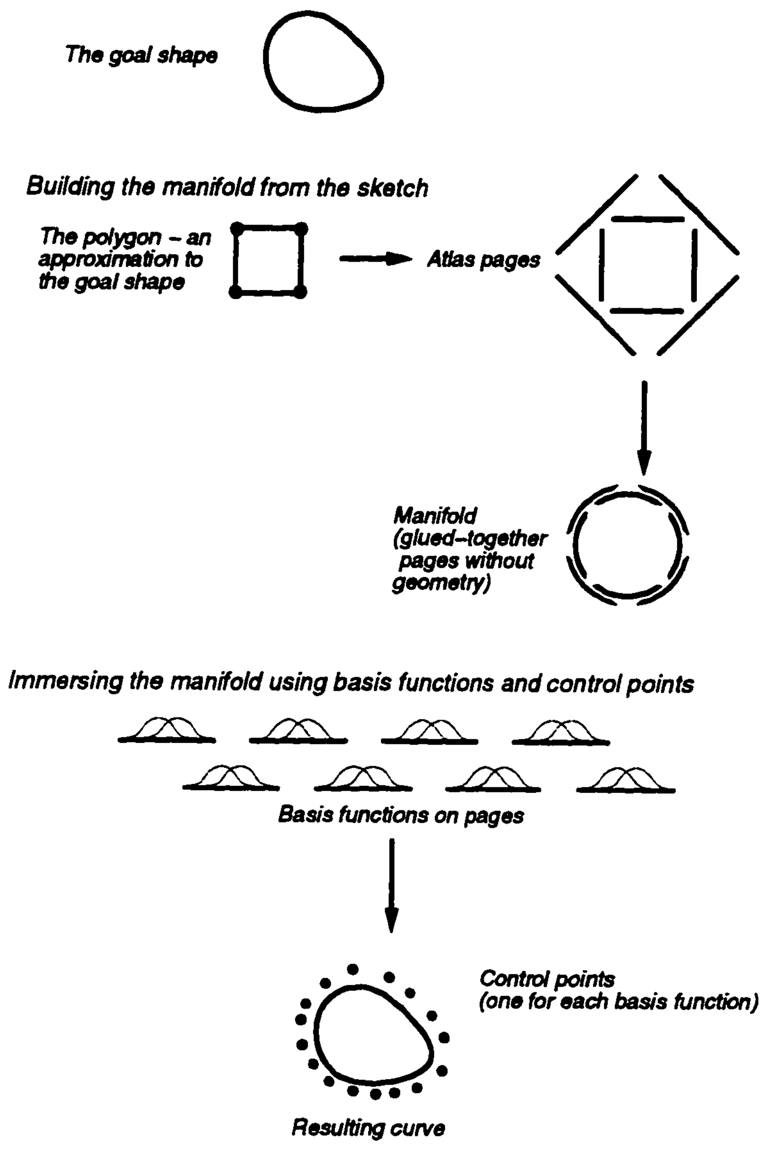

The goal shape
Building the manifold from the sketch
The polygon - an approximation to the goal shape
Atlas pages
Manifold (glued-together pages without geometry)
Immersing the manifold using basis functions and control points
Basis functions on pages
Control points (one for each basis function)
Resulting curve

Figure 3.7: An outline of the curve construction process. A manifold is built from the sketch polygon and then immersed using control points and basis functions.

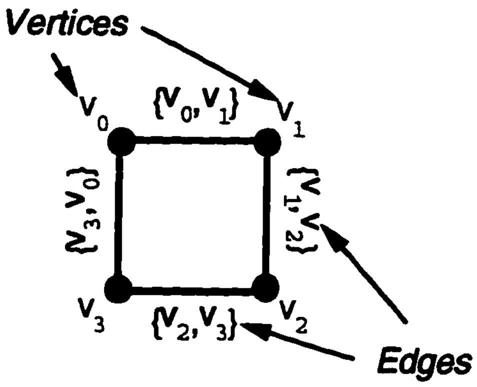


Figure 3.8: The polygon sketch, with four vertices and four edges.

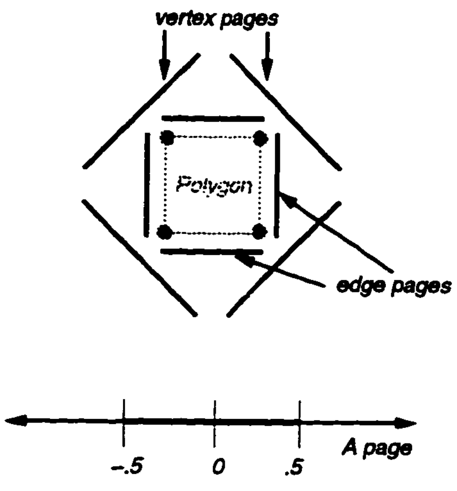


Figure 3.9: The eight atlas pages corresponding to the four vertices and four edges of the sketch polygon.

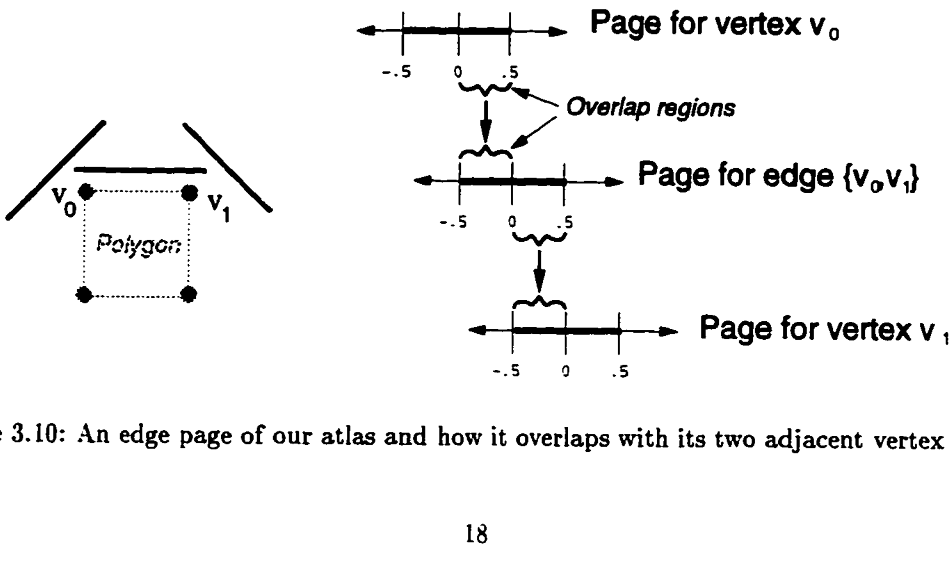


Figure 3.10: An edge page of our atlas and how it overlaps with its two adjacent vertex pages.

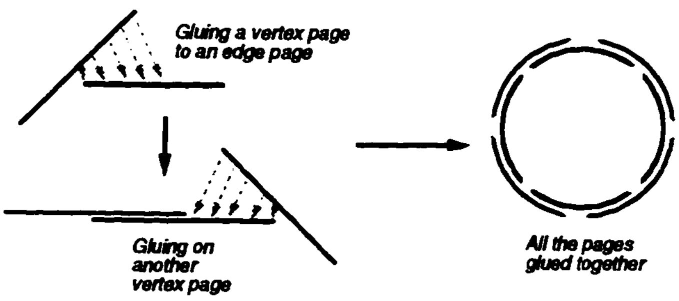

Gluing a vertex page to an edge page
Gluing on another vertex page
All the pages glued together

Figure 3.11: Left: Gluing the two adjacent vertex pages to the edge page. Right: All of the pages glued together.

we already have an idea of how the pages should overlap - the connectivity information of the polygon. Each edge of the polygon contains two vertices; we therefore make the edge page corresponding to that edge overlap with the pages for the two vertices. Similarly, a vertex is adjacent to one or two edges, so the corresponding vertex page overlaps with one or two edge pages. In our example, the edge page corresponding to the edge $\{v_0, v_1\}$ overlaps with the vertex pages for the vertices v_0 and v_1 (see Figure 3.10). We want the pages to overlap as much as possible (imagine navigating from France to Spain using an atlas; the more the atlas pages overlap, the easier it is to establish a correspondence between the two pages). However, we do not want to complicate the gluing process by having two vertex (or edge) pages overlap. These restrictions lead to defining the overlap regions to be one of $(-0.5, 0)$ or $(0, 0.5)$, i.e., the left and right halves of the pages.

We need more information than just the fact that the pages overlap; we also need a function that identifies points of one page with the points on an overlapping page. For example, we need a function that takes the region $(0, 0.5)$ on the page for vertex v_0 to the region $(-0.5, 0)$ on the page for edge $\{v_0, v_1\}$. This function is called a *transition function*. The transition functions in our example are simply translations. For example, if $t = 0.3$ is a point on the page for v_0 then the corresponding point in the page for the edge $\{v_0, v_1\}$ is $t - 0.5 = -0.2$.

We now have a set of pages and information on how they overlap. How do we turn this into an "object"? Imagine gluing each of the pages onto a stretchy bit of string. Now glue the bits of string together according to the overlap information. The points that were related via the overlaps (e.g., the point 0.3 on the page for the vertex v_0 and the point -0.2 on the edge page $\{v_0, v_1\}$) get glued together to form a single point on this fatter piece of string. This creates an object (see Figure 3.11) in which matching pairs of points on the pages have been glued together to form a single point (a "string sandwich"). This object is our manifold.[2]

In summary, we created a page for each element of the polygon, decided how those pages overlapped using the connectivity information of the polygon, then glued the pages together

[2]In practice this "gluing" is generated by taking equivalence classes under an equivalence relation generated by the transition functions.

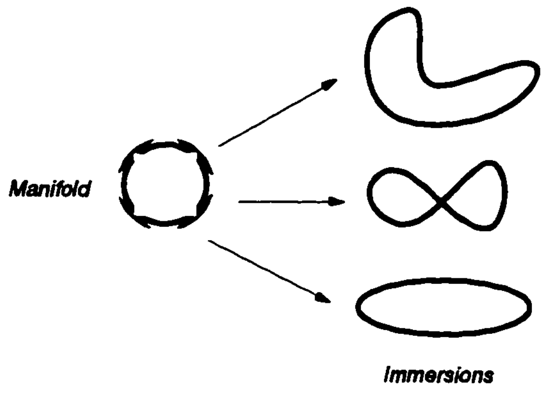


Figure 3.12: The glued-together pages stretched out into several shapes.

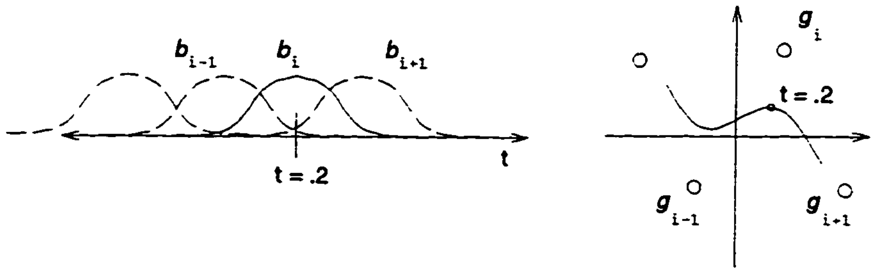


Figure 3.13: Defining a B-spline. Left: Four basis functions defined on the real line. Right: The corresponding four control points and the resulting curve.

using the overlap maps. This is not the end of the story; we have an object that has the same *topology* as our polygonal sketch, but it has no *geometry* (the manifold could be drawn as a circle, an ellipse, a square, etc.)

3.3 Adding geometry to the manifold

We can take our glued-together bits of stretchy string and stretch it out into a variety of shapes (provided that shape has the topology of a circle), as shown in Figure 3.12. Describing what shape to stretch the string into is called defining an *immersion*.[3] In this section we describe how to assign geometry to, or immerse, the manifold. This is accomplished by drawing on a traditional technique used to immerse the real line into space - B-splines. What we have described so far is essentially a different way to build the real line (or the *domain*) of our curve. In traditional B-splines this is unnecessary; the basis functions are built on the existing real line.

[3] An immersion is an embedding that can cross itself, as demonstrated in the middle shape in Figure 3.12.

Aside 2 B-splines – a brief description: *For this discussion, refer to Figure 3.13. A complete discussion of splines can be found in [BBB87]. A B-spline curve* $\gamma\colon \Re \rightarrow \Re^2$ *maps a subset of the real line into space using* basis functions *and* control points. *The* ith *basis function* $b_i : \Re \rightarrow \Re$ *tells how much of the* ith *control point* $g_i \in \Re^2$ *to take at a given point on the real line. The curve is a* blend *of these control points, where the blending is controlled by the basis functions:*

$$\gamma(t) = \sum_{i=0}^{n} b_i(t) g_i$$

For example, at $t = 0.2$ *we take a blend of the three control points: a little of* g_{i-1}*, a lot of* g_i*, and a little of* g_{i+1}*.*

The basis functions can be defined in a variety of ways; the different spline types are distinguished by their different basis functions.

To create a B-spline curve we must first define the basis functions on the real line – the b_{i-1}, b_i, b_{i+1} functions of Figure 3.13. Next we associate a control point with each basis function. The resulting curve is then a blend of those control points.

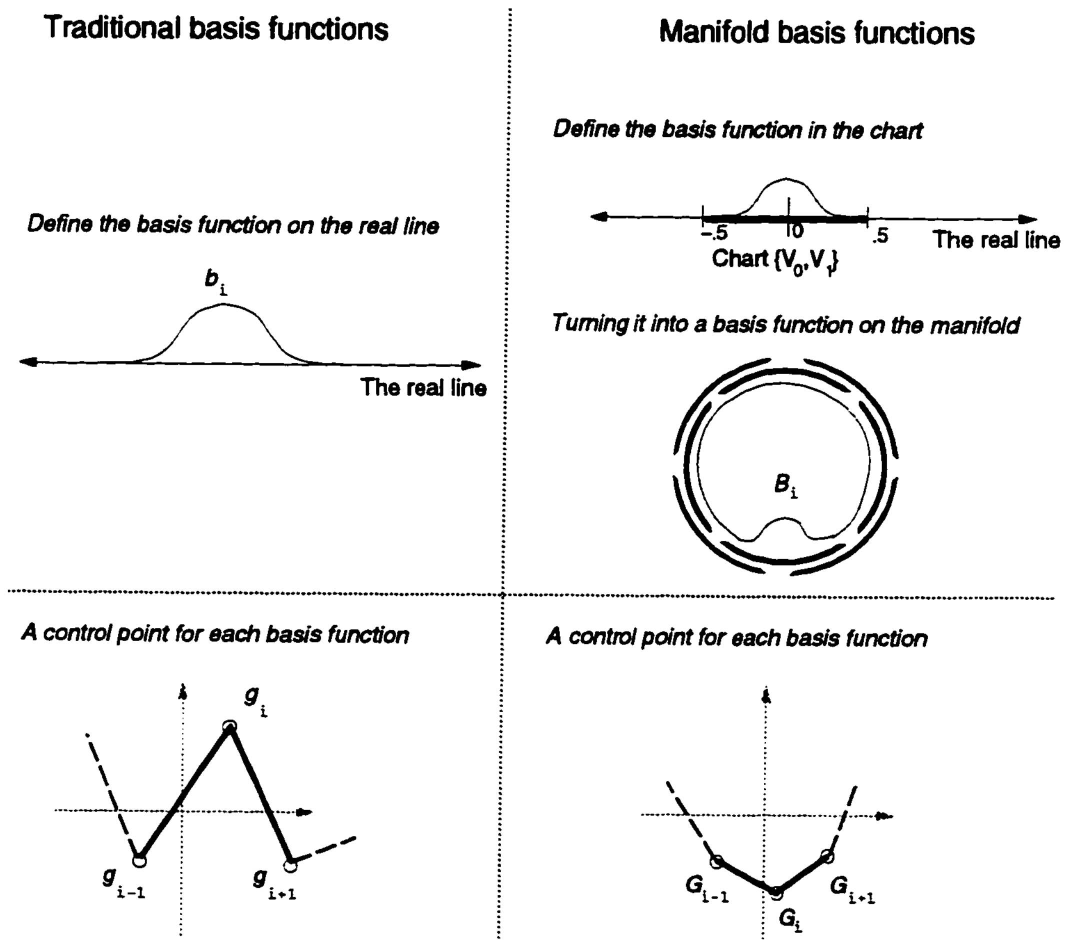
To immerse the manifold we duplicate these steps except instead of building the basis functions[4] on the real line we build them on our new "real line" – the manifold (see Figure 3.14.

Building basis functions on the manifold is difficult. Suppose we take a bit of the manifold and draw it flat, as shown in Figure 3.15. This bit of manifold looks a lot like the real line. The problem is identifying the relationship between points on the real line and points on the bit of manifold. To avoid this issue we do not build basis functions directly on the manifold; instead, we define each basis function on a page of the atlas, as shown on the right of Figure 3.14. Each page has a coordinate system, so defining a function on a page makes sense.

Suppose we have a basis function defined on a page of the atlas. Turning this into a function defined on the manifold is relatively straightforward; the exact mechanics are left until later (see Section. 6.1) but pictorially we get the manifold basis function shown on the right of Figure 3.14. Because of the way we build functions on the manifold, the support of each basis functions lies within a single chart, i.e., it is zero on the part of the manifold that does not contain the chart. It is possible to define a function whose support is contained in multiple charts but guaranteeing continuity in this case becomes difficult.

Our goal is to create a curve that behaves in a manner similar to a spline. This means we need to create functions on the charts that look like traditional basis functions. Figure 3.16 shows how to define two such basis functions on each of three pages of the atlas. Note how the functions can be drawn on the overlapping pages, using the transition functions to determine what the function "looks like" on the overlapping page.

[4] We call them "basis functions" because they are related in concept to traditional basis functions.



Traditional basis functions
Manifold basis functions
Define the basis function on the real line
b_i
The real line
Define the basis function in the chart
-.5
0
.5
The real line
Chart $\{V_0,V_1\}$
Turning it into a basis function on the manifold
B_i
A control point for each basis function
g_i
g_{i-1}
g_{i+1}
A control point for each basis function
G_{i-1}
G_i
G_{i+1}

Figure 3.14: Left: The construction process for a B-spline curve. Right: The construction process for a manifold curve. Note that a B-spline basis function is defined on the entire real line while a manifold basis function is first defined on a chart, then promoted to a function on the manifold.

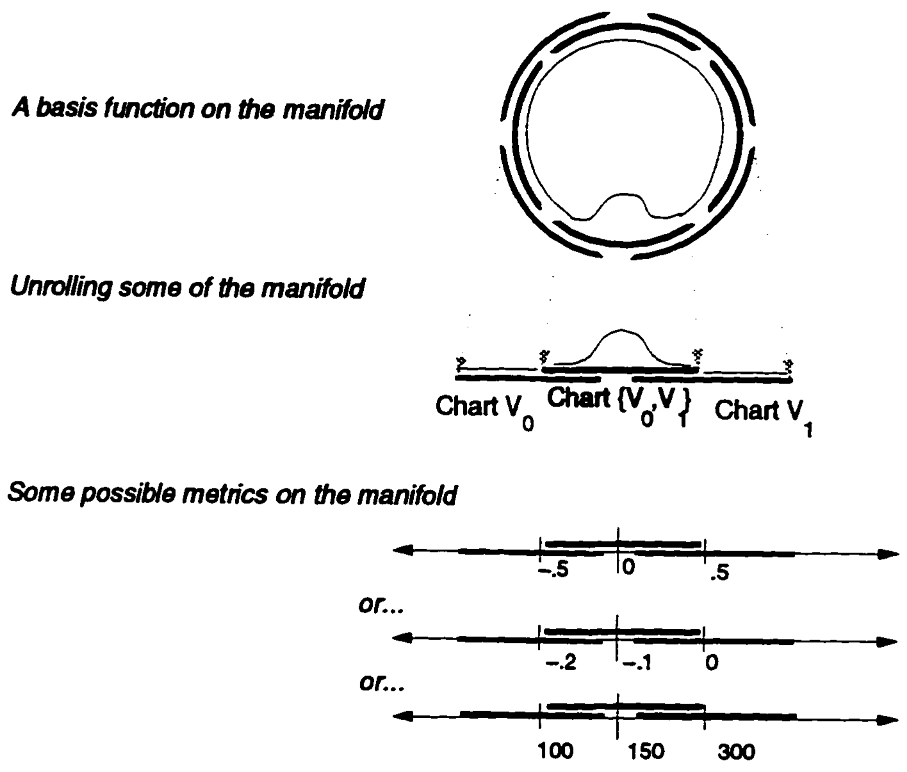

A basis function on the manifold
Unrolling some of the manifold
Chart V_0
Chart $\{V_0 V_1\}$
Chart V_1
Some possible metrics on the manifold
-.5
0
.5
or...
-.2
-.1
0
or...
100
150
300

Figure 3.15: The manifold itself has no inherent metric. The unrolled bit of manifold can be put into correspondence with the interval $(-5, 5)$, or $(-2, 0)$, or any other contiguous subset of the real line.

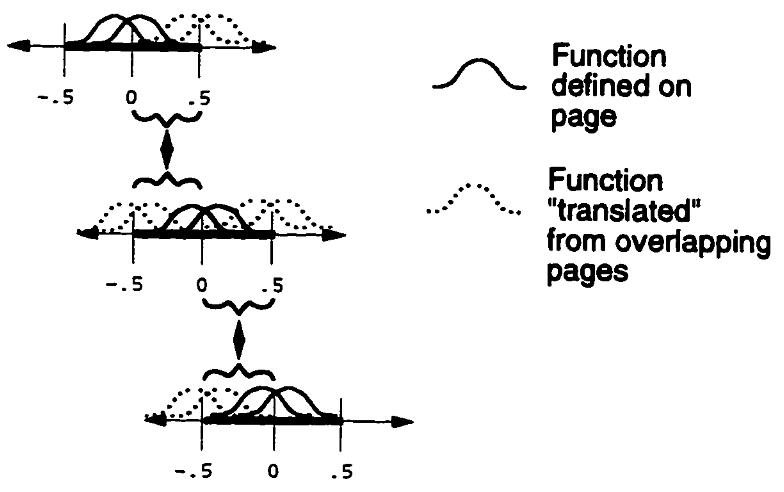


Figure 3.16: Three pages of the atlas. Two basis functions are defined on each page (bold curves). We also show the functions from the overlapping pages and how they would appear on the page (dotted curves).

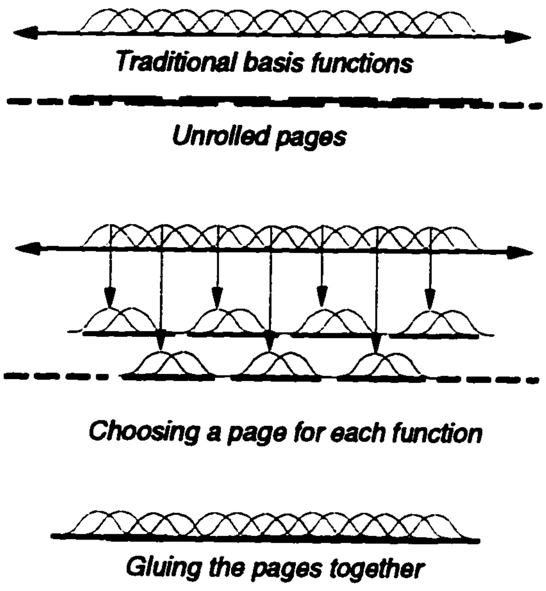

Traditional basis functions
Unrolled pages
Choosing a page for each function
Gluing the pages together

Figure 3.17: How to define the basis functions on the pages of the manifold so that when the pages are glued together the basis functions overlap in a manner similar to traditional basis functions: we unroll a bit of the manifold and line up the "pages" with a segment of the real line so that the support of each basis function is contained within a single page.

The difficulty is picking these functions so that when turned into a set of basis functions on the manifold they resemble traditional basis functions. We first take a bit of the manifold, draw it flat, then stretch it so it lines up with the real line, as shown in Figure 3.17. Each basis function is nonzero on some subset of the real line (the *support* of the function). We line up the manifold with the real line so that the support of each basis function lies within a single page. We now assign each basis function to the page that contains its support. Note that the new functions drawn on the manifold resemble the traditional basis functions on the real line.

Figure 3.18 shows an immersion of our example manifold, with two basis functions (and hence two control points) per page.

3.4 Summary

A polygonal sketch supplies the desired topology of the manifold. The manifold is related to the polygon in the following ways:

- The pages in the atlas. There is one page in the atlas for each element in the polygon.
- The overlaps of the pages. Pages overlap if and only if their corresponding elements in the polygon are adjacent.

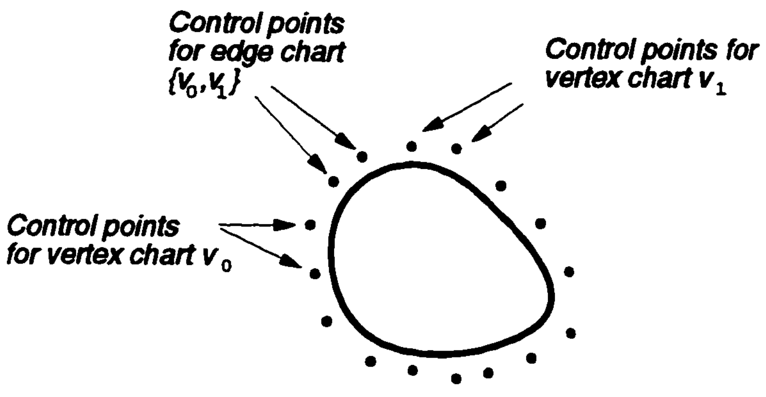

Control points for edge chart $\{v_0,v_1\}$
Control points for vertex chart v_1
Control points for vertex chart v_0

Figure 3.18: The manifold immersed into 2-space. The topology of the immersion is determined by the topology of the manifold but the geometry of the immersion is determined by the locations of the control points.

Another way to view this construction is that the manifold is a "smoothness structure" on the polygon. This structure is abstract in that it has no $2D$ geometry, but it is topologically related to the polygon. In fact, we can establish a one-to-one correspondence between the points of the manifold and those of the polygon.

We assign geometry to the manifold by immersing it using basis functions and control points. The basis functions are constructed on the pages of the manifold in a way that mimics traditional B-splines when they are drawn on the manifold. The locations of the control points determine the shape of the immersion — the immersion is a curve that smoothly blends between the control points.

3.5 Commentary on this approach

In this section we explore some of the benefits and consequences of this approach, now that its basic flavor has been established. We discuss first the breakdown of the construction process into two steps, building the manifold and then immersing it, and then the parameterization of complicated surfaces.

3.5.1 Two-step process

There are two major reasons to break down the surface construction process into two steps: to establish the continuity of the surface and to provide flexibility in defining immersion functions. The manifold captures the topology of the desired surface as described by the polyhedron, but the manifold itself does not "live" in 3-space. Instead, it serves as a domain that has the same topology as the desired surface, much as a subset of the real line is the domain for a B-spline curve. It is more difficult to define functions on this manifold than on the plane but this disadvantage is outweighed by the coherence the manifold provides.

For example, consider determining the continuity of two Bézier curves joined together as shown in Figure 3.3. At every point on the curves, with the exception of the end points, continuity is determined by taking derivatives of the function defining the curve. At the end points, however, the derivatives only exist from one direction. It is possible to define the continuity of the curves where they join together by considering these one-sided derivatives but this definition differs in format from the derivatives on the remainder of the curve.

In contrast, derivatives of functions defined on the manifold are calculated in the same manner across the entire manifold [MP77], pp. 208-218. This definition is somewhat more complicated than the traditional one from calculus but it is consistent across the manifold.

The practical result of this is that the continuity of the surface does not depend on the locations of the control points. This implies that the geometry of the surface can be changed freely (via the control points) without the need for additional continuity constraints.

In this paper we present a particular immersion based on uniform B-splines. This immersion can easily be extended to non-uniform B-splines, or even replaced with an immersion function based on wavelets, without changing the manifold.

3.5.2 Parameterization

There is no global parameterization for the constructed surface. Instead, the charts provide a local parameterization for any given point, and moving from one point to another on the surface involves moving from one chart's parameterization to another's.

There is no guarantee that the particular set of charts used to build a manifold will result in a "useful" parameterization. However, there is a great deal of freedom in picking charts — as long as the charts and their overlaps meet certain criteria, given in the following section, we are free to add in as many charts as we want. For example, suppose we want to add to the curve on the left of Figure 3.19 a small, detailed bump as shown on the right. We simply add another small chart to the atlas that covers the relevant area and use it to define the bump. This solves the problem of reparameterizing an entire surface in order to define detail on a small portion of it.

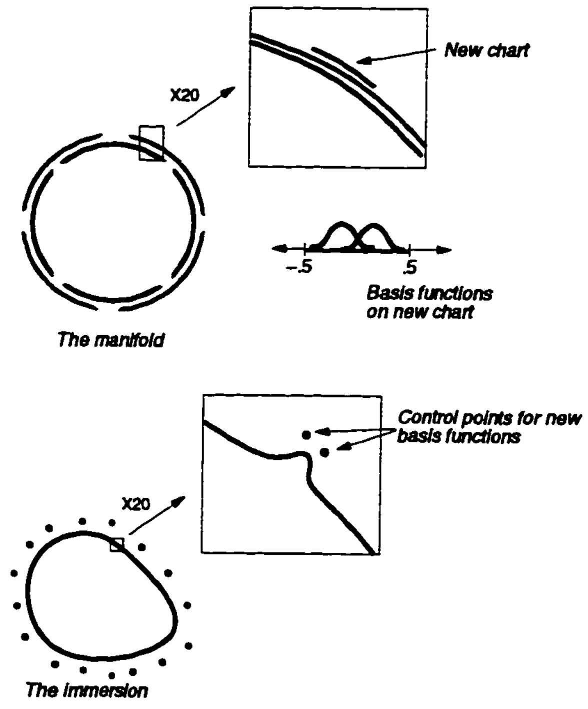

X20
New chart
-.5
.5
Basis functions on new chart
The manifold
Control points for new basis functions
X20
The immersion

Figure 3.19: Adding detail to a manifold curve by adding a small chart to the manifold and defining new basis functions on that chart.

Chapter 4

Building a manifold – an informal discussion

In this section we extend the curve example of Section 3.2 to a surface and begin to develop the terminology needed to discuss manifolds, going on to formalize this information in Chapter 5. The layout of this section is as follows:

- An informal definition of the world atlas example. The atlas is an example of an analytic study of a surface — we begin with a complicated surface (the world) and analyze it using simpler surfaces (the rectangular pages).
- A more complete discussion of the construction of a world from an atlas, or the inverse of the analytic approach. This is an example of a synthetic approach — we begin with simple surfaces and construct a more complicated surface from them.
- An informal discussion of the construction of a manifold from a polyhedron, using the synthetic approach previously outlined. This is the same construction process as used in the curve example (Section 3.1) but with a surface instead of a curve.

4.1 Manifolds, an analytic view

Suppose we have a surface (in the atlas example, this surface is the world) with the property that, when looked at locally, it resembles the plane.[1] We can take a bit of this surface and draw it in the plane, as shown in Figure 3.5. The function that maps the bit of surface to the plane is called a *chart function* (in the atlas example, this function results in a page of the atlas). To describe the entire surface we cover it with these chart functions, which can take different

[1]This discussion applies to any dimension, i.e., any object that is locally like $\Re^n$. For example, a surface is locally like $\Re^2$ and a curve is locally like $\Re^1$.

amounts of the surface to the plane (the atlas page containing Antarctica covers more of the world than the page for France) and can map the bits of surface to different shapes in the plane (imagine an atlas made out of round pieces of paper). There are three basic properties a set of chart functions must have, however, for the surface and the set of functions to constitute a manifold:

- Every spot on the surface must be covered by the domain of some chart function, i.e., every part of the world must appear on some page of the atlas.
- The chart functions must be well behaved, i.e., no creasing, or mapping two points on the surface to the same point in the plane. The chart functions must map the bit of surface into an open set in $\Re^2$.
- The chart functions must be consistent. Suppose the domains of two chart functions overlap on the surface (like the two pages shown in Figure 3.5). Take the intersection of these domains on the surface and map it into the plane using the first chart function (call this area in the plane A), then do the same thing with the second chart function (call this area B). A "nice" map must exist between area A and area B. As an example, consider two atlas pages, one for Germany and one for France. Each of these pages contains a picture of the same part of the world, i.e., the French-German border; the pages do not line up exactly but there is enough information (names of towns, roads, etc.) on the pages that a correspondence between the two is easily found.

If the chart functions meet more formal versions of these criteria, detailed in Section 5.1, then we have a manifold.

This analysis began with a complicated surface (the world) and produced a description of it in terms of simpler surfaces (the pages). Note that the existence of a surface is crucial to this analysis — we use the surface to produce an atlas for that surface. Suppose we do not have a surface to begin with; can we reverse this process, i.e., "make up" an atlas and produce a surface from it? The next section defines a method for building a surface from an atlas; the definition has many similarities to the previous one.

4.2 Manifolds, a synthetic view

In this section we begin with a set of *charts*,[2] i.e., bits of the plane, and information about how those charts overlap and then stitch the charts into a manifold by associating points in charts with points in other charts. Imagine that someone hands you the pages of an atlas; you could reconstruct the world the pages came from, in a sense, by gluing the pages together where they

[2] In the previous discussion a chart function is a map from the surface to the world. In this discussion a chart is a subset of $\Re^n$.

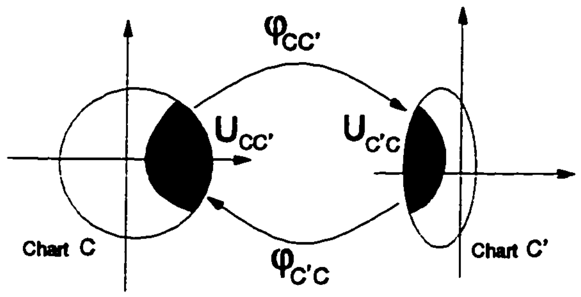


Figure 4.1: Two charts and the inverse property: for all $x \in U_{CC'}$ (the left red region), $x = \varphi_{CC'}(\varphi_{C'C}(x))$.

overlap. For this type of construction to work, the charts and the overlap information must have the following properties:

- Each chart is an open, connected subset of the plane (for example, a disk, or a rectangle without boundary). When referring to charts in the following discussion we use the labels C and C'.
- There must be information on what part of a chart overlaps with another chart. The area of a chart C that overlaps with another chart C' is labeled $U_{CC'}$ (and the part of chart C' that overlaps with chart C is therefore labeled $U_{C'C}$). Figure 4.1 shows two charts with these "overlap" regions demarked. Note that not all of the charts overlap, meaning that for some charts C and C', $U_{CC'} = U_{C'C} = \emptyset$.
- Knowing that the region $U_{CC'}$ of chart C corresponds to the region $U_{C'C}$ in C' is not sufficient; we also need to know the exact correspondence between the points in the two regions. This correspondence is given by a *transition function* and of the following form:

  $$\varphi_{CC'} : U_{CC'} \rightarrow U_{C'C}$$

  These transition functions tell how to stretch the pages before they are glued together. In the world atlas example we have no explicit transition functions but we do have the information drawn on the map, e.g., town names, road names, etc., which establishes a correspondence between the pages. In the previous definition (Section 4.1) this information was required to have a certain consistency; the transition functions for this definition must also be consistent. This is guaranteed by the following four criteria:

  - The transition functions are one-to-one, onto, and smooth, i.e., there is no creasing, no mapping of two points to a single point, etc.
  - If two charts overlap, then going back and forth between them using the transition functions puts you back where you started (see Figure 4.1):

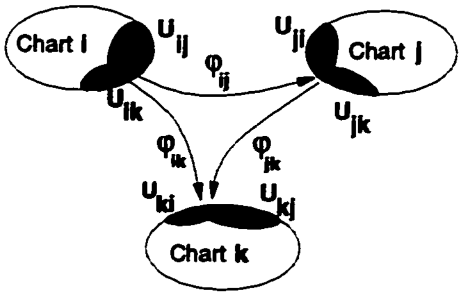


Figure 4.2: Three charts, their overlaps, and the co-cycle condition: for all $x \in U_{ij} \cap U_{ik}$ (the purple region in chart i) $\varphi_{ik}(x) = \varphi_{jk} \circ \varphi_{ij}(x)$.

$$\forall x \in U_{CC'} \quad x = \varphi_{CC'}(\varphi_{C'C}(x))$$

– The transition function relating a chart to itself is the identity function:

$$\forall x \in C \quad \varphi_{CC}(x) = x$$

– Any combination of transition functions that take a point through multiple charts puts you back where you started. This is called the *co-cycle condition* (see Figure 4.2):

$$\forall x \in U_{ij} \cap U_{ik} \quad \varphi_{ik}(x) = \varphi_{jk} \circ \varphi_{ij}(x)$$

• Every pair of overlapping charts can be glued together and the result embedded into $\Re^n$. This condition is needed later to prove that the entire glued-together object is "nice."

With these three objects (the charts, the overlap regions $U_{CC'}$, and the transition functions $\varphi_{CC'}$) we can build a manifold. This is a matter of making all of the points related by the transition functions become a single "point" in the final object. Details are given in Section 5.2; the process corresponds to gluing the atlas pages together.

4.3 Building a manifold

This section describes how to build a set of charts and transition functions from a polyhedral sketch of the desired manifold. A formal definition of the polyhedron is given in Appendix B; informally, the polyhedron conforms to the standard definition, with two restrictions. First, the vertices have valence four (see the top of Figure 4.3) and second, the faces have fewer than

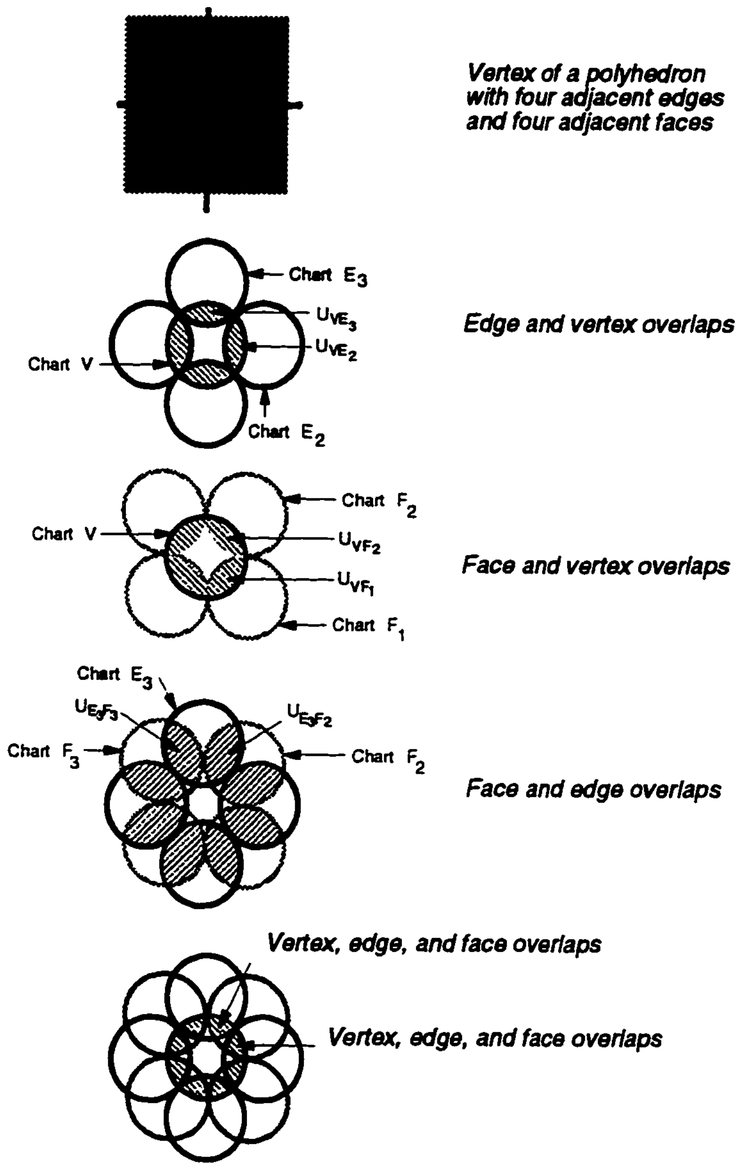


Figure 4.3: Top: A vertex v and adjacent elements $e_0, \ldots, e_3$ and $f_0, \ldots, f_3$. Bottom: The charts and overlaps resulting from those elements.

seven sides. (This second limitation is necessary mostly for practical reasons, not mathematical ones.) For the following discussion we restrict ourselves to polyhedron without boundary; in Chapter 7 we extend the construction to polyhedron with boundary.

Using the polyhedron as a guide we define a finite number of charts, the overlap regions between the charts, and the transition functions. This process exactly parallels the construction outlined in Section 3.2 for a curve. The basic relationship between the polyhedron and the constructed atlas is this:

- The charts: There is one chart in the atlas for each element in the polyhedron. An aside on notation: a chart corresponding to a vertex v is called a *vertex chart* and is labeled V. Similarly, we use E for an edge chart and F for a face chart.

- The overlap regions: Two charts in the atlas overlap if and only if their respective elements in the polyhedron are adjacent to each other. This implies that each vertex chart overlaps with four edge charts and four face charts.[3] Each edge chart overlaps with two vertex charts and two face charts. A vertex chart never overlaps with another vertex chart, and similarly for edge and face charts.

- The transition functions: These functions are empty unless the two charts overlap. There are three distinct cases in which charts overlap: vertex and edge charts, vertex and face charts, and edge and face charts. Within one of these cases the individual functions are all similar; therefore we define three basic types of transition functions:
  - edge-to-face transitions (a rotation and a translation)
  - vertex-to-face transitions (a stretching or shrinking)
  - edge-to-vertex transitions (a composition of the above functions)

We first walk through how a set of charts in the area around a vertex relate to each other in an abstract sense (see Figure 4.3). Then the actual shapes of the charts and overlap regions are defined, along with the transition functions. A formal description of the charts and transition functions is given in Section 5.2.

4.3.1 A vertex chart and its neighbors

Figure 4.3 shows a vertex with the adjacent four edges and faces. It also shows a schematic of the charts generated from this bit of the polyhedron and how they overlap. Note that this is just a sketch of the overlaps; the real shapes of the charts are not given. In Figure 4.3

- The vertex chart overlaps with the four edge charts.

- The vertex chart also overlaps with the four face charts.

[3] Recall that each vertex has valence four.

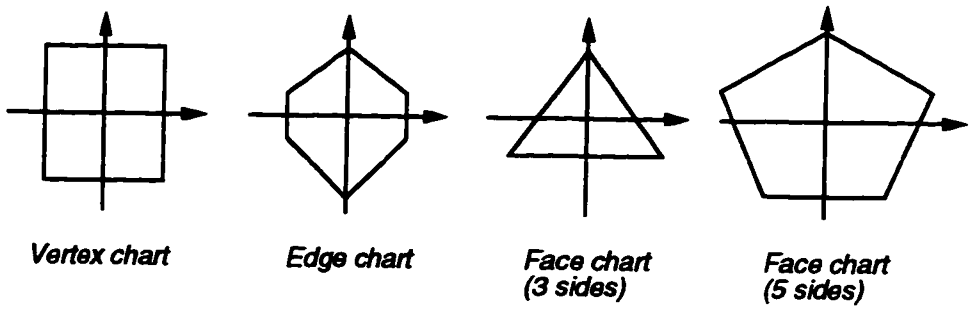


Figure 4.4: A sketch of the different chart types.

- Each edge chart overlaps with two of the face charts.
- Each face chart overlaps with two of the edge charts. (Note that an n-sided face chart overlaps with n edge charts; in this example the remaining edge charts are not shown).

Defining consistent transition functions on areas where only two charts overlap is fairly straightforward: define $\varphi_{CC'}$ to be an invertible, well behaved function and then define $\varphi_{C'C}$ to be its inverse. For example, we define the transition function between an edge chart and a face chart to be a translation followed by a rotation (see Figure 4.5). The transition function from the face chart back to the edge chart is therefore a rotation followed by a translation.

A difficulty arises when defining a transition function on an area where more than two charts overlap. In this case the co-cycle condition must also be satisfied (see Figure 4.7). The only case in which this is a problem involves exactly three charts: one each of a vertex, edge, and face chart. A simple way to satisfy the co-cycle condition is to define two of the functions, say the edge-to-face function and the face-to-vertex function, then define the third function (the edge-to-vertex one) by the composition of the other two (see the top half of Figure 4.7). This almost suffices, but each overlap region needs two such composed functions: one for the upper half and one for the lower half. There is no guarantee that the two composed functions will agree on the boundary between them. To solve this problem, we create a "gap" between the areas where three charts overlap and blend between the two composed functions.

4.3.2 The charts, overlaps, and transition functions

Figure 4.4 shows the three different chart types and their approximate shapes (unit square, diamond, and regular polygon). Exact details are given in Section 5.3. The diamond of the edge chart is constructed by joining together two "wedges" from unit polygons; which unit polygons are used depends on the number of sides of the faces adjacent to the edge. The overlap regions are summarized below.

- Vertex charts: Each quadrant of a vertex chart overlaps a face chart. Each side of the square overlaps an edge chart.

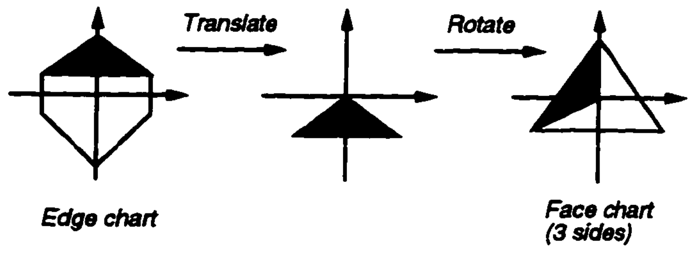


Figure 4.5: The transition function between an edge and a face chart consists of a translation followed by a rotation.

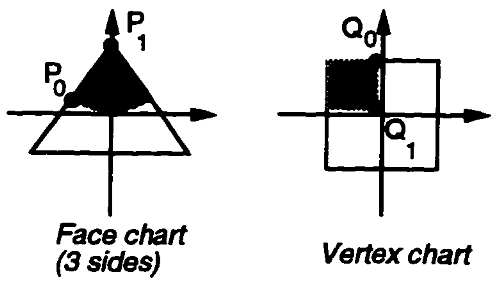


Figure 4.6: The transition function between a face and a vertex chart is a projective transform.

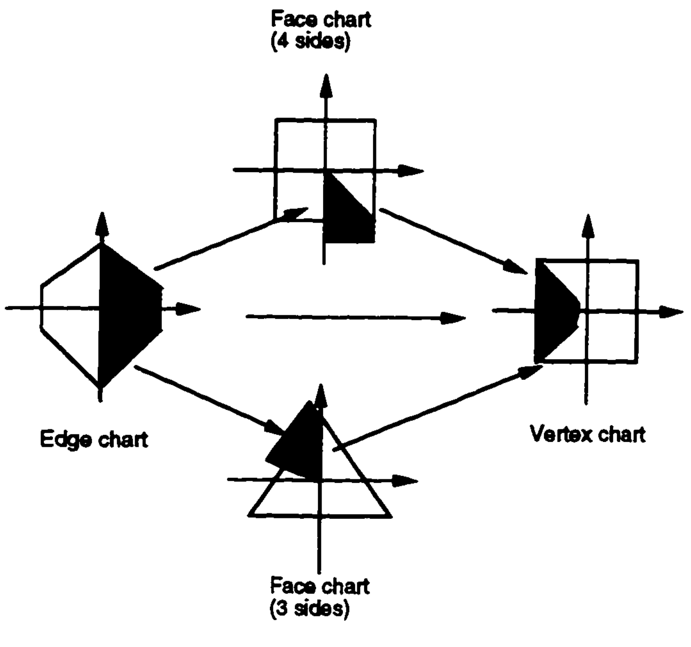


Figure 4.7: The transition function between an edge and a vertex chart is a blend of the two functions formed by composing the map from the edge chart to one of the face charts with the map from the face chart to the vertex chart.

- Edge charts: One face chart overlaps the top half of the edge chart, another overlaps the bottom half. One vertex chart overlaps the left half of the diamond, the other vertex chart the right half.
- Face charts: Each corner of the face chart overlaps a vertex chart. Each side of the face chart overlaps an edge chart.

The transition functions are as follows:

- Edge-to-face: a translation followed by a rotation (Figure 4.5).
- Vertex-to-face: the quadrant is distorted using a projective map (Appendix A) to fit in the corner of the face chart (Figure 4.6).
- Edge-to-vertex: a blend of the two composed functions defined on the upper and lower halves of the edge chart (Figure 4.7).

Chapter 5

Formal description of a manifold

This section contains two definitions; the first is the traditional analytic definition of a manifold, the second is a formal definition of the synthetic approach, i.e., building a manifold from a set of charts. Following this is a formal construction of a manifold from a polyhedron using the second definition.

5.1 Traditional manifolds

Manifolds were introduced in the 1890s and formalized in the 1920s in order to describe objects whose topology was more complicated than that of Euclidean space. The notion was that an object "locally like" Euclidean space could be studied in much the same way as Euclidean space. In one view, a manifold is a structure imposed on an object – a division of that object into overlapping regions, each of which corresponds nicely with a portion of the Euclidean plane. As a concrete example, consider once again a world atlas. As we have seen, every point on the world can be found in at least one chart in the atlas and sometimes in several. A path from one point to another can be found by tracing a line through the charts. Where the path must cross from one chart to another, the two charts overlap enough that one can locate oneself on the second chart. The individual charts are regions of $\Re^2$ but taken together they represent a sphere [MYV93]. There are also implicitly defined maps from one page to another. Thus Brussels and its environs may appear on two different atlas pages: the page for the Benelux countries and also in the upper right corner of the page for France. The labels for Brussels and the surrounding towns, etc., establish a correspondence between the upper right corner of the page for France and the lower left corner of the Benelux page.

The following definition of a manifold is taken from [ST67].

Definition 1 *A* C^k-differentiable manifold *of dimension* n *is a pair* (X, Φ) *where* X *is a Hausdorff topological space and* Φ *is a collection of maps such that the*

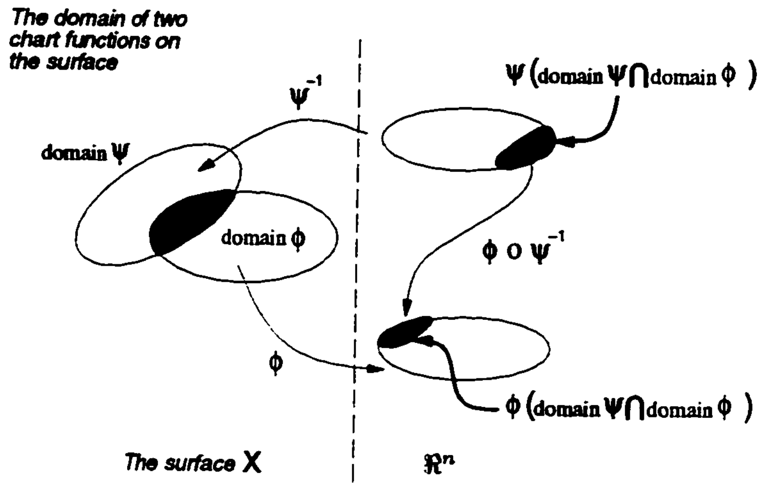

The domain of two chart functions on the surface
ψ^{-1}
ψ (domain $\psi \cap$ domain ϕ)
domain ψ
domain ϕ
$\phi \circ \psi^{-1}$
ϕ
ϕ (domain $\psi \cap$ domain ϕ)
The surface X
$\Re^n$

Figure 5.1: Two overlapping maps ϕ and ψ.

conditions listed below hold. (See Figure 5.1.)

1. *$\{domain\ \phi\}_{\phi \in \Phi}$ is an open covering of X.*
2. *each $\phi \in \Phi$ maps its domain homeomorphically onto an open set in $\Re^n$.*
3. *for each $\phi,\psi \in \Phi$ with (domain $\phi \cap$ domain ψ) $\neq \emptyset$, the map $\psi \circ \phi^{-1}$ is a C^k map from ϕ(domain $\phi \cap$ domain ψ) $\subset \Re^n$ into ψ(domain $\phi \cap$ domain ψ) $\subset \Re^n$.*
4. *Φ is maximal relative to (2) and (3); that is, if ψ is any homeomorphism mapping an open set in X onto an open set in $\Re^n$ such that, for each $\phi \in \Phi$ with (domain $\phi \cap$ domain ψ) $\neq \emptyset$, $\psi \circ \phi^{-1}$ and $\phi \circ \psi^{-1}$ are C^k maps from ϕ(domain $\phi \cap$ domain ψ) to ψ(domain $\phi \cap$ domain ψ) and from ψ(domain $\phi \cap$ domain ψ) to ϕ(domain $\phi \cap$ domain ψ), respectively, then $\psi \in \Phi$.*

The first condition ensures that every point in X is contained in the domain of at least one ϕ. The second condition ensures that the domain of each ϕ is homeomorphic to a ball in $\Re^n$. The third condition ensures that the functions are consistent. The last condition is largely technical; it requires that every possible valid map be in the atlas.

This definition relates to the informal discussion in Section 4.1 in the following way:

- (domain ϕ) is the bit of the world that the atlas page covers (the red square painted on the world in Figure 3.5).
- ϕ is the mapping from the bit of the world to the rectangular page (the red arrow taking the red square to the atlas page).

- Condition 1 above ensures that every point on the surface X shows up on some page in the atlas.
- Condition 2 above ensures that the maps are "nice", i.e., they do not fold or crease.
- Condition 3 above ensures that when more than one map covers a spot on the world then the two overlapping maps are consistent, i.e., we can find Brussels and its environs on both the page for France and the page for Benelux.
- Condition 4 is violated in the world atlas example. To satisfy it, every possible atlas page would have to be included. That means that a map for every town, every building in every town, every street, etc., must also be in the atlas.

5.2 Constructive manifold

Definition 1 above takes an object X and covers it with local mappings to $\Re^n$. Because our approach to building a surface has no object X *a priori* (if we had the surface, we would be done), we describe a manifold not in terms of an object, but instead in terms of regions of $\Re^n$ and transition functions between these regions. (We call these regions *charts.*[1]) In Section 5.2.1 we show that this definition is equivalent to the traditional one. To continue the analogy with an ordinary atlas: an atlas enables one to imagine the world as a whole even if the world does not exist *a priori.* (In fact, many video games do exactly this: the fictitious world through which the player moves is not a model of any real place.)

We now give a formal definition of the kind of object we described informally in Section 4.2 and then show that this object is a manifold by Definition 1. This definition proceeds in two steps: we begin by defining a *proto-manifold* (charts and transition functions) and then build a manifold from this proto-manifold using an equivalence relation ("gluing" the charts together).

Definition 2 A C^k-differentiable proto-manifold K of dimension n *consists of the following:*

1. *A finite set A of open sets in $\Re^n$. A is called the* proto-atlas. *Each element $c \in A$ is called a* chart.
2. *For every pair of charts $c, c' \in A$, the subset $U_{cc'} \subset c$. If $c = c'$ then $U_{cc'} = c$.*
3. *A set of functions Φ, called* transition functions. *For every pair of charts $c, c' \in A$ the transition function $\varphi_{cc'} \in \Phi$ is a map $\varphi_{cc'} : U_{cc'} \rightarrow U_{c'c}$. Note that $U_{cc'}$ and $U_{c'c}$ may well be empty. The following conditions on the transition functions must hold:*

   (a) $\varphi_{cc'}$ is one-to-one, onto, and C^k-differentiable.

[1] This usage differs from the usual definition of a chart on a surface, which is a *map* from the surface to $\Re^n$.

(b) $\varphi_{cc'}^{-1} = \varphi_{c'c}$

(c) $\forall x \in U_{CC}$, $\varphi_{cc}(x) = x$,

(d) $(\varphi_{jk} \circ \varphi_{ij})(x) = \varphi_{ik}(x)$ *for* $x \in U_{ik} \bigcap U_{ij}$. *(This is called the* co-cycle condition *- see Figure 4.2).*

4. *Let* c *and* c' *be two charts in* A. *We define a relation on the disjoint union of the two charts,* $c \sqcup c'$, *as follows: if* $x \in c$, $y \in c'$, *and* $y = \varphi_{cc'}(x)$ *then* $x \sim y$. *(This is an equivalence relation because* $\varphi_{cc'} = (\varphi_{c'c})^{-1}$.*) Let* $K_{cc'}$ *be the quotient of* $c \sqcup c'$ *by* $\sim$. *Then there must exist an embedding* $\mathcal{E}_{cc'}$ *of* $K_{cc'}$ *into* $\Re^n$. Note: The purpose of this condition is to ensure that M^K is a Hausdorff space; it is probably stronger than it needs to be.

This definition relates to the discussion in Section 4.2 in the following ways:

1. The charts are the same, i.e., open disks in the plane.

2. An overlap region ($U_{cc'}$) tells what part of chart c overlaps with chart c' (see Figure 4.1). We need this information for every pair of charts in the proto-atlas (although many of the overlap regions may be empty).

3. A transition function ($\varphi_{cc'}$) tells how to map between the overlap regions $U_{cc'}$ and $U_{c'c}$. These transition functions are defined for every pair of charts (although many of them may be the empty function). The conditions imposed on the functions ensure the following:

   - The mappings are "nice," i.e., no folding.
   - The map taking a chart to itself is the identity function.
   - Going from $U_{cc'}$ to $U_{c'c}$ and back again puts you back where you started (see Figure 4.1).
   - Where several maps overlap they must be consistent. Following the transition functions through the charts puts you back where you started.

4. The two charts glued together (the quotient of $c \sqcup c'$ by the equivalence relation) is embeddable in $\Re^n$.

Next we define a relation that relates points in different pages (i.e., the point "Brussels" on the France page with the point "Brussels" on the Benelux page). This relation dictates how to glue the pages together. If $K = (A, \Phi)$ is a proto-manifold then we can define a relation $\sim$ on the disjoint union of the charts, $\sqcup_{(c \in A)} c$, such that if $x \in c$, $y \in c'$ then $x \sim y$ iff $\varphi_{cc'}(x) = y$. Conditions (1) through (3) in the definition of a proto-manifold ensure that $\sim$ is an equivalence relation (see Appendix C.1).

The relation $\sim$ lets us build a single object from the charts. We declare that if a point x in one chart is taken via $\varphi_{cc'}$ to a point y in another chart ($\varphi_{cc'}(x) = y$) then those two points become a single point on the final object. Continuing the analogy of a world atlas, each chart c is a page of the world atlas, each transition function $\varphi_{cc'}$ is a correspondence between parts of two pages, and the equivalence relation $\sim$ says that "the place labeled Brussels on page 93 is the same as the place labeled Brussels on page 24." The following definition describes the resulting object:

Definition 3 *Let $\sim$ be the equivalence relation described above and K be a proto-manifold as defined above. M^K denotes the quotient of $\sqcup_{(c \in A)} c$ by $\sim$ and Π_K denotes the map taking $x \in \sqcup_{(c \in A)} c$ to $[x] \in M^K$, where $[x]$ is the equivalence class of x.*

We now build a *topology* on M^K by defining a basis for that topology [Mun75]. This topology defines what the open sets in M^K are. For the remainder of this paper, when we refer to the topology of M^K we refer to the topology of the following definition:

Definition 4 *The basis elements $\mathcal{B}$ of M^K are defined to be $\Pi_K(b)$ where b is an open set in some chart c.*

Appendix C.2 contains a proof that M^K is a Hausdorff space, i.e., an object of the type needed in Definition 1.

To show that this object is a manifold we need to verify that M^K satisfies Definition 1. In a sense, these two definitions are inverses of each other; Definition 3 builds M^K from a set of charts, and Definition 1 takes M^K and constructs charts on it. To prove that M^K is a manifold we build maps to $\Re^n$ on M^K; since we have the charts M^K was built from, we can use those charts as a starting point.

5.2.1 Proof that M^K is a manifold

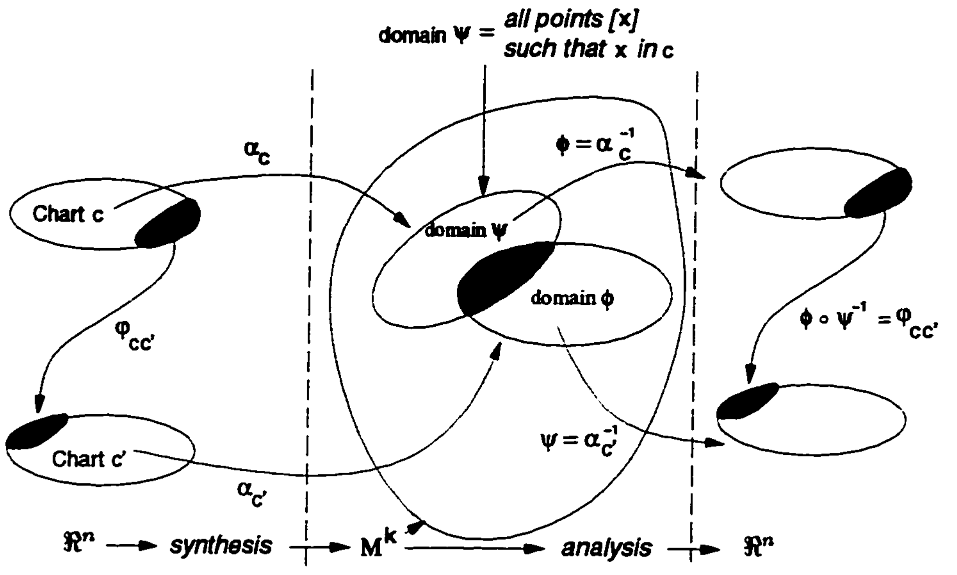
In this section we show that we can construct a manifold structure on M^K by first constructing an atlas $\mathcal{A}$ of C^k-related charts [Spi70] on M^K. We then show that M^K, together with the unique maximal atlas $\mathcal{A}'$ containing $\mathcal{A}$, is a manifold by Definition 1.

Theorem 1 *Let $K = (A, \Phi)$ be a proto-manifold and M^K be as defined above. For $c \in A$, define $\alpha_c : c \to M^K$, as the restriction of Π_K to c,* i.e.,

$$\alpha_c : c \to M^K : x \to \Pi_K(x)$$

For each c define the function $\phi_c : Im(\alpha_c) \subset M^K \to c$ as follows:

$$\phi_c = \alpha_c^{-1}$$



domain ψ = all points [x] such that x in c
α_c
Chart c
$\varphi_{cc'}$
Chart c'
$\alpha_{c'}$
$\phi = \alpha_c^{-1}$
domain ψ
domain ϕ
$\psi = \alpha_{c'}^{-1}$
$\phi \circ \psi^{-1} = \varphi_{cc'}$
$\Re^n$ → synthesis → M^k → analysis → $\Re^n$

Figure 5.2: Going from the charts of Definition 3 to M^K to the chart functions of Definition 1.

Let $\Phi = \{\phi_c : c \in A\}$. Then $\mathcal{A} = (M^K, \Phi)$ is a C^k-related atlas.

Aside 3 Here is what the theorem says: *An atlas of C^k-related charts is very similar to the atlas described in Definition 1 except the maps between the bits of plane $(\phi \circ \psi^{-1})$ are only guaranteed to be C^k-continuous and the atlas is not maximal,* i.e., *not every possible atlas page is included. To prove that we can construct a C^k-related atlas, we take the charts M^K was built from and make a set of maps to $\Re^n$ (*i.e., *chart functions) from them by taking inverses. Figure 5.2 shows the charts M^K was built from on the left, M^K in the middle, and the chart functions of Definition 1 on the right.*

Proof:

First we show that α_c is one-to-one (it is trivially a map onto its image) and therefore ϕ_c is well-defined. Let x, $x' \in c$. If $\alpha_c(x) = \alpha_c(x')$ then x and x' are in the same equivalence class $\Pi_K(x) = \Pi_K(x')$ so $x \sim x'$. By the definition of $\sim$, with both charts being c, the map ϕ_{cc} satisfies $\phi_{cc}(x) = x'$. The function ϕ_{cc} is defined to be the identity, so $x = x'$. Hence α_c is one-to-one. So ϕ_c can be defined on $Im(\alpha_c)$, and is one-to-one and onto its image.

To show that ϕ_c is continuous and hence a homeomorphism, we need to show that for every open set $U \subset c$, $(\phi_c)^{-1}(U)$ is an open set. This is true by the definition of open sets on M^K (see Definition 4).

To show that the ϕs are C^k-related we need to show that for any two charts c, c' the following two maps are C^k:

$$\begin{aligned}\phi_c \circ \phi_{c'}^{-1} &= \phi_c \circ \alpha_{c'} = \varphi_{c'c} \\ \phi_{c'} \circ \phi_c^{-1} &= \phi_{c'} \circ \alpha_c = \varphi_{cc'}\end{aligned}$$

But $\varphi_{c'c}$ and $\varphi_{cc'}$ are by hypothesis C^k differentiable. □

We next need to show that there is a maximal atlas $\mathcal{A}'$ containing $\mathcal{A}$ and that this maximal atlas, together with M^K, satisfies Definition 1. ($\mathcal{A}'$ is needed to satisfy Condition four of Definition 1; otherwise $\mathcal{A}$ would be sufficient.) The following lemma is from [Spi70]:

Lemma 1 *If A is an atlas of C^k-related charts on Hausdorff space M, then A is contained in a maximal atlas $\mathcal{A}'$ for M.*

Prop 1 *The pair $(M^K, \mathcal{A}')$ is a manifold by Definition 1.*

Proof:

Condition 1 of Definition 1 is satisfied because $\mathcal{A} \subset \mathcal{A}'$ and A covers M^K. Conditions 2 and 3 are satisfied by the construction of the maximal atlas. The last condition is trivially satisfied since $\mathcal{A}'$ is maximal. □.

5.3 Building a manifold: the charts and the transition functions

This section gives the specifics for constructing a manifold from a polyhedron without boundary. Extending this construction to a polyhedron with boundary is covered in Section 7. See Appendix B for the notation used for polyhedra.

5.3.1 The charts

Recall that the proto-manifold associated with a polyhedron is defined by a set of charts (the proto-atlas), their overlaps, and transition functions between the overlap regions.

The proto-atlas associated with the polyhedron is defined as follows. There is a chart in the atlas for each element in the polyhedron. We label the correspondences between them as follows: to each vertex $v \in \mathcal{V}$ we associate a chart C_v; the set of all vertex charts is denoted by $\mathbf{V} = \{C_v\}_{v \in \mathcal{V}}$. Similarly, the set of all edge charts is denoted by $\mathbf{E} = \{C_e\}_{e \in \mathcal{E}}$ and the set of all face charts by $\mathbf{F} = \{C_f\}_{f \in \mathcal{F}}$. The entire proto-atlas A is then $\mathbf{V} \bigcup \mathbf{E} \bigcup \mathbf{F}$.

Aside 4 A note on notation: *Henceforth, the letter V denotes the vertex chart associated with the vertex v, similarly, E denotes an edge chart and F a face chart.*

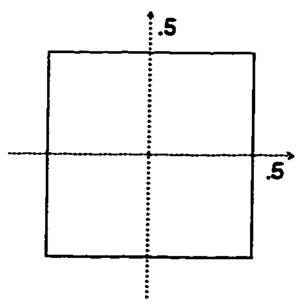


Figure 5.3: A vertex chart is a unit square without boundary.

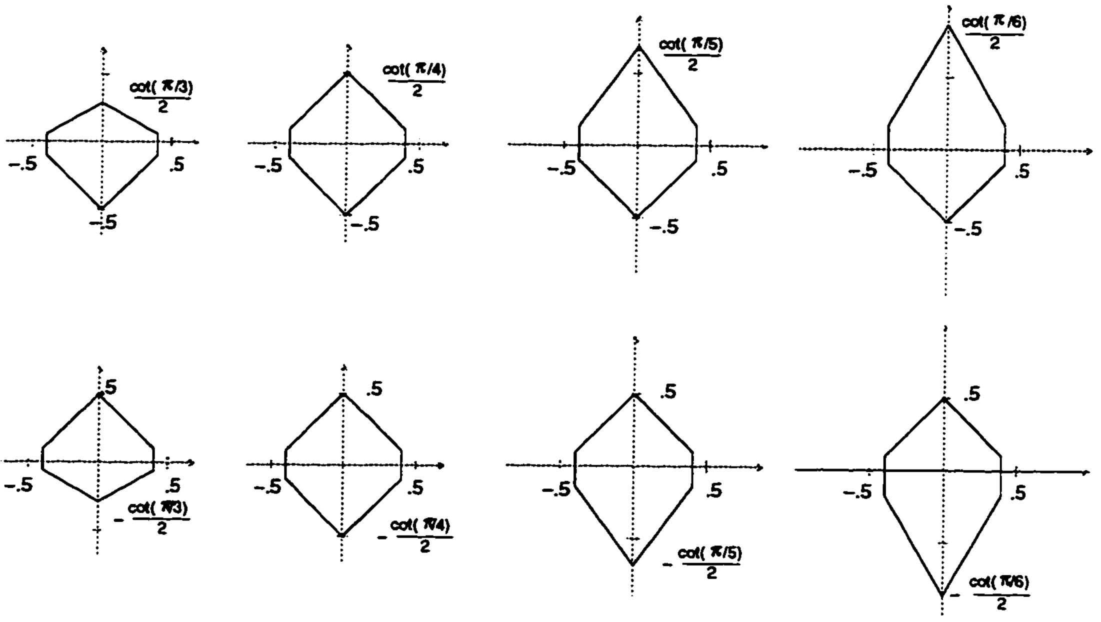


Figure 5.4: Example shapes of the edge charts. Top, left to right: The upper face has three, four, five, and six sides, the lower face has four sides. Bottom, left to right: The lower face has three, four, five, and six sides, the upper face has four sides.

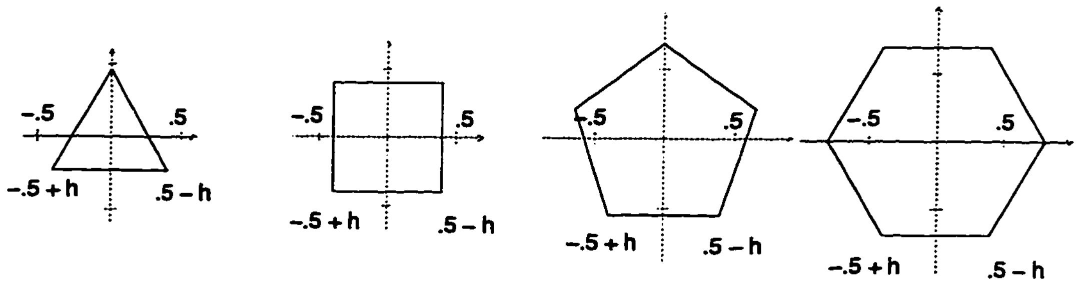


Figure 5.5: The face charts are regular polygons centered at the origin with edge length $1-2h$. For these figures, $h = 0.1$. From left to right the corresponding face has three, four, five, and six sides.

Each vertex chart is a unit square without boundary (see Figure 5.3).

Each edge chart is a diamond with two corners chopped off (see Figure 5.4). Let h be a constant (we discuss the choice of h in Section 8.1). Then the width from the left edge to the right edge is $1-2h$. The diamonds may be asymmetrical in the vertical direction; the heights of the top and bottom halves are determined by the number of sides of the faces adjacent to the edge. Let f_u and f_l be the two faces adjacent to the edge e corresponding to the edge chart E. Let n_u be the number of sides of f_u, and similarly for n_l. Then the chopped diamond is

$$\begin{array}{ccc} & (0, \frac{\cot(\pi/n_u)}{2}) & \\ (-.5+h, h\cot(\pi/n_u)) & & (.5-h, h\cot(\pi/n_u)) \\ (-.5+h, -h\cot(\pi/n_l)) & & (.5-h, -h\cot(\pi/n_l)) \\ & (0, -\frac{\cot(\pi/n_l)}{2}) & \end{array}$$

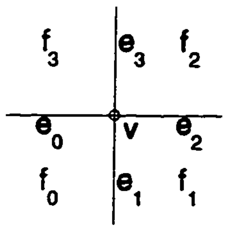
Each face chart is a regular polygon without boundary that is centered at the origin (see Figure 5.5). Let h be the constant given above and let f be the face corresponding to the face chart F. Then F is a regular polygon with the same number of sides as f and edge length $1-2h$.

5.3.2 The overlap regions

The chart overlaps are determined by the adjacency relationships in the polyhedron. Charts within a set never overlap. For example, if V, $V' \in \mathbf{V}$ are two different vertex charts then $U_{VV'} = U_{V'V} = \emptyset$. Two charts have a non-empty overlap if and only if their respective elements in the polyhedron are adjacent. This is summarized below:

- $U_{VV'} = \begin{cases} \emptyset & V \neq V' \\ V & V = V' \end{cases}$

- $U_{EE'} = \begin{cases} \emptyset & E \neq E' \\ E & E = E' \end{cases}$



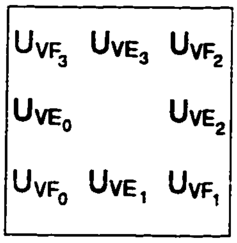


Figure 5.6: Left: The elements of the polyhedron adjacent to the vertex v. Right: The locations of the U_{VF_t} and U_{VE_t} within the vertex chart.

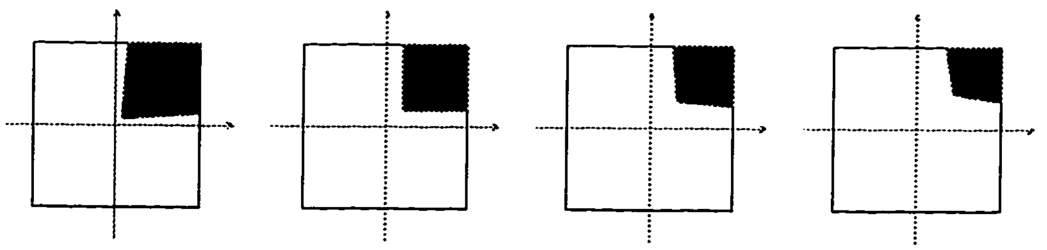


Figure 5.7: The shape of U_{VF_2} for $h = 0.1$. From left to right: Face f_2 has three, four, five, and six sides.

- $U_{FF'} = \begin{cases} \emptyset & F \neq F' \\ F & F = F' \end{cases}$
- $U_{VE} \neq \emptyset$ if and only if $v \in e$.
- $U_{VF} \neq \emptyset$ if and only if v is a vertex of f.
- $U_{EF} \neq \emptyset$ if and only if e is an edge of f.

Vertex chart overlaps. Each vertex chart V overlaps with exactly four face charts $F_0, \ldots, F_3$ and 4 edge charts $E_0, \ldots, E_3$. If the corresponding vertex v and its surrounding elements are labeled as on the left of Figure 5.6, then the locations of the overlap regions are as shown on the right. The U_{VF_t} overlap the quadrants of V and the U_{VE_t} overlap the sides of V. The overlap region U_{VF_t} is defined to be the quadrilateral without boundary formed by taking $\varphi_{F,V}$ of the four corners defining U_{VF_t} (see Figure 5.7). The overlap region U_{VE_t} is defined to be $\varphi_{E,V}(U_{E,V})$. Examples of these regions are shown in Figure 5.8.

Edge chart overlaps. Each edge chart E overlaps two vertex charts V_l and V_r and two face charts F_u and F_l. If the corresponding edge e is labeled as on the left of Figure 5.9, then the overlap regions are located as shown on the right. The overlap region U_{EV_l} is the left half of the edge chart, i.e., all points $(x, y) \in E$ such that $x < 0$. Similarly, U_{EV_r} is the right half of the edge chart (see Figure 5.10). The overlap region U_{FE_u} is all points $(x, y) \in E$ such

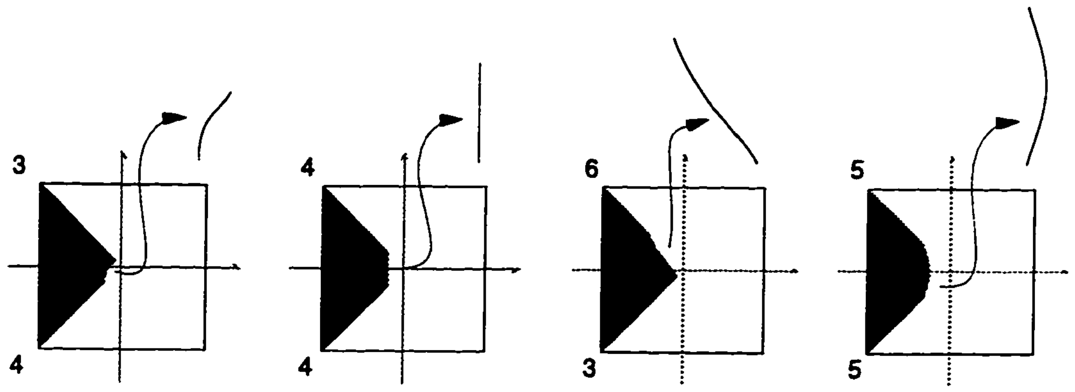


Figure 5.8: Example shapes of U_{VE_0} for $h = 0.1$; F_3 and F_0 have the number of sides indicated.

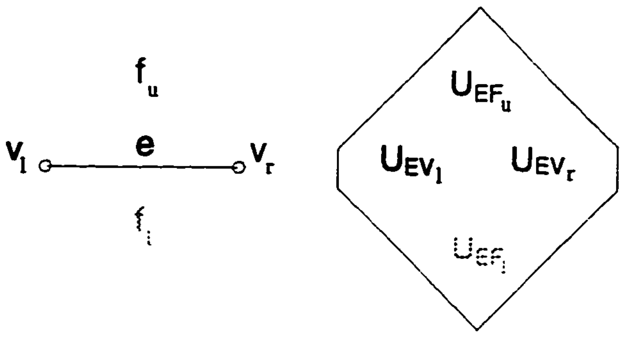


Figure 5.9: Left: The elements of the polyhedron adjacent to the edge e. Right: The locations of the U_{EV_i} and U_{EF_i}.

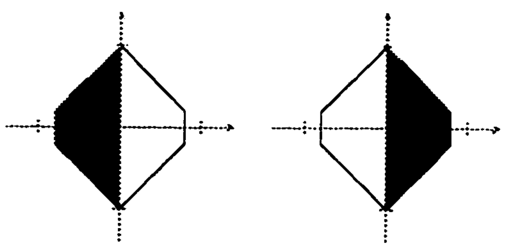


Figure 5.10: The shape of U_{EV_l} and U_{EV_r}; both adjacent faces have four sides.

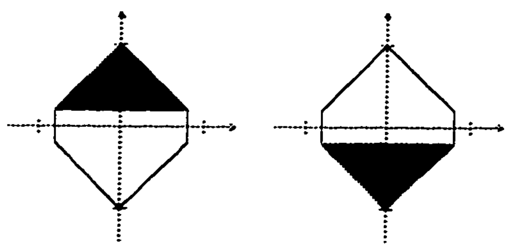


Figure 5.11: The shape of U_{EF_u} and U_{EF_l}; both adjacent faces have four sides.

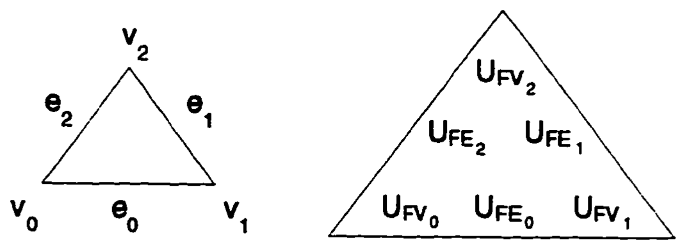


Figure 5.12: Left: The elements of the polyhedron adjacent to face f. Right: The locations of the U_{FV_i} and U_{FE_i}.

that $y > h\cot(\pi/n_u)$. Similarly, U_{EF_l} is all points $(x, y) \in E$ such that $y < -h\cot(\pi/n_l)$ (see Figure 5.11).

Face chart overlaps. Each face chart F with n sides overlaps with n vertex charts $V_0, \ldots, V_{n-1}$ and n edge charts $E_0, \ldots, E_{n-1}$. If the corresponding face f is labeled as on the left in Figure 5.12, then the overlap regions are positioned as shown on the right. Each region U_{FV_i} is a quadrilateral without boundary. The edges of the quadrilateral are formed by joining the midpoints of the edges to the origin (see Figure 5.13). The overlap region U_{FE_i} is a triangle without boundary formed by joining the origin to two adjacent corner points (see Figure 5.14).

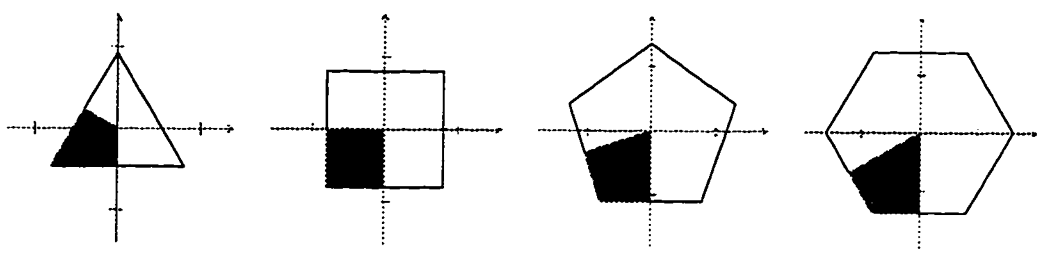


Figure 5.13: The shape of U_{FV_0} for $h = 0.1$. From left to right: The corresponding face has three, four, five, and six sides.

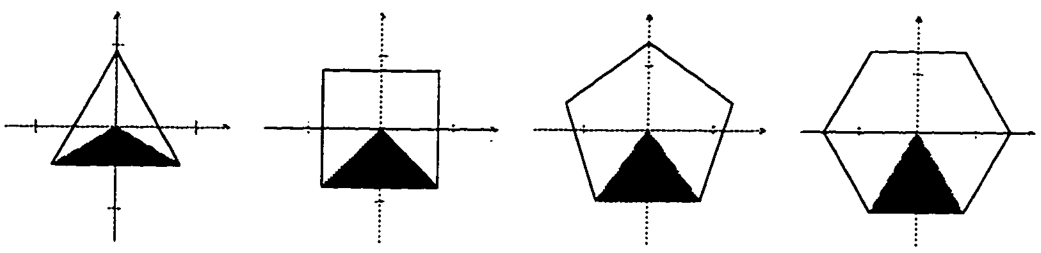


Figure 5.14: The shape of U_{FE_0} for $h = 0.1$. From left to right: The corresponding face has three, four, five, and six sides.

5.3.3 The transition functions

As described in Section 4.3, the non-trivial transition functions fall into one of three categories: edge-to-face, face-to-vertex, and edge-to-vertex transition functions (the transition functions φ_{cc}, $c \in \mathcal{A}$ are defined to be the identity function). We define them in that order.

The edge-to-face transition function φ_{EF}. This function breaks into two cases depending upon whether the face overlaps the upper or lower half of the edge chart. Let E, F_u, F_l, and the overlap regions be as shown in Figure 5.15.

The function φ_{EF_u} translates U_{EF_u} to the origin, then rotates it to the correct edge of F_u. Let n_u be the number of sides of F_u. The amount of translation is $d_u = -1/2\cot(\pi/n_u)$ and the amount of rotation is some multiple of $\theta_u = (2\pi)/n_u$. (The multiple depends on the edge of F_u overlapped by the chart E; if the edges are labeled as in Figure 5.12, then the multiple is the edge number.):

$$\varphi_{EF_u}(s,t) = \{s\cos(\theta_u) - (t-d_u)\sin(\theta_u), (t-d_u)\cos(\theta_u) + s\sin(\theta_u)\} \tag{5.1}$$

The function φ_{EF_l} translates U_{EF_l} to the origin, then rotates it to the correct edge of F_l. Let n_l be the number of sides of F_l. The amount of translation is $d_l = 1/2\cot(\pi/n_l)$ and the amount of rotation is π plus some multiple of $\theta_l = (2\pi)/n_l$ (as above, the multiple is the edge number):

$$\varphi_{EF_l}(s,t) = \{s\cos(\theta_l+\pi) - (t+d)\sin(\theta_l+\pi), (t+d_l)\cos(\theta_l+\pi) + s\sin(\theta_l+\pi)\}$$

To invert these functions, first rotate and then translate.

The face-to-vertex transition function φ_{FV}. Let V, F, and the overlap regions be as shown in Figure 5.16. The function φ_{FV} "stretches" the quadrilateral U_{FV} to fit into a quadrant of the vertex chart using a projective transformation $\zeta : \Re^2 \to \Re^2$. The effect of the projective transform is illustrated in Figure 5.17: it takes a quadrilateral $q_0, \ldots, q_3$ and performs an edge-preserving map to the quadrilateral $p_0, \ldots, p_3$ (details are in Appendix A).

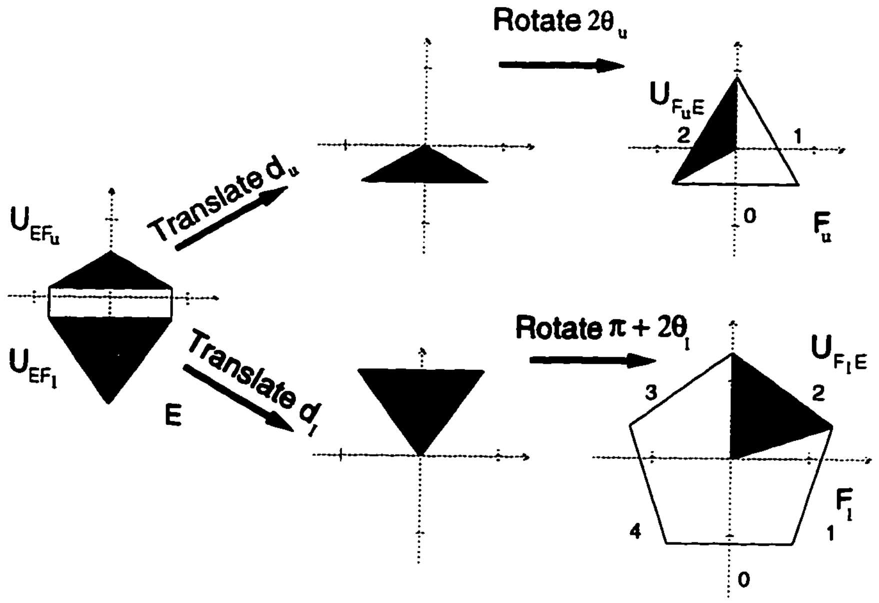


Figure 5.15: The transition functions φ_{EF_u} and φ_{EF_l}. Top: U_{EF_u} is mapped to the second edge of F_u. Bottom: U_{EF_l} is mapped to the second edge of F_l.

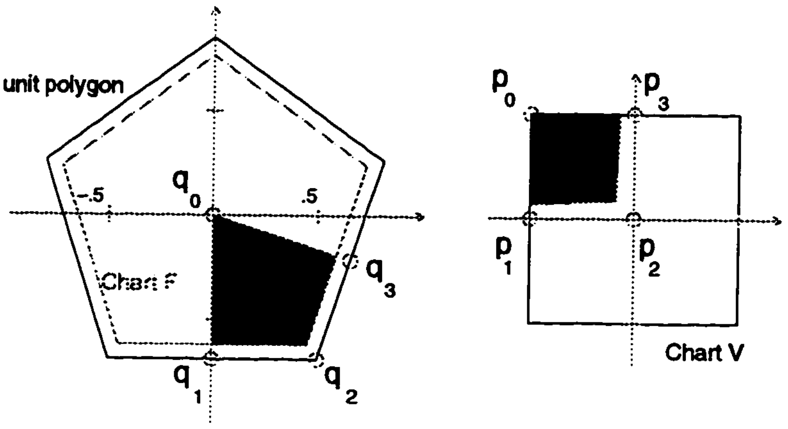


Figure 5.16: The transition function φ_{FV} is a restriction of the projective transform taking the quadrilateral $\{q_i\}$ to the square $\{p_i\}$.

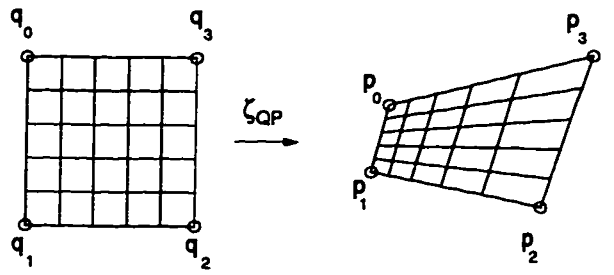


Figure 5.17: The projective transform ζ_{QP} takes the quadrilateral $\{q_i\}$ to the quadrilateral $\{p_i\}$. Note that lines are preserved.

Let the four points $q_0, \ldots, q_3$ be the four vertices of a corner of a unit polygon (q_1 and q_3 lie at the midpoints of the two adjacent edges), as shown in Figure 5.16, and let $p_0, \ldots, p_3$ be the four corners of a quadrant of the vertex chart. Let ζ_{QP} be the unique projective transform taking $q_0, \ldots, q_3$ to $p_0, \ldots, p_3$ (and ζ_{PQ} be the unique projective transform taking $p_0, \ldots, p_3$ to $q_0, \ldots, q_3$). We define the transition function as the projective transform restricted to the appropriate overlap region:

$$\begin{aligned} \forall x \in U_{FV}, \varphi_{FV}(x) &= \zeta_{QP}(x) \\ \forall x \in U_{VF}, \varphi_{VF}(x) &= \zeta_{PQ}(x) \end{aligned}$$

In Appendix A.3 we prove that $\zeta_{QP} = (\zeta_{PQ})^{-1}$, i.e., that $(\varphi_{FV})^{-1} = \varphi_{VF}$.

The edge-to-vertex transition function φ_{EV}. This function is built by blending betwen compositions of the previous two functions. The function is defined by this method, instead of directly, to ensure that the co-cycle condition is satisfied. Let E, V, F_u, F_l, and the overlap regions be as shown in Figure 5.18. The function φ_{EV} is built by blending the following two composed functions:

$$\varphi_{F_uV} \circ \varphi_{EF_u}$$
$$\varphi_{F_lV} \circ \varphi_{EF_l}$$

To define φ_{EV} we first show that $\varphi_{F_uV} \circ \varphi_{EF_u}$ and $\varphi_{F_lV} \circ \varphi_{EF_l}$ can be extended to the shaded region shown in Figure 5.18, i.e., the region between the domains of the two functions. Extending φ_{EF_u} to this region is trivial, since the function φ_{EF_u} was originally defined by restricting a function defined on the entire plane (Eq. 5.1) to the region U_{EF_u} (and similarly for φ_{EF_l}). The function φ_{F_uV} is the restriction of the projective transform; this function is undefined on certain lines of the plane, as shown in Figure A.2. These lines do not intersect the shaded region into which we wish to extend $\varphi_{F_uV} \circ \varphi_{EF_u}$ into and therefore the following definition is valid:

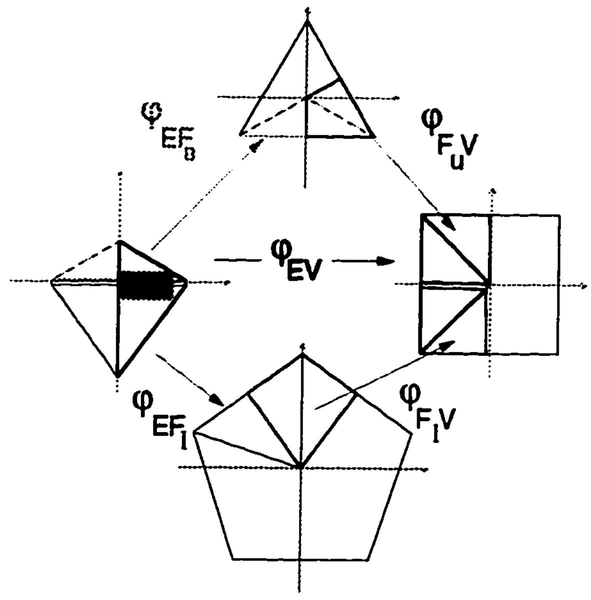


Figure 5.18: The transition function φ_{EV} for $h = 0.1$.

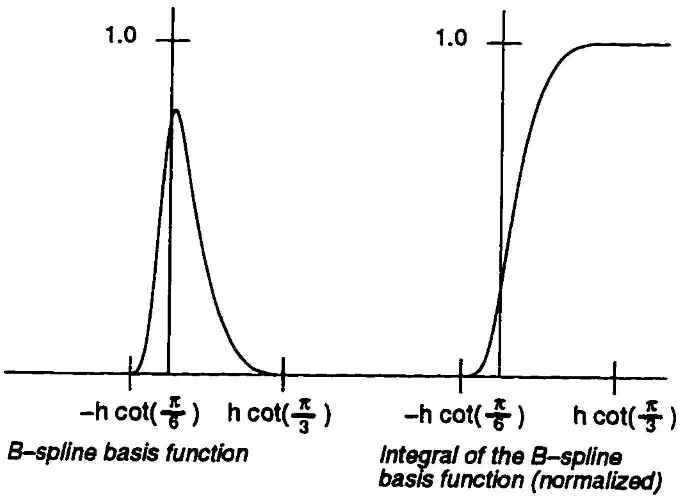


Figure 5.19: Left: The B-spline basis function used to make the blend function. Right: The blend function β for $h = 0.2$ and $k = 2$.

$$\varphi_{EV}(s,t) = (1-\beta(t))\varphi_{F_l V} \circ \varphi_{EF_l} + \beta(t)\varphi_{F_u V} \circ \varphi_{EF_u}$$

where $\beta : \Re \to [0,1]$ is the blend function shown in Figure 5.19. This function has the following properties (where n_u and n_l are the number of sides of the face charts F_u and F_l, respectively):

- $\beta(t) = 0$ for $t < -h\cot(\pi/n_l)$
- $\beta(t) = 1$ for $t > h\cot(\pi/n_u)$
- β is C^k for a given k (the desired continuity of the manifold).
- The derivative of β is bound by the function

$$\beta'(t) \quad < \quad \begin{cases} (\frac{h\cot(\pi/6)+t)}{(h\cot(\pi/6))} & \text{if } t \leq 0 \\ (\frac{h\cot(\pi/6)-t)}{(h\cot(\pi/6))} & \text{if } t > 0 \end{cases} \tag{5.2}$$

- $\beta(t)$ is monotonically increasing.

In Appendix D we show that for $k \geq 0$ the function φ_{EV} is invertible, one-to-one, and onto.

We now show how to construct such a blend function for a given degree $k \geq 0$ from a non-uniform B-spline. Let $r_0 = -h\cot(\pi/n_l)$ and $r_{k+1} = h\cot(\pi/n_u)$. Let $\hat{\beta}$ be the C^{k-1} non-uniform B-spline defined by a knot vector whose endpoints are r_0 and r_{k+1} and whose middle knot is at 0 (if k is even) or whose middle two knots average to 0 (if k is odd).[2] Refer to Eq. 6.3 for the definition of a B-spline. Then

$$\beta(t) = \frac{\int_{r_0}^{t} \hat{\beta}(s)ds}{\int_{r_0}^{r_{k+1}} \hat{\beta}(s)ds}$$

This function is clearly C^k since its derivative is C^{k-1}. Since $\hat{\beta}$ is 0 to the left of r_0 and to the right of r_k, the function β must be 0 to the left of r_0 and constant to the right of r_k (we normalize to ensure this constant is 1). Finally, since $\hat{\beta}$ is everywhere non-negative, the integral in the numerator increases as t increases. The derivative is bounded because the area under the derivative curve is 1, its peak is at or near 0, the width of its support is less than $2h\cot(\pi/6)$, and it is monotonically decreasing to either side of the peak.

5.3.4 Proof that this structure is a proto-manifold

To show that the set of charts and transition functions defined in this section constitute a proto-manifold we need to show that they satisfy the conditions in Definition 2. The most

[2] If $k = 0$ then chose the two knots so that the resulting box function fits under the bounding function 5.2.

difficult part is satisfying the co-cycle condition; informally, we have satisfied this by ensuring that no more than three charts overlap at any point and that when they do, one of the three transition functions is defined to be the composition of the other two.

Prop 2 *The atlas $A = \mathbf{V} \bigcup \mathbf{E} \bigcup \mathbf{F}$, overlap regions, and transition functions defined above form a C^k-differentiable proto-manifold.*

Proof:

We first show that the charts are open sets and the overlap regions are open balls contained in their respective chart.

1. Each chart is defined as an open set in the plane that is homeomorphic to a disk.
2. We need to show that the overlap regions $U_{cc'}$ are homeomorphic to open balls in $\Re^2$. Most of the overlap regions are empty; these trivially satisfy this condition. The overlap regions U_{cc} are defined to be the entire chart c, which satisfies this condition.

   The non-empty overlap regions fall into two classes: those that we define directly (U_{EF}, U_{FV}, U_{VF}, U_{FE}, and U_{EV}) and those that are defined by the transition functions (U_{VE}). In the first case, the overlap regions are homeomorphic to open balls in the charts by their definitions.

   To show that U_{VE} is an homeomorphic to an open ball we use the following theorem [Mun75]:

   > **Theorem 2** *(Brouwer theorem on invariance of domain for $\Re^2$). If U is an open subset of $\Re^2$ and $f : U \rightarrow \Re^2$ is continuous and injective, then $f(U)$ is open in $\Re^2$ and f is an embedding.*

   Since the transition function φ_{EV} is continuous and injective (see Appendix D), by the theorem the overlap region U_{VE} is an open set and the transition function φ_{EV} is an embedding. To show that U_{VE} is homeomorphic to an open ball, we note that there exist a homeomorphism of U_{EV} to an open ball and a homeomorphism from U_{VE} to U_{EV}, and any composition of two homeomorphisms is also a homeomorphism.

   We still need to show that $U_{VE} \subset V$. It suffices to show that the two composed functions making up φ_{EV} both map their extended regions into the vertex chart; since the vertex chart is convex any convex combination of points in the image of the two composed functions is also in the vertex chart. Figure 5.20 shows these extended regions.

We now show that the transition functions satisfy the consistency conditions of Definition 2. The non-trivial transition functions are those defined between charts of different types whose elements are adjacent; the following discussion assumes the transition functions discussed are non-empty. (The empty functions trivially satisfy the conditions of Definition 2.)

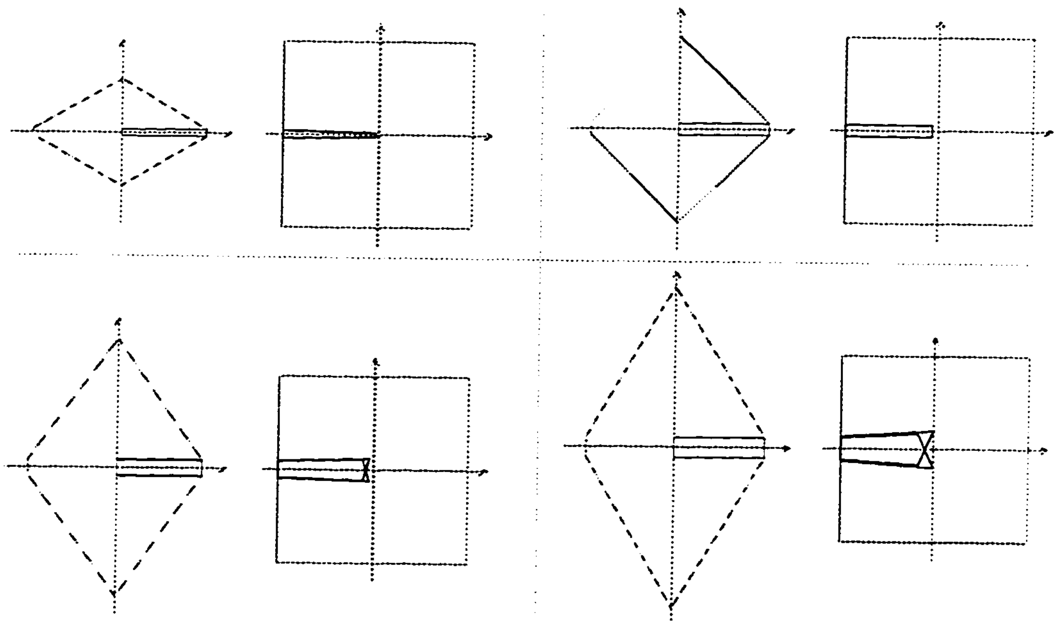


Figure 5.20: The limits of the blend regions mapped into a vertex chart via the upper and lower composed functions ($k = 0$). From top left to bottom right, the faces have three, four, five, and six sides.

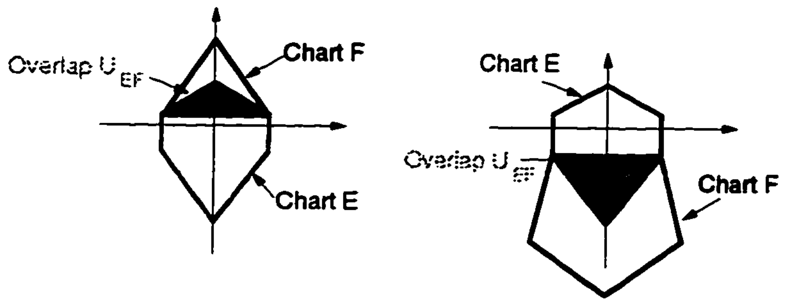


Figure 5.21: Embedding an edge-face pair. Left: The face chart overlaps the upper edge of edge chart. Right: The face chart overlaps the lower edge of edge chart.

We first show that the transition functions are one-to-one, onto, and C^k differentiable. Secondly, we show that $\varphi_{cc'} = (\varphi_{c'c})^{-1}$. Thirdly, we show that the φ_{cc} functions are the identity functions. Finally, we prove the co-cycle condition.

1. The edge-to-face transition functions (and their inverses) are a combination of a translation and a rotation and hence one-to-one, onto, and C^∞-differentiable. The face-to-vertex transition functions (and their inverses) are also one-to-one, onto, and C^∞-differentiable because the projective transform restricted to the overlap region is (see Appendix A.2). The edge-to-vertex transition functions (and their inverses) are one-to-one, onto, and C^k where k is the continuity of the blend function β (see Appendix D).

2. $\varphi_{EF} = (\varphi_{FE})^{-1}$ by definition. $\varphi_{VF} = (\varphi_{FV})^{-1}$ because $\zeta_{PQ} = (\zeta_{QP})^{-1}$. That $\varphi_{EV} = (\varphi_{EV})^{-1}$ is proven in Appendix D.

3. The $\{\varphi_{cc}\}_{c \in A}$ functions are the identity function by definition.

4. The co-cycle condition is trivially satisfied when one or two charts overlap. Four or more charts never overlap. When three charts overlap there is exactly one of each type of chart: vertex, edge, and face. Satisfying the co-cycle condition in this case is the same as showing that

$$\varphi_{EV} = \varphi_{FV} \circ \varphi_{EF};$$

but we defined φ_{EV} as the composition of the other two functions on each such triple-overlap area.

Finally, we need to show that any pair of charts can be embedded into the plane. If the charts do not overlap then this condition is trivially satisfied. When two charts do overlap the pair is one of the following three types: edge-face, face-vertex, or edge-vertex.

1. To embed an edge-face pair, embed E using the identity function and F by φ_{FE} extended linearly to all of F (see Figure 5.21).

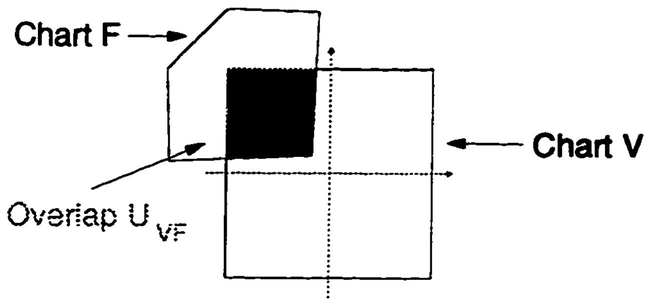


Figure 5.22: Embedding a face-vertex pair for a 5-sided face chart.

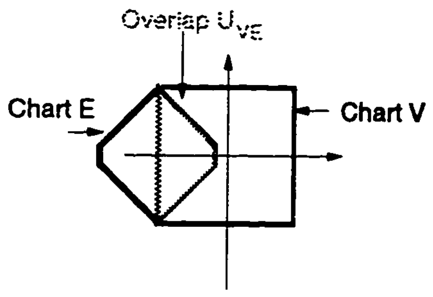


Figure 5.23: Embedding an edge-vertex pair.

2. To embed a face-vertex pair, embed V using the identity function and F by φ_{FV} extended (as a projective map) to all of F (see Figure 5.22). The discontinuity of this map does not cross the chart F, as shown in Appendix A.4.

3. To embed an edge-vertex pair, embed V using the identity function, U_{EV} by $\varphi_{EV}(U_{EV})$, and $E - U_{EV}$ by rotating and translating the region to abut the appropriate edge of V (see Figure 5.23). □

Chapter 6

Immersing the manifold

In the previous section we discussed building a structure (the manifold) whose topology is the same as a given polygonal sketch. This manifold has no associated 3D geometry; to produce a surface we need to define an immersion function on the manifold. The choice of an immersion function greatly influences the quality and usability of the resulting surface. There is a wide variety of possible immersion functions; the one described here was chosen because it satisfies these criteria:

- The immersion has a set of "controls" that can be used to alter its shape (i.e., control points):
  - The controls for the immersion are local – changes in one control alter only a small, local area of the surface.
  - The affordances between the surface and its controls are straightforward – changing a control changes the surface in an intuitive way.
- The surface is of any arbitrary, user-defined continuity.
- The surface is aesthetically pleasing (admittedly a somewhat subjective goal).
- The surface is quick to construct (we can build the surface interactively).

The immersion described here is based heavily on B-splines because they meet the above criteria and have proven to be a powerful modeling technique. Before describing the immersion itself we discuss two related topics: defining functions on the manifold (Section 6.1) and B-spline knot vectors (Section 6.2). This establishes the vocabulary necessary for defining basis functions on the manifold. We first define basis functions on a curve manifold and then extend the discussion to surfaces (Section 6.3).

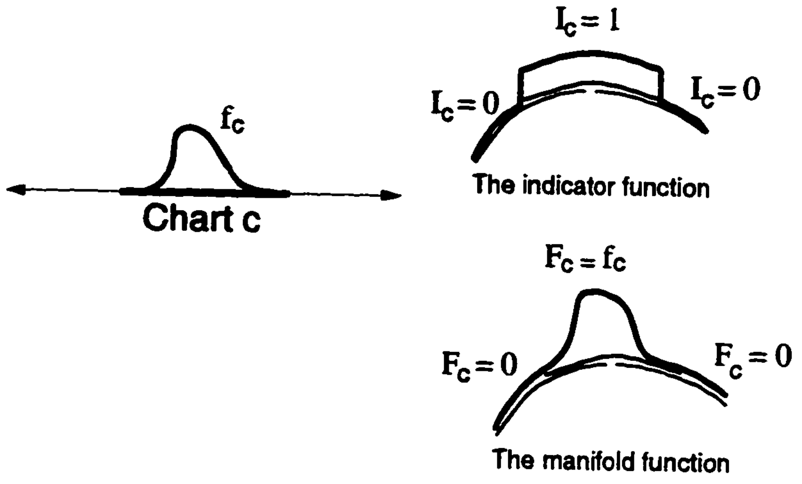


Figure 6.1: Extending a function $f_c : c \to \Re$ defined on a chart to a function $F_c : M^K \to \Re$ defined on the manifold using the indicator function $\mathcal{I}_c$.

6.1 Defining functions on the manifold

This section is largely technical. It provides the notation and conditions necessary for building a C^k function on the manifold from a C^k function in a chart, a construction demonstrated pictorially in Section 3.3. In Section 6.1.1 we describe a *partition of unity* and a method for building one.

A function f_c defined on a chart c can be extended to a function F_c on the manifold by letting F_c take on the values of f_c for points in the image of c and 0 for the remaining points of the manifold (see the 2D example in Figure 6.1). This corresponds to separating the manifold into two parts, the part that contains the image of the chart c and the part that does not. This notion is captured with the following function $\mathcal{I}_c : M^K \to \{0, 1\}$ (called an *indicator function*):

$$\mathcal{I}_c(p) = \begin{cases} 1 & \text{if } p \in \alpha_c(c) \\ 0 & \text{otherwise} \end{cases} \tag{6.1}$$

We define zero times an undefined function as zero (i.e., when α_c^{-1} is undefined). We can now define F_c as follows:

$$F_c(p) = \mathcal{I}_c(p) f_c(\alpha_c^{-1}(p))$$

This defines F_c, but what about its continuity? This definition of continuity on the manifold is from [MP77], page 209.

Definition 5 *Let M be a C^k manifold. Let $p \in M$ and let $f : M \to \Re$. f is* differentiable of class C^l *($l \leq k$) at p if there is a chart c such that $p \in \alpha_c(c)$ and*

$$f \circ \alpha_c^{-1} : c \to \Re$$

is of class C^l at $\alpha_c^{-1}(p) \in \Re^n$. The continuity of f is at best k because this must also hold when composed with the transition function of any overlapping chart c' which also contains p:

$$\varphi_{cc'} \circ (f \circ \alpha_c^{-1}) : c' \to \Re$$

which is at best C^k.

Clearly the continuity of F_c is at best the continuity of the manifold; in the following discussion we assume the continuity of M^K is greater than or equal to the continuity of f_c. The continuity of F_c on the region $\alpha_c(c)$ is the continuity of f_c, by the definition of continuity on the manifold. On the remainder of the manifold F_c is zero; the only region in question is the boundary of the region $\alpha_c(c)$. If the function f_c is zero at the boundary of the chart then the values will match up, i.e., there is no positional discontinuity. Similarly, to ensure that F_c is C^k-continuous, the function f_c must be at least C^k-continuous and its first k derivatives must be zero at the boundary of c.

6.1.1 Partition of unity

A partition of unity on a set D is a set of n functions $\{F_i : D \to \Re\}_{1 \le i \le n}$ that sum to one everywhere, i.e.,

$$\forall d \in D, \quad \sum_{i=1}^{n} F_i(d) = 1$$

Typically, each individual function $F_i(d)$ satisfies

$$\forall d \in D, \quad 0 \le F_i(d) \le 1$$

The basis functions used to define a B-spline are an example of a partition of unity; on the domain of the spline the sum of the basis functions is one.

We can build a partition of unity from a set of proto-functions $\{\hat{F}_i : D \to \Re\}_{1 \le i \le n}$ provided the following is true:

$$\forall d \in D, \quad \sum_{i=1}^{n} \hat{F}_i(d) > 0$$

We build the partition of unity by dividing each function by the sum of all of the functions:

$$F_i(d) = \frac{\hat{F}_i(d)}{\sum_{j=1}^{n} \hat{F}_j(d)} \tag{6.2}$$

6.2 Knot vectors and basis functions for manifolds

Our goal is to create functions on the manifold as similar to B-spline basis functions as possible. B-splines derive many of their characteristics from the properties of their basis functions:

- The basis functions are non-negative and sum to one. This ensures that the resulting curve lies within the convex hull of the control points.
- The support of each basis function is local, so that the corresponding control point influences only a part of the curve.
- If the basis functions are C^k then the resulting curve is also C^k.
- The left half of the basis function is monotonically increasing while the right half is monotonically decreasing. The influence of the corresponding control point has a peak that falls off nicely, contributing to the intuitive "feel" of moving a control point.

By defining basis functions on the manifold that have these properties, the resulting immersion will behave in a manner similar to a B-spline.

The major issues to address are how many basis functions to build, where their support is nonzero, and what they look like. For B-splines this information is contained in a *knot vector*. In this section we discuss the role of the knot vector in defining B-spline basis functions and then show how a similar idea can be used to define basis functions on a curve manifold of the type found in Chapter 3. We build the basis functions in the charts; they can be extended to functions on the manifold using the techniques of Section 6.1.

In Section 6.3 we extend the discussion to the plane and hence to surface manifolds. For the remaining discussion, let k be the desired continuity of the curve or surface.

6.2.1 Knot vectors for curves

We define a knot vector on the real line and list some of its properties. A knot vector is an increasing[1] sequence of n_r numbers $\{r_0, \ldots, r_{n_r-1}\}$ that describes where the "breaks" are between the polynomials making up a spline (see Figure 6.2). A knot vector with n_r knots defines $n_r - (k+2)$ basis functions of degree k in the following manner $(0 < i \leq n_r - (k+2)$, $0 \leq d \leq k)$:

$$b_i^{-1}(t) = \begin{cases} 1 & \text{if } r_i \leq t < r_{i+1} \\ 0 & \text{otherwise} \end{cases}$$

$$b_i^d(t) = \frac{(t - r_i)}{r_{i+k+1} - r_i} b_i^{d-1}(t) + \frac{(r_{i+k+1} - t)}{r_{i+k+2} - r_{i+1}} b_{i+1}^{d-1}(t) \tag{6.3}$$

[1] We do not consider duplicated knots here.

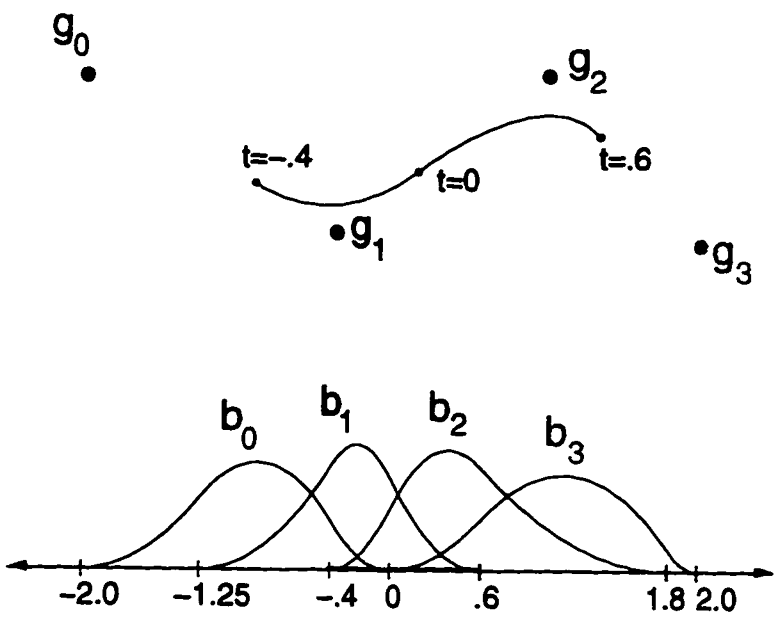


Figure 6.2: A knot vector $(-2.0, -1.25, -0.4, 0, 0.6, 1.8, 2.0)$, the four basis functions of degree one it defines, and the resulting curve.

The knot vector has the following effects on the basis functions for the spline:

- There are $n_r - (k+2)$ basis functions. The support for the ith basis function lies between the knot r_i and the knot r_{i+k+2}. This means that every subset $[r_i, r_{i+k+2}]$ of the real line contains the entire support of one basis function.
- On any given interval $[r_i, r_{i+2}]$ there are at most $k+2$ nonzero basis functions and everywhere but the end intervals there are exactly $k+2$ nonzero basis functions. (As an aside, the visual smoothness of higher order basis functions ($k > 2$) comes largely from the fact that more control points are involved in the blend at every point, not from the higher order continuity of the basis functions.)

6.2.2 Knot vectors for curve manifolds

The knot vector serves two purposes: it determines the supports of the basis functions and it ensures that enough basis functions overlap at every point. To define basis functions on a manifold we begin by defining something similar to the knot vector on the manifold, then use it to define the support of the basis functions. Because the basis functions are defined initially in the charts, we need a way to determine what part of the knot vector lies in a given chart. Once we have this subset of knots, we can use it to define a traditional knot vector in the chart. This traditional knot vector is then used in Equation 6.3 to define basis functions on the chart (see Figure 6.4).

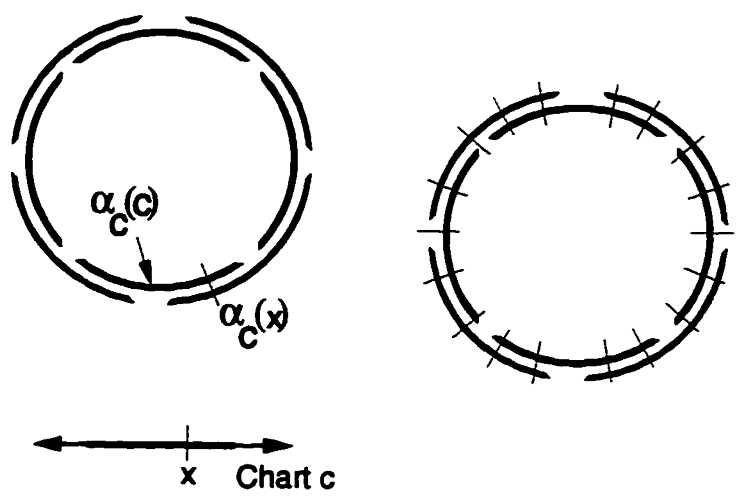


Figure 6.3: Left: Defining a knot. Right: A knot vector κ on the manifold with 16 knots.

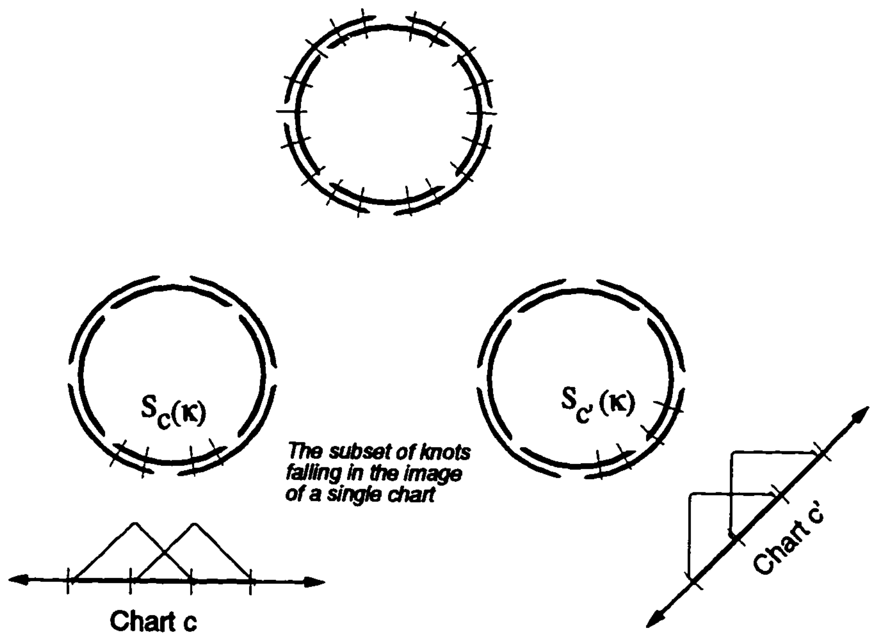


Figure 6.4: Top: A knot vector $\mathcal{K}$ on the manifold. Bottom: The knots falling in the image of the chart c (left) and the chart c' (right). These knots are drawn in the corresponding chart c or c', along with the basis functions of degree $k = 1$ resulting from them.

We now carry out this construction for the curve manifold of Chapter 3. The knot vector on the manifold is defined as an unordered set of points on the manifold. If κ is an element of this set, then it is in the equivalence class of at least one point in some chart:

$$\kappa = \alpha_c(x) \text{ for some point } x \in c$$

The knot vector $\mathcal{K}$ on the manifold is then an unordered set of $n_{\mathcal{K}}$ knots (see Figure 6.3):

$$\mathcal{K} = \{\kappa_i\}_{0 \leq i < n_{\mathcal{K}}}$$

Given a chart c, we can impose an order on a subset of $\mathcal{K}$. Let $S_c(\mathcal{K})$ be the set of knots that fall within the chart c, i.e.,

$$S_c(\mathcal{K}) = \{\kappa \in \mathcal{K} : \mathcal{I}_c(\kappa) = 1\}$$

($\mathcal{I}_c$ is the indicator function for the chart c.) We impose an order on $S_c(\mathcal{K})$ as follows:

$$\forall \kappa_i, \kappa_j \in S_c(\mathcal{K}), \quad \kappa_i < \kappa_j \text{ iff } \alpha_c^{-1}(\kappa_i) < \alpha_c^{-1}(\kappa_j)$$

Figure 6.4 shows a set of knots on the example manifold of Section 3.2 and the resulting ordered subset in two charts, c and c'.

Aside 5 Knot ordering: *A subset of knots falling in the image of the overlap region $U_{cc'}$ has one ordering imposed upon it by the chart c and another by the chart c'. For the curve manifold, either these two orderings are the same or one is the reverse of the other.*

When we extend this technique to the surface manifold and hence to knots in a plane, defining an ordering on every chart is not so simple. We define an ordering in the vertex charts (where there is a natural grid) and use it to impose an ordering on the knots in the edge and face charts.

We now consider defining basis functions on the manifold using $\mathcal{K}$. Within a given chart c we have an ordered list of knots $S_c(\mathcal{K})$ that can be treated as a traditional knot vector on the real line, i.e., we can use it to define basis functions with Equation 6.3. It takes $k+2$ knots to define a single basis function with that equation, so we define a basis function for every ordered list of $k+2$ knots in the subset $S_c(\mathcal{K})$. Note that the same list of $k+2$ knots may appear in a different chart as well; we chose one of the two charts to define the basis function associated with that list of knots.

Aside 6 Choosing a chart: *For this particular manifold the choice of the chart makes no difference – the basis functions are the same regardless. This is because the transition functions are translations and the basis functions are invariant under translation. When defining basis functions for the surface this holds true only for areas corresponding to four-sided faces.*

We can now construct basis functions on the manifold using the manifold knot vector $\mathcal{K}$. The remaining question is what $\mathcal{K}$ should consist of: we want to satisfy the two properties above, i.e., that each interval between two adjacent knots be contained in the supports of $k+2$ basis functions and that the support of each basis function contain $k+2$ intervals. The latter is satisfied by the construction of the basis functions; it remains to ensure that enough basis function supports overlap at every point.

For the region between two adjacent knots to be contained in the supports of $k+2$ basis functions, there must be $k+1$ knots to the left and $k+1$ knots to the right of the two adjacent knots (see Figure 6.5). This can be achieved in one of two ways: one chart can contain all of the needed knots, or the knots can be partially split between two charts. In the latter case, one chart has $k+1$ knots to the left of the interval and the other has $k+1$ knots to the right of the interval. Additionally, the left chart must have k_a knots to the right of the interval and the right chart must have k_b knots to the left of the interval, where $k_a + k_b = k$.

To achieve this we place the knots of the manifold knot vector as follows. Let $\delta_k = \frac{1}{(2(k+2))}$:

$$\mathcal{K} = \{\kappa : \kappa = \alpha_c(-.5 + \frac{\delta_k}{2} + j\delta_k), \quad \forall c \in \mathcal{A}, \quad 0 \leq j < 2(k+2)\}$$

As shown in Figure 6.6, this produces a sufficient number of knots in each chart to define a basis function for each knot.

In summary, we define a "knot vector" on the manifold by dividing up the charts into $2(k+2)$ intervals. We then define a basis function for every interval of $k+1$ knots, thus ensuring that each knot interval is contained in the support of $k+2$ basis functions. Example knot vectors and their corresponding basis functions are shown in Figure 6.7.

6.3 Defining basis functions on the manifold

In this section we demonstrate how to extend the ideas of the previous section to the plane and hence to surface manifolds. Since the charts are subsets of $\Re^2$ instead of $\Re^1$, we divide the manifold into 2D regions instead of 1D intervals. As in the curve example, we ensure that each region is contained in the supports of some number of basis functions and that the support of each basis function consists of a given number of regions.

We begin with the "rectangular areas" of the manifold, i.e., areas where all the corresponding faces in the polyhedron are four-sided. In the curve example the manifold was divided into intervals by dividing the charts into line segments; here we divide the manifold into approximately square regions. To do this we "grid" the manifold. Each grid square is defined by four knots that are connected together to form a square. We require that each grid square be contained in the supports of $(k+2) \times (k+2)$ basis functions and the support of each basis function consist of $(k+2) \times (k+2)$ grid squares, i.e., a square $(k+2)$ knots on a side.

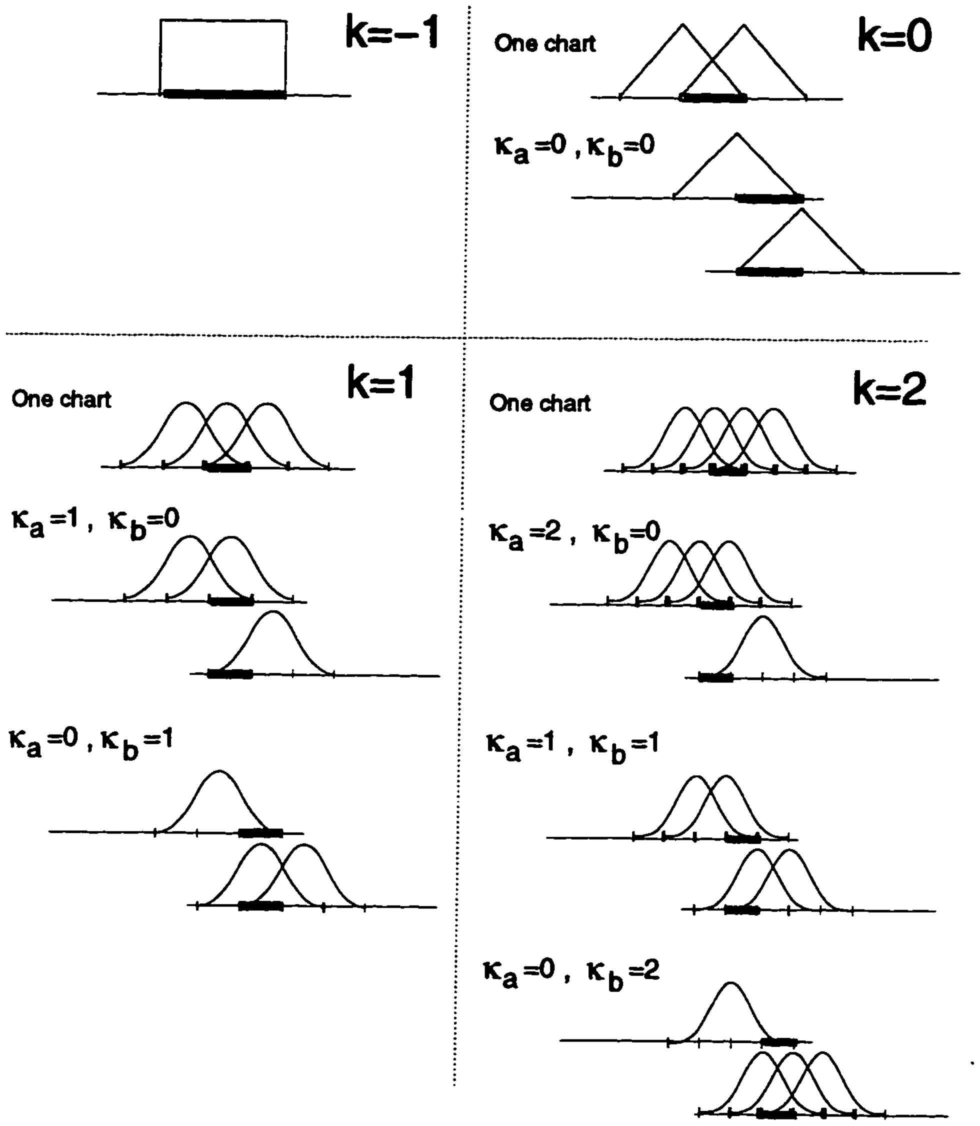


Figure 6.5: The number of knots needed to ensure that an interval between two adjacent knots is contained in the support of $k+2$ basis functions. The knots can be distributed amongst the charts as shown.

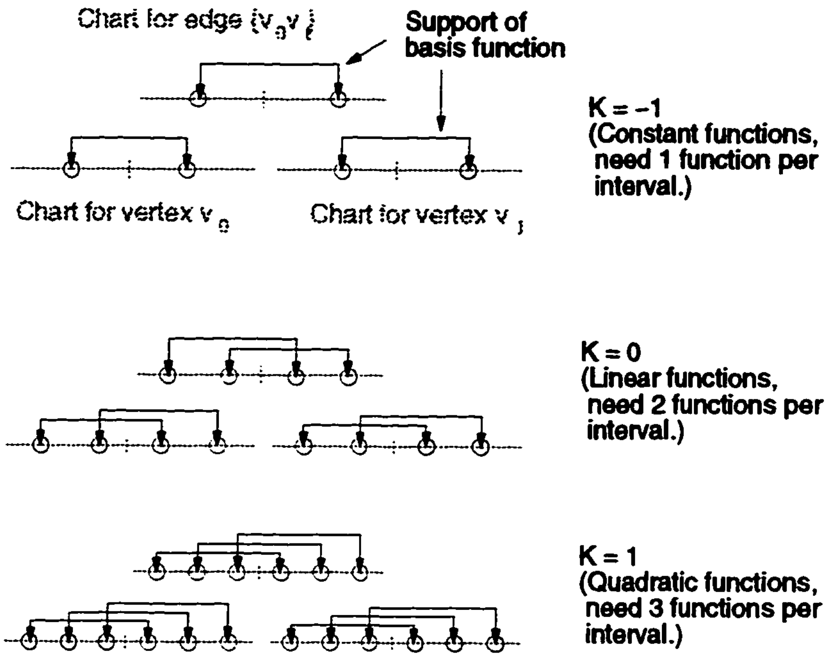


Chart for edge {v_0v_1}
Support of basis function
Chart for vertex v_0
Chart for vertex v_1
K = -1
(Constant functions, need 1 function per interval.)
K = 0
(Linear functions, need 2 functions per interval.)
K = 1
(Quadratic functions, need 3 functions per interval.)

Figure 6.6: Examples of assigning $2(k+2)$ knots to each chart for three overlapping charts of the manifold.

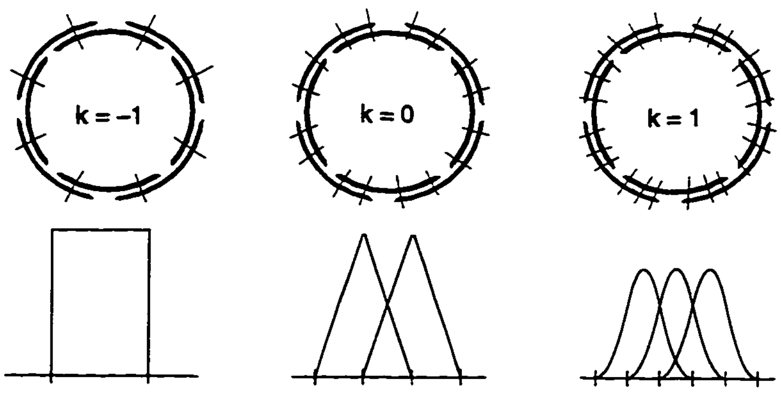


Figure 6.7: The manifold knot vector and corresponding basis functions on a chart for various values of k.

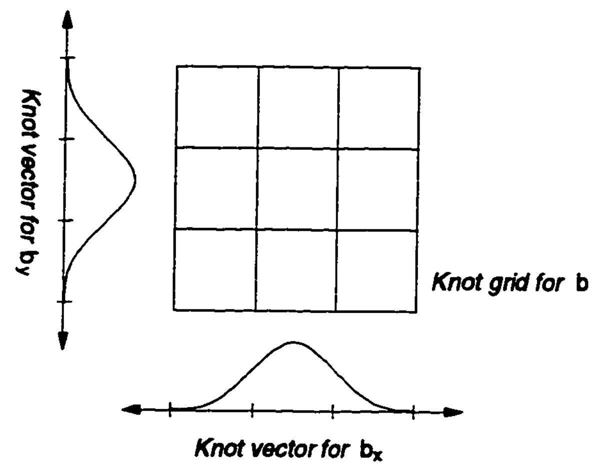


Figure 6.8: A grid of knots in the plane formed by the cartesian product of two curve knot vectors ($k = 1$).

6.3.1 Knots in the plane

One method for defining a B-spline basis function b in the plane is to take a *tensor product* of two B-spline curves, b_x and b_y:

$$b(s,t) = b_x(s)b_y(t)$$

where b_x and b_y are each defined using a knot vector. The function b is then defined by the *cartesian product* of these two knot vectors, instead of by a single knot vector. We call this product a knot *grid* because it forms a grid of points in the plane (see Figure 6.8). This knot grid is a $(k+2)$-by-$(k+2)$ grid of points because each of the curve knot vectors has $(k+2)$ knots in it.

6.3.2 Knot sets for surface manifolds

We require that the knot set for the manifold form a grid when the knot set is drawn in a vertex chart. This grid is then used to create knot grids for B-spline basis functions. To ensure that the knots form grids, we define the manifold knot set as follows. For each vertex chart we create a grid of points. These points, mapped to the manifold, form the knot set. Note that, unlike the curve example, we define knots only in the vertex charts. Let

$$\delta_k = \frac{1}{2(2k+3)} \tag{6.4}$$

$$x_i = -.5 + \frac{\delta_k}{2} + i\delta_k$$

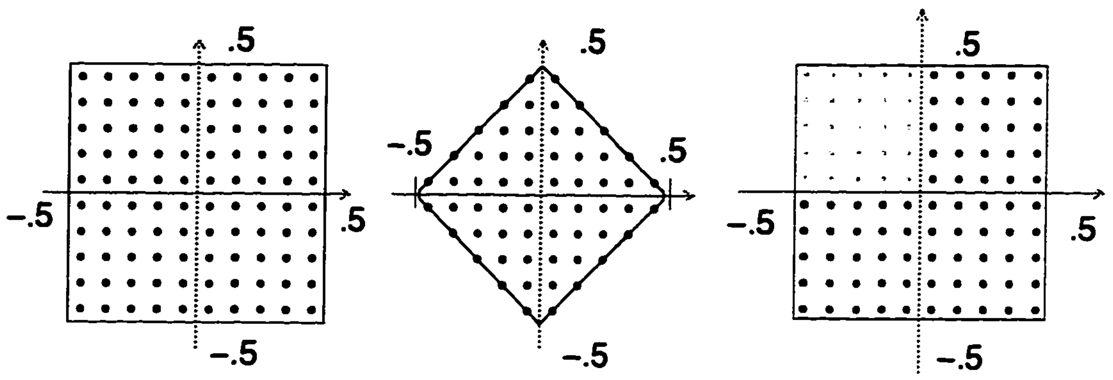


Figure 6.9: The knot set for $k = 1$ ($\delta_k = 0.1$) in a vertex chart (left), an edge chart (middle) and a four-sided face chart (right).

$$y_j = -.5 + \frac{\delta_k}{2} + j\delta_k$$

Then the manifold knot set $\mathcal{K}$ is

$$\mathcal{K} = \{\kappa : \kappa = \alpha_V(x_i, y_j),\ \ \forall V \in \mathbf{V},\ \ 0 \leq i, j < 2(2k+3)\}$$

The knot set forms a grid when drawn in a vertex chart. If we (temporarily) restrict the manifold to four-sided face charts, then each of the transition functions is a combination of a translation and a rotation by some integer multiple of $\pi/2$. Because of this, the knot set also forms a grid when drawn in an edge or face chart (see Figure 6.9). In analogy to the curve example, we could define a chart basis function for each $(k+2)$-by-$(k+2)$ grid of knots. Unfortunately, extending this definition to the three-, five-, and six-sided face charts is impossible because the knots do not form a regular grid in these cases (see Figure 6.10). We can, however, use an equivalent method for defining basis functions that has the advantange that it can be extended to the non-four-sided cases.

Let g_k be any four knots forming a quadrilateral in some chart. Let ζ_{g_k} be the projective transform from the quadrilateral to the unit square. Now let b be the C^k tensor product B-spline whose support is the unit square and whose knot spacing is equal. We define a basis function as follows:

$$b \circ \zeta_{g_k} \tag{6.5}$$

In essence, this equation "stretches" the basis function b so that its support becomes the quadrilateral defined by the four knots g_k. Note that the equation depends only on the locations

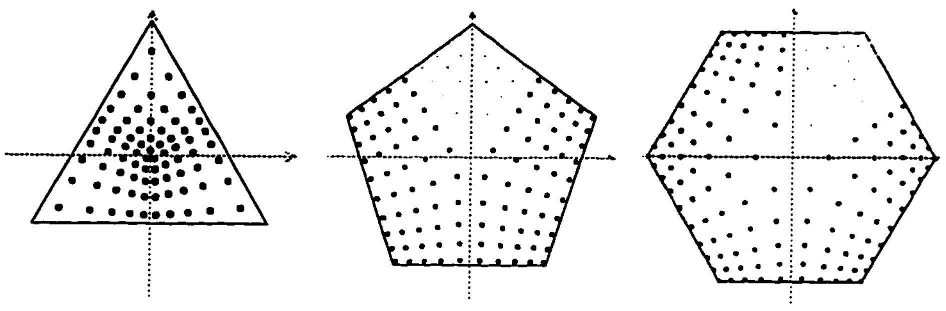


Figure 6.10: The knot vector in three-, five-, and six-sided face charts ($k = 1$). The knots form an n-sided pattern in the center of the chart.

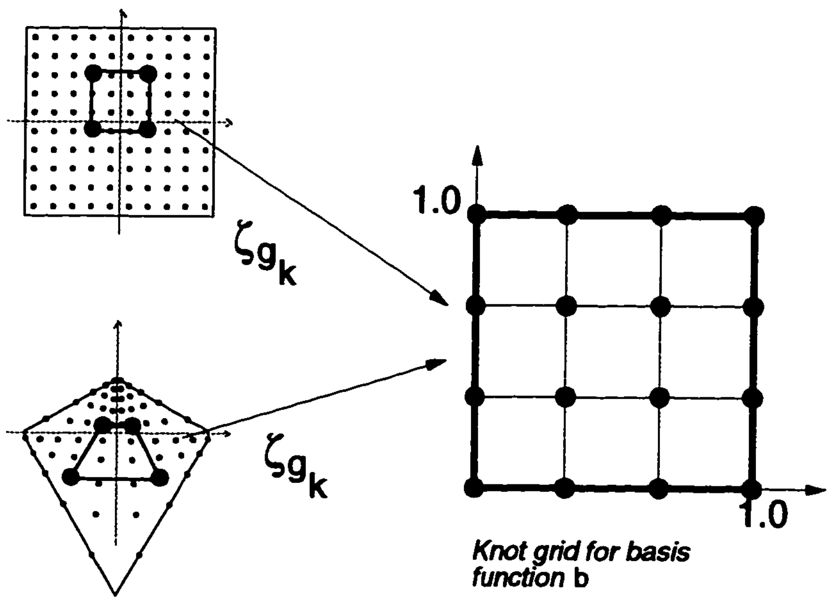


Figure 6.11: Defining an equivalent basis function using only the four corner knots instead of the entire grid. The projective map ζ_{g_k} takes the four corner knots to the four corners of the unit square. The basis function is defined to be the composition $b \circ \zeta_{g_k}$. Top: A vertex chart. Bottom: A three-sided face chart.

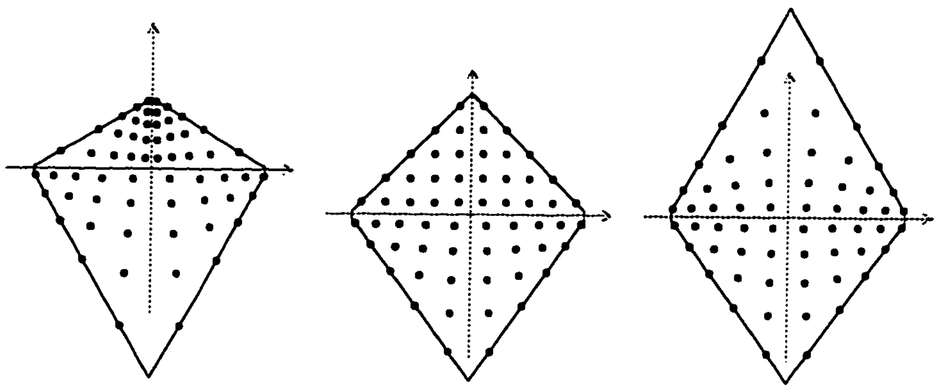


Figure 6.12: The knot vector in three different edge charts ($k = 1$). The knots form a distorted grid.

of the four knots g_k, not a grid of knots. To define a basis function we simply chose the four knots which form the "corners" of the support of the function.

In the four-sided regions, the g_k are chosen to be the four corner knots of a $(k+2)$-by-$(k+2)$ grid of knots. The quadrilateral in this case is a square; therefore the projective transform ζ_{g_k} is a rigid motion and takes the knots in the square g_k to the knots of b (see Figure 6.11). The basis function $b \circ \zeta_{g_k}$ is therefore identical to one defined using the $(k+2)$-by-$(k+2)$ grid of knots.

In the non-four-sided case we chose the g_k as follows: the knot set, when drawn in an edge chart, forms a distorted grid (see Figure 6.12). Choose any four knots that form the corners of a (distorted) $(k+2)$-by-$(k+2)$ grid. The knot set, when drawn in a face chart, forms a grid-like pattern near the edges of the face chart but an n-sided pattern in the center (see Figure 6.10). In this case, choose any four knots which meet the following criterion:

- The knots are separated on three sides by $k+1$ knots in a row.
- The knots form a convex quadrilateral (this is only a concern for three-sided charts).

Some examples of these are shown in Figure 6.13.

It is possible for four knots to form a basis function in two different charts c and c' if all four knots lie in the image of $U_{cc'}$. In this case we select a single definition: we select a vertex chart basis function over that for an edge or face chart (and similarly an edge chart one over a face chart one).

Note that the basis functions no longer sum to one in the middle of the non-four-sided face charts. We fix this by dividing by the sum of the basis functions (see Section 6.1.1). Some care

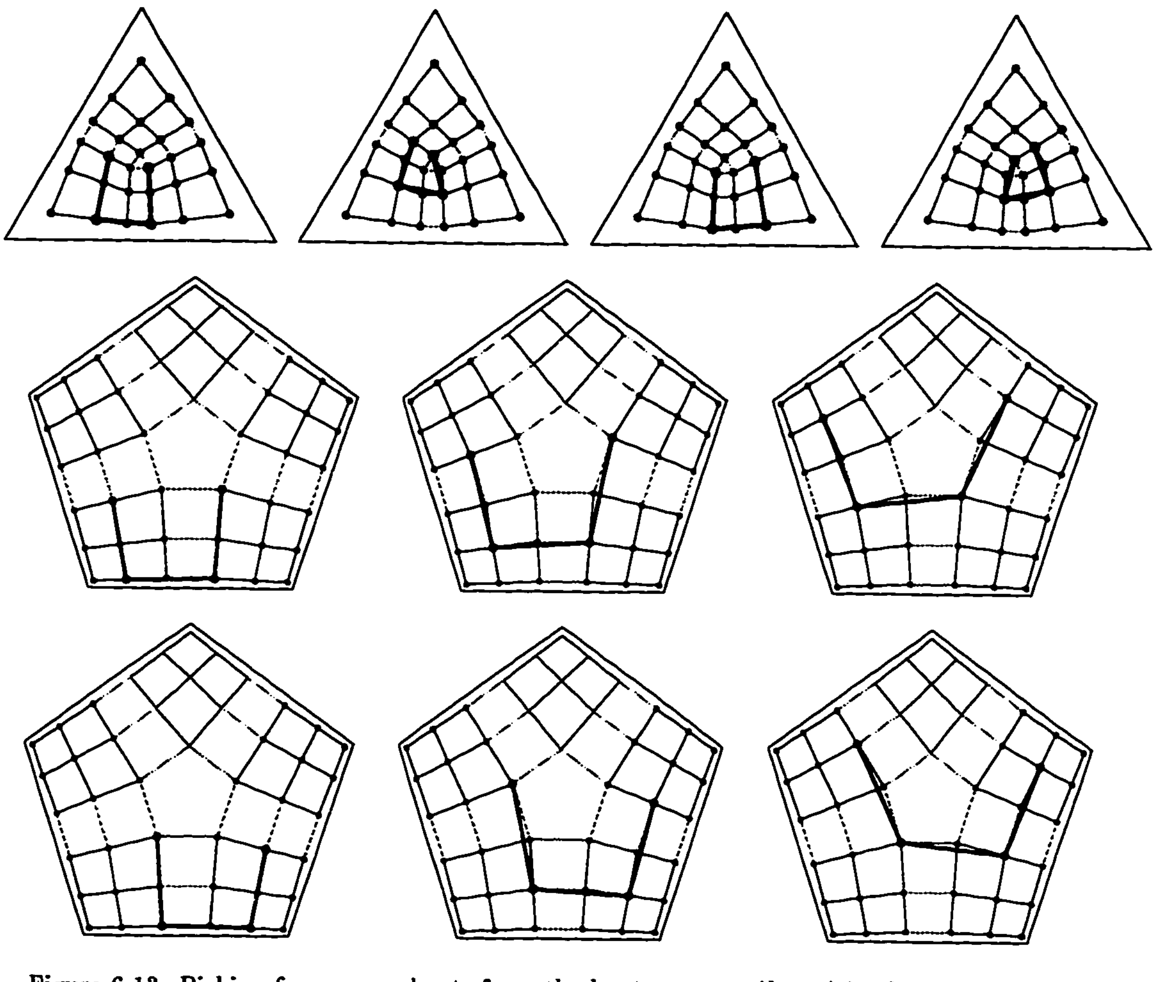


Figure 6.13: Picking four corner knots from the knot patterns ($k = 0$) in the face charts. Note that the knots contained within the four corner knots may not form a grid.

is needed in the $k = -1$ case to ensure that every point in the manifold is contained in the support of some function.

Chapter 7

Manifolds with boundary

To extend this discussion to surfaces with boundary, we need to define what a manifold with boundary is (a modification of Definition 1), how to define one from charts and transition functions (a modification of Definition 2), and how to build one from a polyhedron. The definition of a manifold with boundary is very similar to the original definition except the mappings to $\Re^n$ may be mappings to "open sets with boundary" as well as open sets [Spi70].

Similarly, the charts and overlap regions of Definition 2 are expanded to include open sets with boundary; the remainder of the definition is unchanged. Creating a manifold with boundary from a polyhedron with boundary involves defining new chart types for the boundary elements in the polyhedron; the overlap regions and the transition functions are largely unaffected.

The immersion of a manifold with boundary is defined exactly as before. The algorithm for constructing the basis functions, however, must be altered slightly to construct the basis functions along the boundary.

7.1 The polyhedron with boundary

To sketch a surface with boundary we use a polyhedron with boundary (see Appendix B). Each vertex on the boundary has exactly one, two, or three contiguous faces adjacent to it (and hence two, three, or four edges). Each boundary edge has exactly one face adjacent to it. For each of these different boundary elements (three vertex and one edge) we define a new chart type. The following sections discuss what the new chart types look like, along with their corresponding overlap regions and transition functions.

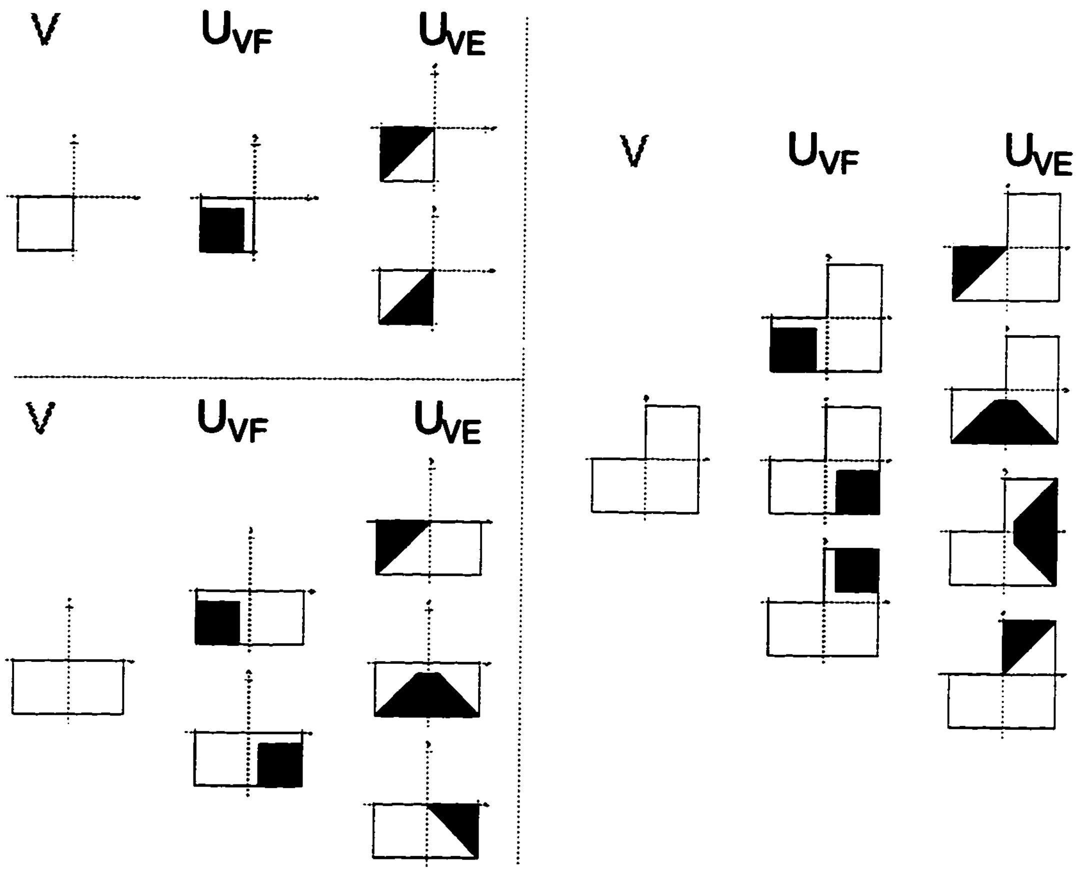


Figure 7.1: The shape and overlap regions of the three types of boundary vertex charts. The adjacent faces have four sides.

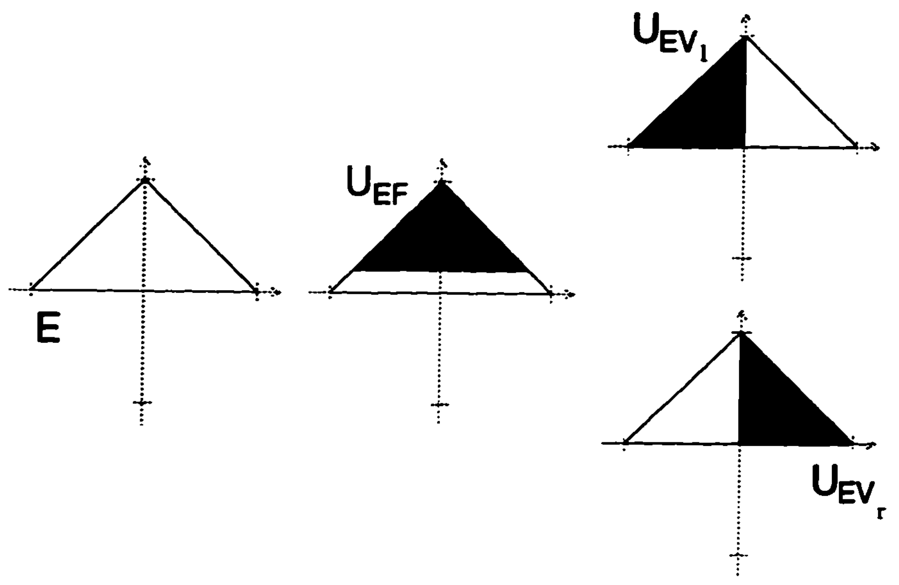


Figure 7.2: The shape and overlap regions of a boundary edge chart (the adjacent face has 4 sides).

7.2 The charts and the overlaps

The new vertex charts consist of some number of the quadrants of the unit square and the adjacent parts of the x- and y-axes. The number of quadrants depends on the number of faces adjacent to the vertex; if there are n adjacent faces then there are n quadrants. The quadrants must be adjacent to each other. The overlap regions (U_{VF} and U_{EV}) are defined as before (see Figure 7.1).

Each boundary edge chart is a triangle without boundary, joined to the adjacent part of the x-axis. The exact shape of the triangle is determined by the number of sides, n_f, of the face adjacent to it. The three corners of the triangle are:

$$(0, \frac{\cot(\pi/n_f))}{2}$$

$$(-0.5, 0) \qquad\qquad (0.5, 0)$$

The overlap regions U_{EF} and U_{EV} are defined as before (see Figure 7.2).

There are no new face chart types.

7.3 The transition functions

Both the face-to-vertex and face-to-edge functions are unchanged. A boundary edge has only one face overlapping it; therefore the edge-to-vertex function in this case is just the composition of φ_{EF} and φ_{FV} extended to all of U_{EV} (see Figure 7.3).

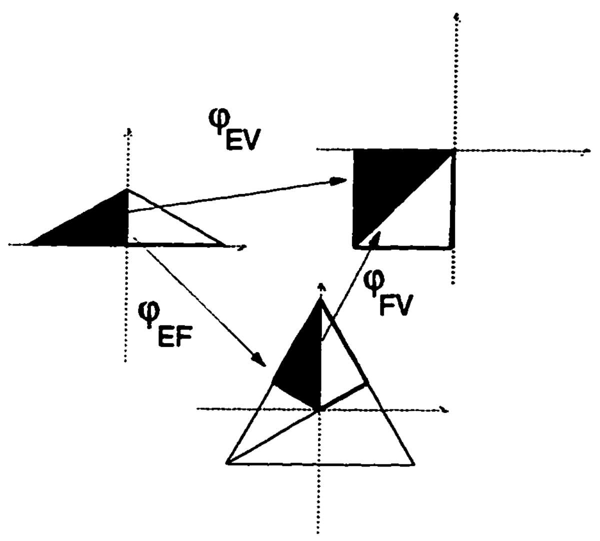


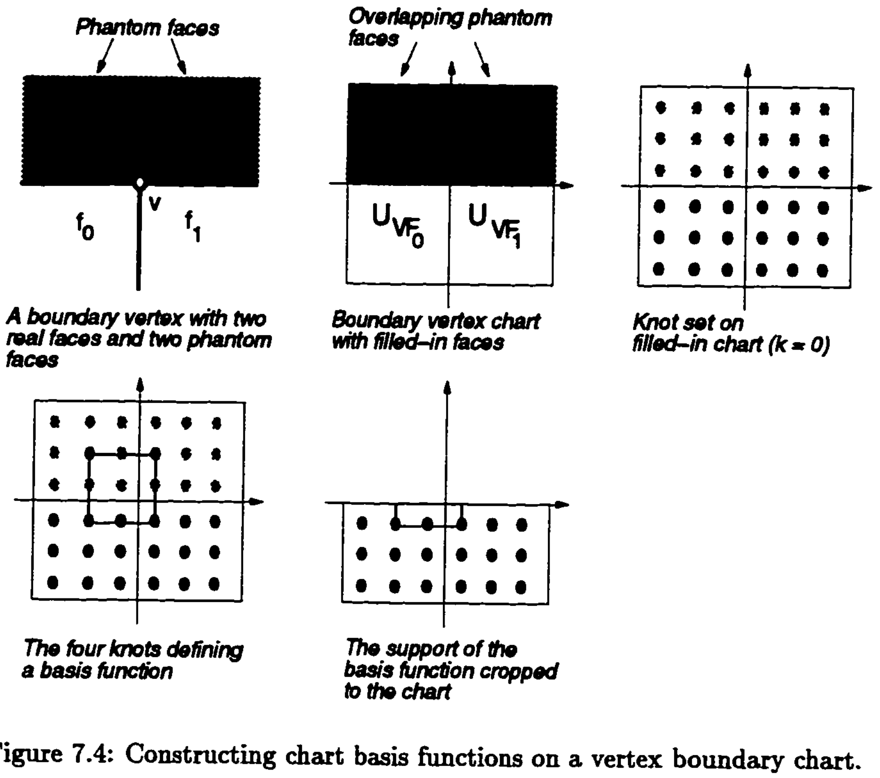
Figure 7.3: The transition function φ_{EV} for the boundary case is build from a single composition.

7.4 The immersion

The charts are of two types, interior charts or boundary charts. The basis functions on the interior charts are defined exactly as described before. To define the basis functions on the boundary charts, the "missing" overlapping faces of the boundary charts are filled in with phantom four-sided face charts. The basis construction algorithm is then run normally to produce a set of functions on the "filled-in" boundary charts. The supports of these functions are then clipped to the boundary chart to produce the boundary basis functions.

So, for example, a boundary vertex chart with two overlapping face charts is expanded to a unit square, where the two top quadrants overlap with the two "phantom" face charts. We build the knot set on the unit square as before and construct a function for each $(k+2) \times (k+2)$ grid of knots. The support of some of these functions lies partly (or entirely) within the filled-in area of the chart; we clip the support of these functions to the boundary vertex chart, i.e., the area including and below the x-axis (see Figure 7.4).

Each boundary edge chart has two boundary vertex charts that overlap it; we map the knot set (defined on the filled-in area of the vertex charts) from the vertex charts to the edge charts. This fills in the knot set for the edge chart, where knots in the filled-in part of the vertex charts are mapped to the filled-in part of the edge chart (see Figure 7.5). As before, we construct a function for each $(k+2) \times (k+2)$ grid of knots and clip the result to the boundary edge chart.



Phantom faces
v
f0
f1
A boundary vertex with two real faces and two phantom faces
Overlapping phantom faces
U VF0
U VF1
Boundary vertex chart with filled-in faces
Knot set on filled-in chart (k = 0)
The four knots defining a basis function
The support of the basis function cropped to the chart

Figure 7.4: Constructing chart basis functions on a vertex boundary chart.

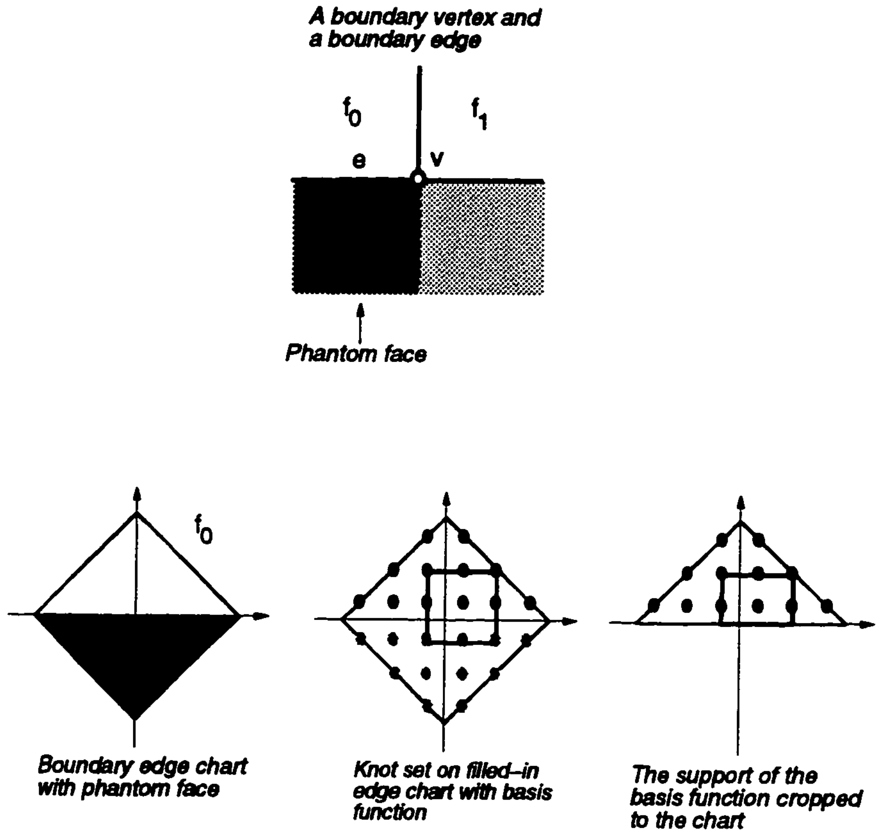

A boundary vertex and a boundary edge
f_0
f_1
e
v
Phantom face
f_0
Boundary edge chart with phantom face
Knot set on filled-in edge chart with basis function
The support of the basis function cropped to the chart

Figure 7.5: Constructing basis functions on an edge boundary chart.

Chapter 8

Additional issues

The following sections discuss some issues that have not been fully dealt with.

8.1 The gap h

This section discusses the choice of the value for the gap h. The gap must be small enough that all of the knots in the knot set fall within the overlap region U_{VF}. Figure 8.1 shows the knot set in a vertex chart and corresponding overlap regions obtained with different values of h.

The knot located at $(\delta_k/2, \delta_k/2)$ must lie within the overlap region U_{VF_2} (where δ_k is given in Equation 6.4). We solve for the value of h for which the corner of U_{VF_2} is exactly $(\delta_k/2, \delta_k/2)$.

$$\begin{aligned} (\delta_k/2, \delta_k/2) &= \varphi_{FV}(0.5 - h, -h\cot(\pi/6)) && (8.1) \\ \rightarrow h &= \frac{0.0416667}{1.33333 + 1.\,k} && (8.2) \end{aligned}$$

We set h to be slightly smaller than this to ensure that the knots all lie within the overlap region.

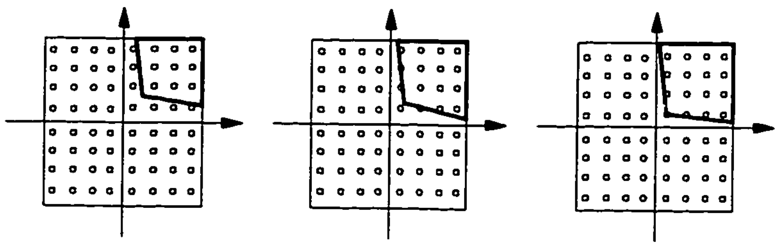


Figure 8.1: The knots in the vertex chart and the overlap region U_{VF} for decreasing values of h (F has three six).

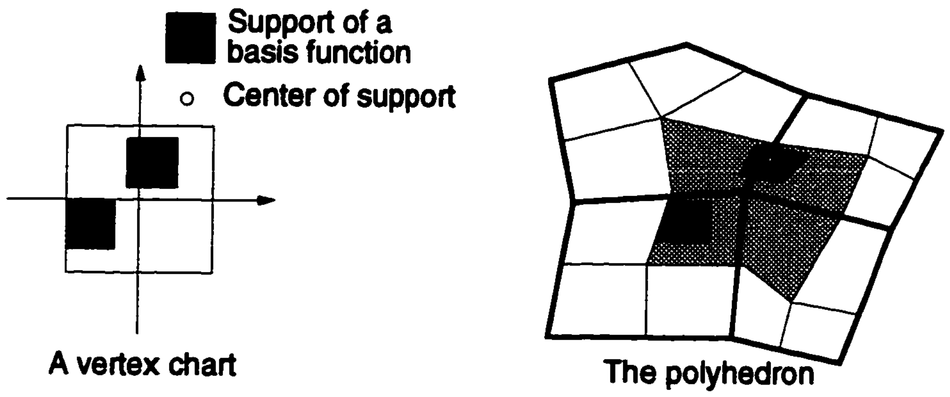


Figure 8.2: Left: A vertex chart on the manifold with the support and centers of two basis functions marked. Right: The sketch polyhedron subdivided once with the corresponding locations of the corresponding control points marked.

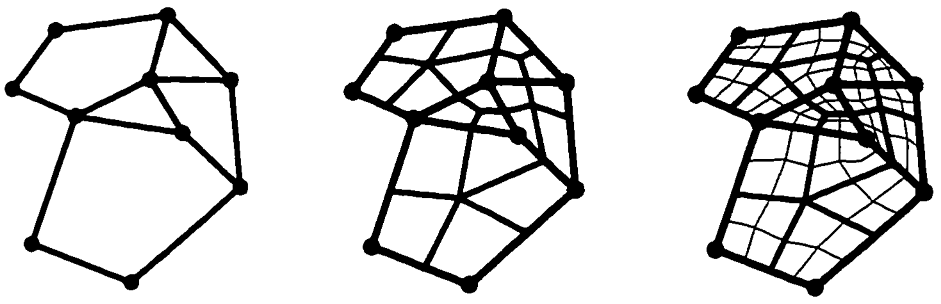


Figure 8.3: Left: Original polyhedron P. Middle: After one level of subdivision (P'). Right: Subdividing again (P'').

8.2 Initial locations for the control points

Although the choice of control points is completely unconstrained, we chose an initial set of values that reflects the user's intentions, as specified by the geometry of the polyhedral sketch. Since the Catmull-Clark [CC78] subdivision process produces a smooth surface from an arbitrary polyhedron, we use the subdivision surface to compute the placement of the control points. We could place the control points on the polyhedron itself but that would lead to a blocky surface.

Each control point G is associated with a basis function B whose center of support is at some manifold point $p \in M^K$. We want G to be the point on the subdivision surface that "corresponds" to p. To do this we need to associate the points of the subdivision surface $\mathcal{L}$ with the points of the manifold. We describe this association with a mapping $\mathcal{H} : M^K \to \mathcal{L}$. To set the control point G we assign it to its corresponding point on the subdivision surface, $\mathcal{H}(p)$.

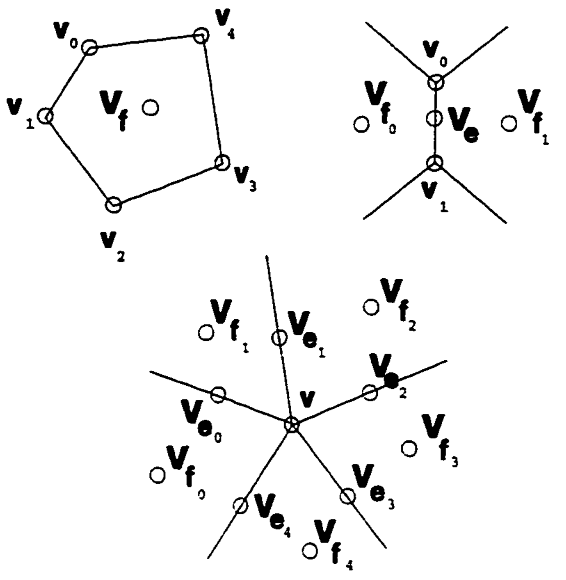


Figure 8.4: Upper left: A face f of P with the new point $v_f \in P'$. Upper right: An edge of P and the two adjacent centroid locations. Bottom: A vertex v of P and the adjacent edge and face points.

Aside 7 Subdivision surfaces: *See [CC78] for a complete description of subdivision surfaces.*

Subdivision takes a polyhedron P and produces a new polyhedron P'. For each vertex, edge, and face in P there is a vertex in P'. These vertices are connected together by four-sided faces (see Figure 8.3).

A face f in P produces a vertex v_f in P' whose location is the centroid of the face f in P (see upper left of Figure 8.4). If v_i are the locations of the vertices of the face f then

$$v_f = \frac{1}{n}\sum_{1}^{n} v_i$$

An edge $\{v_0, v_1\}$ in P produces a vertex v_e in P' by averaging the centroids of the two faces adjacent to the edge and the two vertices v_0 and v_1. If v_{f_0} and v_{f_1} are the previously calculated centroids of the adjacent faces (see upper right of Figure 8.4) then

$$v_e = \frac{1}{4}(v_0 + v_1) + \frac{1}{4}(v_{f_0} + v_{f_1})$$

A vertex v in P produces a vertex v_v in P' by averaging the points in P' corresponding to the n adjacent faces v_{f_i} and the n adjacent edges v_{e_i} of v. Let $a, b \in \Re$, $0 < a, b < 1$

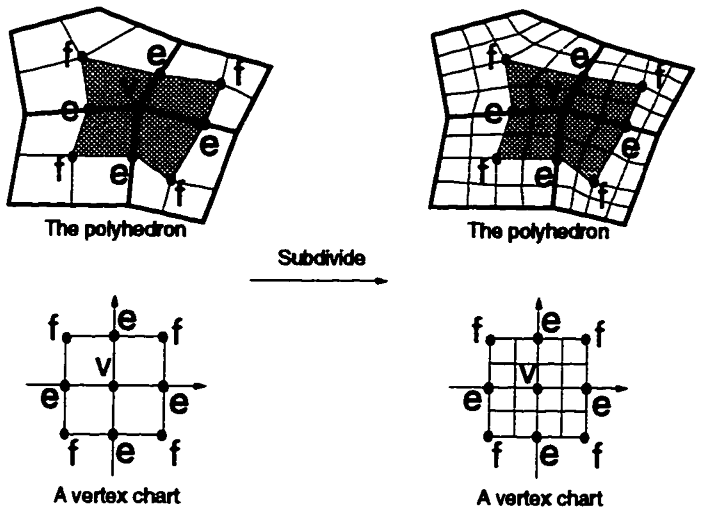


Figure 8.5: An abstraction of the subdivision process on a vertex chart. The surface has been subdivided twice.

$$v_v = \frac{a}{n}\sum_1^n v_{f_i} + \frac{b}{n}\sum_1^n v_{e_i} + (1-(a+b))v$$

To define $\mathcal{H}$, we first note that after one level of subdivision we have one subdivision point for each chart in M^K. If l_c is the subdivision point corresponding to the element c in the polyhedron then we set $\mathcal{H}(\alpha_c(0,0)) = l_c$. This relates the origins of the charts to the subdivision points generated by the first level of subdivision. The subdivision points, therefore, lie in the vertex charts as shown in the left of Figure 8.5.

Recall that the sketch polyhedron has vertices of valence four, so that after one level of subdivision every face in the subdivision surface is four-sided. We define $\mathcal{H}$ to take each face of $\mathcal{L}$ into a quadrant of a vertex chart. Applying another level of subdivision "grids" the vertex chart as shown in the right of Figure 8.5. Each further level of subdivision divides this grid again. We define $\mathcal{H}$ by assigning the grid points (α_V(grid point)) to their corresponding points in the subdivision surface. Eventually, this relates a set of points that are dense in M^K to the subdivision points.[1] This dense set can be extended to M^K in a natural way.

In practice, we compute several levels of subdivision and then interpolate by finding the grid square in which the point $\alpha_V^{-1}(p)$ lies and interpolating between the values of the corners of the grid square.

[1] We assign the points along the boundary of the vertex charts $\alpha_V(V)$ to their adjacent points in M^K.

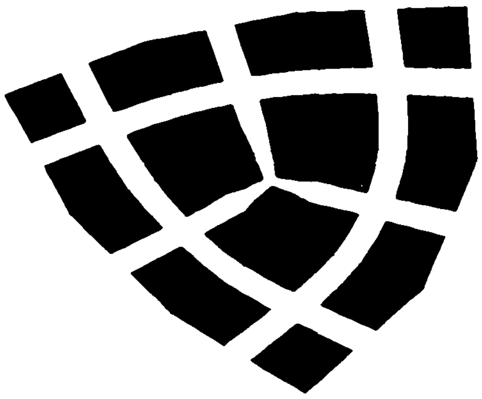


Figure 8.6: Triangulating the interior of a vertex chart ($r = 5$). The polyhedron is a three-sided face surrounded by four-sided faces.

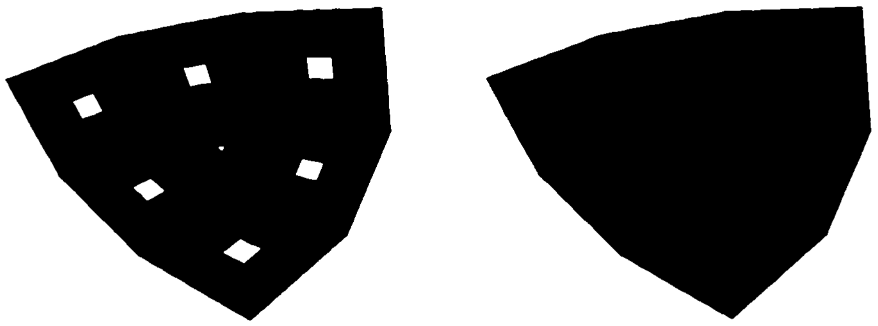


Figure 8.7: Filling in the gaps of the triangulation. Left: Adjoining edges. Right: Filling in the centers of the faces.

8.3 Triangulating the manifold

There is a tradeoff between the number of triangles in the triangulation and how closely that triangulation approximates the surface. The triangulation presented here produces approximately r^2 triangles per vertex for a given resolution r. If the control points are evenly spaced in 3-space then the resulting triangulation will also be evenly spaced in 3-space.

We triangulate the domain by first triangulating the vertex charts as shown in Figure 8.6, where r determines the number of squares. To fill in the remaining gaps, we adjoin the triangles along the boundaries to the triangles of neighboring vertex charts. The edges are filled in with a strip of triangles and the remaining corners with an appropriate n-sided triangulation (see Figure 8.7).

Chapter 9

A history of alternative approaches

This chapter describes the alternative approaches we tried and why we discarded them. The discussion is organized into three categories: the polyhedron, the manifold, and the immersion.

9.1 The polyhedron

Two restrictions are placed on the polyhedron: the vertices have valence four and the faces have six or fewer sides. This restricted polygon was chosen over a general polygon because it simplifies constructing the manifold and imposes a "local coordinate system" on the manifold in that the vertex charts can be modeled with unit squares. If all the vertices have the same valence then the number of possible combinations with $n-$sided faces is reduced, i.e., we need not deal with vertices of valence m adjacent to faces with n sides. By limiting the number of combinations we decrease the complexity of the charts and their overlaps. Alternatively, we could use the dual to the restricted polyhedron, which has vertices of valence m and faces with four sides. This would reverse the role of the vertex and face charts: the face charts would be unit squares and the vertex charts n-sided polygons.

Restricting the vertices of the polyhedron to be of valence four does not limit the topologies the polyhedron can assume; a polyhedron of this restricted form can be produced from a general one by taking the dual of the first subdivision surface (see Aside 7 for a description of subdivision surfaces).

The restriction to faces of fewer than seven sides is largely for programming reasons, not theoretical ones: again, this reduces the number of cases to consider.

Aside 8 The topology of the polyhedron: *There is a correspondence between the*

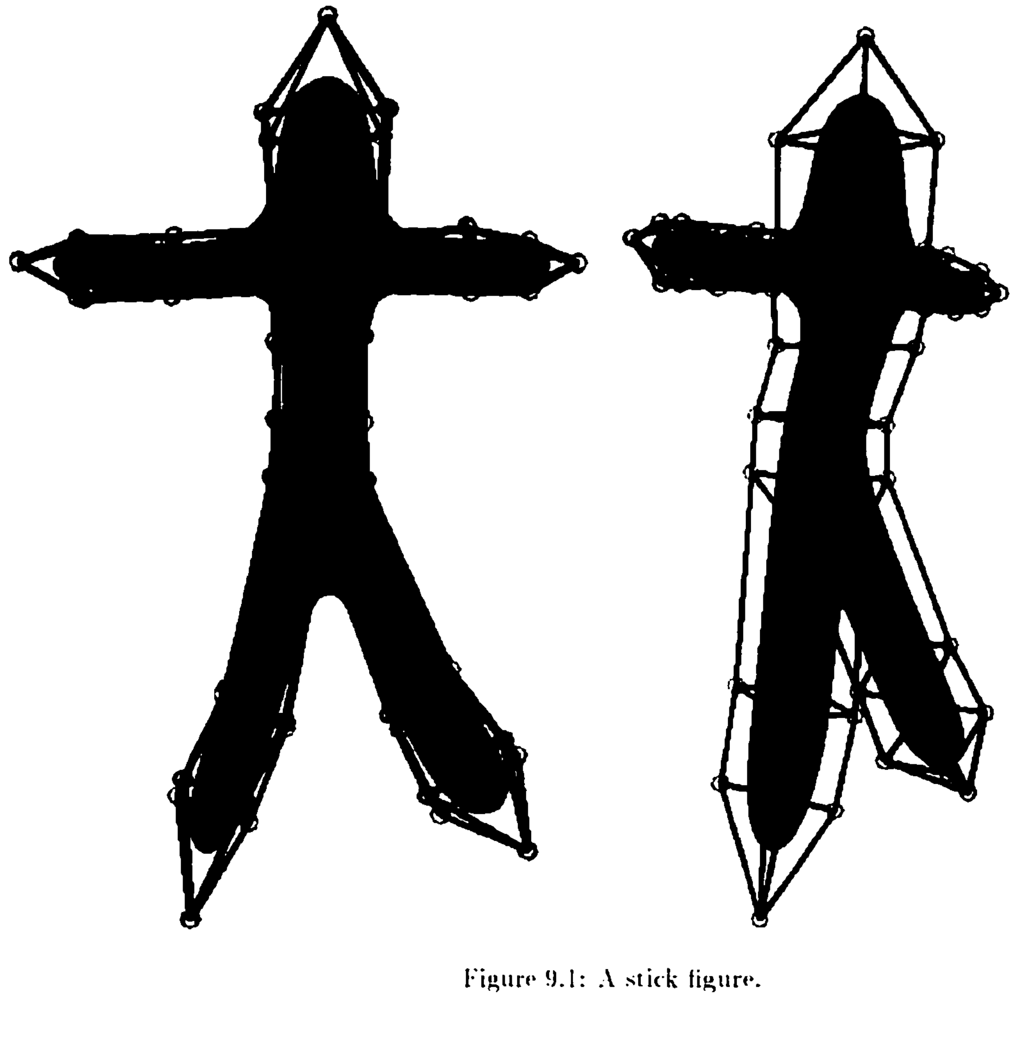


Figure 9.1: A stick figure.

number of sides of the faces and the Gaussian curvature of a valence four polyhedron: three-sided faces are equivalent to $\pi/4$ worth of curvature, four-sided faces are equivalent to zero curvature, and five-sided faces are equivalent to $-\pi/4$ curvature. As an example, consider the "stick figure" in Figure 9.1. The ends of the arms, legs, and head are capped with four three-sided polygons which "curve in" and close off the model. This area is essentially a hemisphere, which has Gaussian curvature π. The branching of the legs and arms is accomplished with four five-sided polygons. These areas are saddle points, with Gaussian curvature $-\pi$. The body, where there is no curvature, is modeled with four-sided polygons.

Obviously, other combinations of polygons can model the same figure. This particular choice has the property that the edge lengths of the polygons are all about the same (i.e., the polygons are not distorted) and the symmetry of the model is reflected in the symmetry of the polyhedron.

9.2 The manifold

The issues involved in creating a manifold are choosing the number of charts and how they overlap. This decision is influenced by the requirements on the transition functions, especially the co-cycle condition.

We chose to create one chart for each element in the polyhedron because this establishes a good correspondence between the elements of the polyhedron and the manifold and also allows substantial overlap between the charts.

Other choices were to create charts for subsets of the elements in the polyhedron, instead of all of the elements. For example, we could create one chart for each vertex, or alternatively one chart for each edge and each vertex. This approach reduces the number of charts in the manifold but increases the complexity of the overlaps and transition functions. The complexity increases because reducing the number of charts means enlarging the charts themselves, so that more than three charts will overlap in places, making the co-cycle condition difficult to meet. Because the cost of charts is relatively low, we chose simple transition functions over fewer charts.

The charts and their overlaps were chosen so that nearly every point in the manifold is in the image of a vertex, edge, and face chart, i.e., the overlaps of the charts are substantial. This allows plenty of room to move gradually from one chart to another when traversing the manifold. It also means we can define basis functions whose supports overlap substantially, even though their support is contained in different charts. This is important because splines rely on this overlap to produce smooth blends between the control points.

If nearly every point in the manifold is in the image of a set of charts (say the vertex charts), then we say that set of charts "nearly" covers the manifold. To formalize this idea, we use the

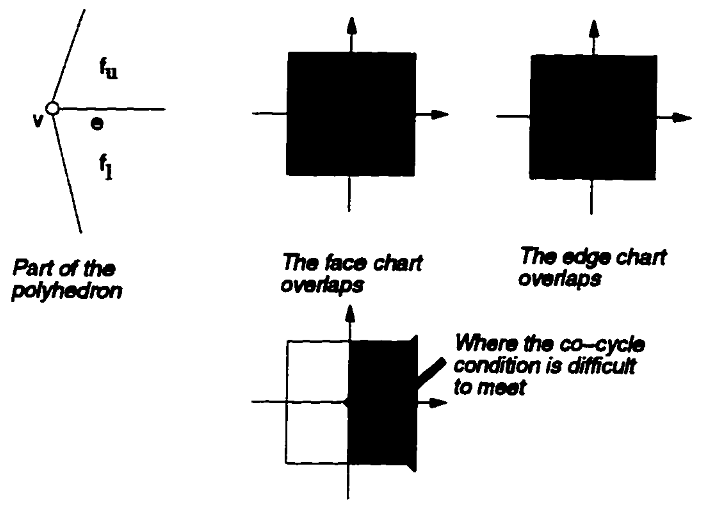


Figure 9.2: Illustration of where the co-cycle condition is difficult to meet.

following notation. Let $a \subset \mathcal{A}$ be a subset of the atlas. The set a *nearly covers* the manifold if the following holds true:

$$\bigcup_{c \in a} \overline{\alpha_c(c)} = M^K$$

where $\overline{x}$ denotes the closure of the set x.

We initially wanted each of the subsets **V**, **E**, and **F** to nearly cover the manifold but this proved to be too difficult, so we settled for having only the set **V** cover the manifold. The reason this is difficult is the co-cycle condition. Take a vertex v and an adjacent edge e. The edge e is adjacent to two faces, f_l and f_u. The vertex v must also be adjacent to these two faces. Therefore the overlap regions U_{VF_l}, U_{VF_u}, and U_{VE} are all non-empty. If each of the subsets covers the manifold then the vertex chart must be covered by $\bigcup \overline{U_{VF_i}}$ (and similarly for the U_{VE_i}). This implies that $U_{VE} \bigcap U_{VF_l} \neq \emptyset$ (and similarly for the other face, f_u). The problem is making $\varphi_{EV} = \varphi_{F_lV} \circ \varphi_{EF_l}$ agree with $\varphi_{EV} = \varphi_{F_uV} \circ \varphi_{EF_u}$ where the two regions U_{VF_l} and U_{VF_u} abut (see Figure 9.2). A similar problem arises for the other possible configurations.

To address this problem we "shrank" the face charts, which created a "gap" between the areas where three charts overlapped, leaving enough room to blend between the composed functions.

The edge charts are defined as chopped diamonds instead of diamonds because of this blending. Suppose the edge chart is a diamond; that makes the overlap region U_{EV} a triangle. If we take φ_{EV} of this triangle, the result is not a triangle in the vertex chart. In fact, depending upon the number of sides of the overlapping faces, the overlap region may map to an area larger than a triangle (see Figure 9.3). To solve this, we cut off the corners of the edge chart.

The transition functions were chosen so as to distort their domain as little as possible; some

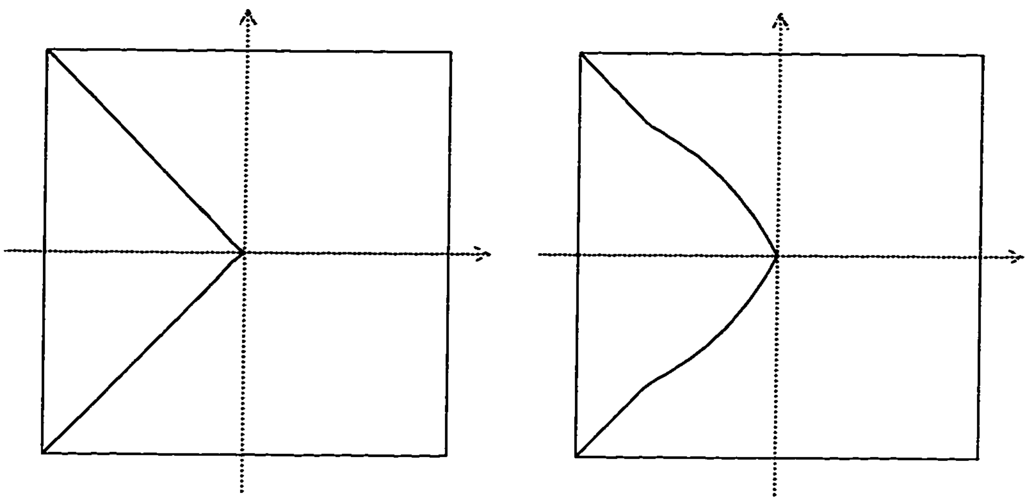


Figure 9.3: Mapping a triangle containing U_{EV} to a vertex chart via φ_{EV} $(h = 0.2)$. Left: The two faces have three sides. Right: The two faces have six sides.

distortion seems inevitable.

9.3 The immersion

The first choice to be made is the type of functions to use in the immersion. We began with splines (i.e., basis functions and control points) because we were trying to extend splines to arbitrary topologies. Another approach we have not tried, but that might prove fruitful, is wavelets – for example, something like the approach in [SS95].

To produce a satisfactory surface from B-splines we address the following four issues:

- The basis functions must be C^k.
- The basis functions must form a partition of unity.
- A sufficient number of basis functions must be nonzero at any point. The number is dependent upon the choice of k (if k increases then the number of functions overlapping at a point must also increase).
- The basis functions must be monotonically decreasing to either side of their peak, i.e., be a "bump" shape.

This list is not exhaustive list, nor are the issues independent; achieving one may be possible only at the expense of another. Ensuring these properties does not guarantee that the resulting surface is "nice," either. It is, however, a point of departure for discussing different approaches for building basis functions.

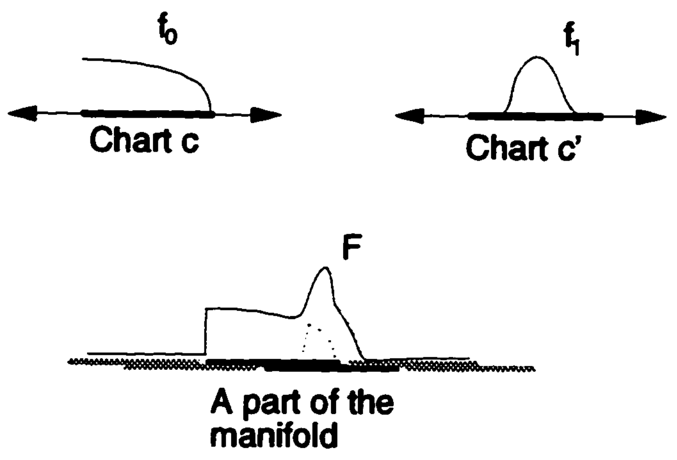


Figure 9.4: Building a manifold function F from two chart functions f_1 and f_2.

9.3.1 Ensuring C^k continuity

In Section 6.1 we discussed one method for building C^k functions on the manifold; these functions have the property that their support is contained entirely within the image of a single chart. We first recap that method, then describe a method for building functions whose support is *not* contained in a single chart.

To build a C^k function F on the manifold we first define a C^k function f in a chart c, where f and its first k derivatives are zero at the boundary of the chart. Next, we promote f to a function on the manifold using the appropriate indicator function (see Equation 6.1):

$$F(p) = \mathcal{I}_c(p) f(\alpha_c^{-1}(p))$$

We can apply a similar technique to define a function whose support is not contained in a single chart. Suppose, for example, we defined a function F from two C^k chart functions, $f_1 : c \to \Re$ and $f_2 : c' \to \Re$, as follows:

$$F(p) = \mathcal{I}_c(p) f_1(\alpha_c^{-1}(p)) + \mathcal{I}_{c'}(p) f_2(\alpha_{c'}^{-1}(p))$$

This construction is illustrated in Figure 9.4. The continuity of F is more difficult to establish than in the previous approach. The problem areas are the boundaries of the regions $\alpha_c(c)$, $\alpha_{c'}(c')$, and $\alpha_c(U_{cc'})$. Guaranteeing continuity in these areas is, in general, not easy. Suppose, however, we define a set of *blending* functions, one for each chart. These functions are similar in concept to the indicator functions except that they are C^k and form a partition of unity. Let $\mathcal{B}_c$ be the blend function for the chart c, i.e.,

$$\begin{cases} 0 \le \mathcal{B}_c(p) \le 1 & \mathcal{I}_c(p) = 1 \\ \mathcal{B}_c(p) = 0 & \mathcal{I}_c(p) = 0 \end{cases}$$

$$\forall p \in M^K, \qquad \textstyle\sum_{c\in\mathcal{A}} B_c(p) = 1$$

Then we can define a function F by defining what F looks like in several charts and blending between the results. Let $f_c : c \to \Re$ be the definition of F on the chart c. Then

$$F(p) \;=\; \sum_{c\in\mathcal{A}} B_c(p) f_c(\alpha_c^{-1}(p)) \tag{9.1}$$

This equation can be used to build functions that are C^k and whose support lies across multiple charts. The chart functions and the blend functions must be C^k, but note that the chart functions need not be zero at the boundary of their chart (since the blend functions are).

The techniques of Section 6.1 can be used to build the blend functions, since the support of a blend function βc is contained in the image of the chart c. A good chart function to use is a C^k B-spline basis function whose peak is at the origin and whose support covers as much of the chart as possible.

9.3.2 Partition of unity

Section 6.1.1 presents one method of ensuring a partition of unity. In summary, this technique produces a partition of unity from a set of proto-functions by dividing each proto-function by the sum of the proto-functions. One problem with this approach is that the original shape of the proto-functions can be greatly distorted by the division, especially if the sum of the proto-functions varies greatly.

A feature of B-spline basis functions is that they form a partition of unity because of their construction process; it is not induced afterwards. There are several different ways to construct these functions; one that can be adapted to manifolds is *convolution*, a technique that produces a set of C^k functions from a set of C^{k-1} functions while maintaining the partition of unity property. We first describe convolution in general, then the problems involved in extending it to the manifold.

Aside 9 Convolution of real-valued functions: *Convolution on the real line is defined by*

$$(f \star g)(s) = \int_{r\in\Re} f(s)g(s-r)dr$$

where f and g are functions defined on $\Re$. This process produces a new function that is defined on $\Re$ and is a "blend" of f and g. If f and g are both "nice" functions, then the function $f \star g$ will be smoother than f and g. Additionally, if we convolve g with a set of functions f_i, where $\sum_i f_i(r) = 1$ for all points $r \in \Re$, then the new set of functions $f_i \star g$ also sums to one. This is summarized in the following lemma:

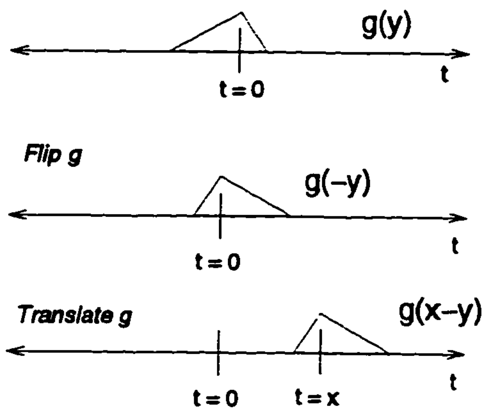


Figure 9.5: "Flipping" g around the origin and translating by x .

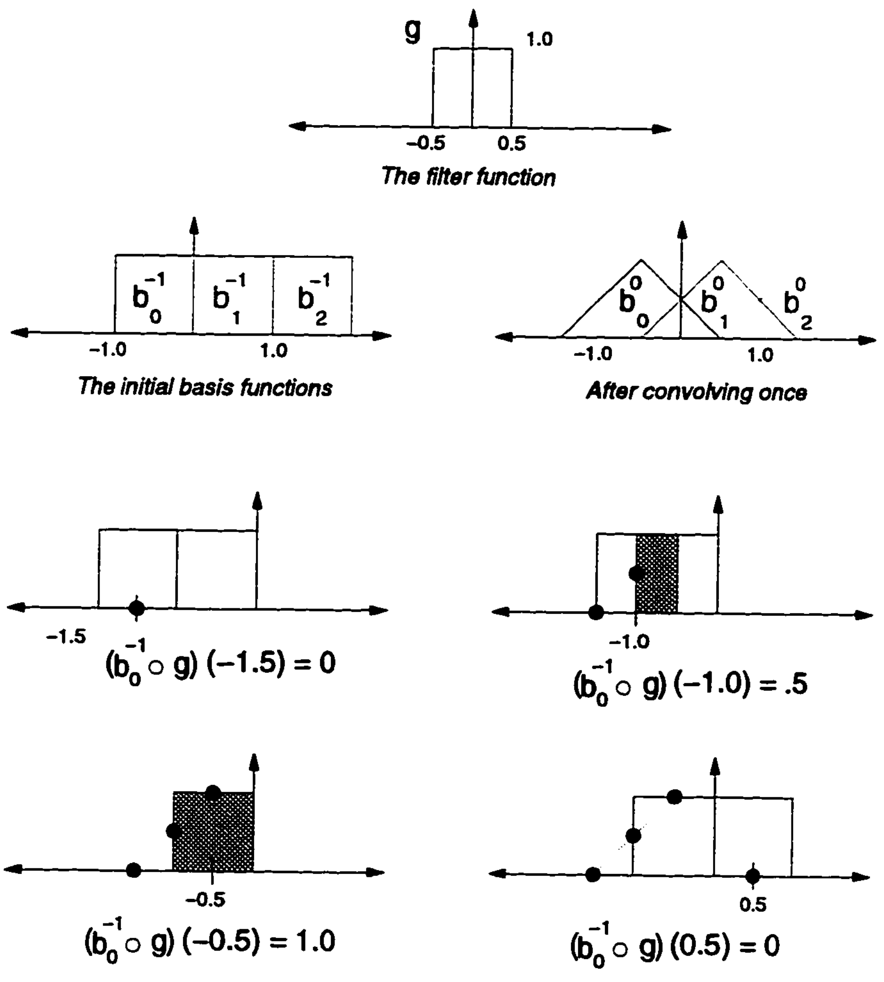

g
1.0
-0.5
0.5
The filter function
b_0^{-1}
b_1^{-1}
b_2^{-1}
-1.0
1.0
The initial basis functions
b_0^0
b_1^0
b_2^0
-1.0
1.0
After convolving once
-1.5
$(b_0^{-1} \circ g)(-1.5) = 0$
-1.0
$(b_0^{-1} \circ g)(-1.0) = .5$
-0.5
$(b_0^{-1} \circ g)(-0.5) = 1.0$
0.5
$(b_0^{-1} \circ g)(0.5) = 0$

Figure 9.6: Using convolution to construct the B-spline basis functions.

Lemma 2 Let $g : \Re \to \Re$ be a piecewise C^0 function with the property that

$$\int_{\Re} g(r)dr = 1$$

Let $i \in \{0, \dots n\}$ for $n \in \mathcal{Z}$. Let $f_i : \Re \to \Re$ be a C^{k-1} function with the property that for all $r \in \Re$

$$\sum_i f_i(r) = 1$$

Then $f_i \star g$ is a C^k function and for all $r \in \Re$

$$\sum_i (f_i \star g)(r) = 1$$

To build the uniform B-spline basis functions using convolution, let the filter function, g, be a box function of width one and height one. Let the f_i be the C^{-1} basis functions, i.e., *box functions that are one on a unit interval. The new functions, $f_i \star g$, will be the C^0 basis functions (see Figure 9.6).*

This approach works only if our filter function g integrates to one and can be "moved" to different parts of the domain via the operation $g(x - y)$. Normally, the function g is defined in $\Re^n$ and the operation $g(x - y)$ is defined by "flipping" g around the origin, translating by x, then evaluating at y (see Figure 9.5).

In our case, the operation $g(p - q)$ must be defined across the manifold, not $\Re^n$. Unfortunately, "flipping" and translating are not defined on the manifold. Instead, we define a family of functions $\{g_q(p) : M^K \to \Re\}_{q \in M^K}$, one for every point q on the manifold, and let $g(p - q) = g_q(p)$. The function g_q corresponds to the filter function "flipped" and translated to the point q.

To define an individual filter function g_q, we use the technique for defining a function across the manifold described in the previous section (see Equation 9.1). This lets us describe what the filter function looks like in each chart and blend between them.

The chart filter function $(g_q)_c$ for the chart c at the point q is defined to be nonzero on a polygonal area centered at the point $\alpha_c^{-1}(q)$. The shape of the polygonal area is determined by the chart type; if the chart is a vertex or edge chart then the polygon is a square, otherwise it is a regular polygon with the same number of sides as the chart (this is to preserve the symmetry of the convolved functions). The height of the polygon is chosen so that the integral of $(g_q)_c$ is one. The filter function on the manifold is then

$$g_q(p) = \sum_{c \in \mathcal{A}} B_c(p)(g_q)_c(\alpha_c^{-1}(p))$$

The function we are convolving with (f) is similarly defined by a blend of chart functions $f_c : c \to \Re$ (we define these later):

$$f(p) = \sum_{c \in \mathcal{A}} \mathcal{B}_c(p) f_c(\alpha_c^{-1}(p))$$

The convolution of a function f with g is computed by performing the convolution in each of the charts and blending between the results (the blending being performed by the blend functions $\{\mathcal{B}_c\}_{c \in \mathcal{A}}$):

$$\begin{aligned}
(f \star g)(p) &= \int_{q \in M^K} f(p) g(p-q) dq \\
&= \int_{q \in M^K} f(p) g_q(p) dq \\
&= \int_{q \in M^K} \sum_{c \in \mathcal{A}} \mathcal{B}_c(p) f_c(\alpha_c^{-1}(p)) \sum_{c' \in \mathcal{A}} \mathcal{B}_{c'}(p)(g_q)_{c'}(\alpha_{c'}^{-1}(p)) dq \\
&= \sum_{c' \in \mathcal{A}} \mathcal{B}_{c'}(p) \sum_{c \in \mathcal{A}} \mathcal{B}_c(p) \int_{q \in M^K} f_c(\alpha_c^{-1}(p))(g_q)_{c'}(\alpha_{c'}^{-1}(p) dq)
\end{aligned}$$

But $\mathcal{B}_c(p) = 0$ unless $p \in \alpha_c(c)$

$$= \sum_{c' \in \mathcal{A}} \mathcal{B}_{c'}(p) \sum_{c \in \mathcal{A}} \mathcal{B}_c(p) \int_{(x,y) \in c} f_c(x,y)(g_q)_{c'}(\varphi_{c'c}(x,y)) dx dy$$

Each of these integrals is confined to $\Re^2$.

Recall that we need to define the initial set of functions; in the B-spline case the initial functions were the C^{-1} basis functions, i.e., box functions. The only restriction on these functions is that they must sum to one everywhere on the manifold. One possible set is the indicator functions for the vertex charts, modified to include the part of the manifold that is not in the image of the vertex chart.

In practice, computing these convolutions for anything but the linear case proved intractable. It is that a recursive definition, such as the one developed for Box splines [DM84], might more prove tractable.

9.3.3 Number of overlapping functions

The visual smoothness of the surface depends not only on the continuity of the functions but also on the number of control points that are blended together. For example, consider an immersion of a curve defined by basis functions and control points. If only two basis functions overlap at any given point, then no matter what the continuity of the functions the "curve" will consist of blends of two control points, i.e., linear segments.

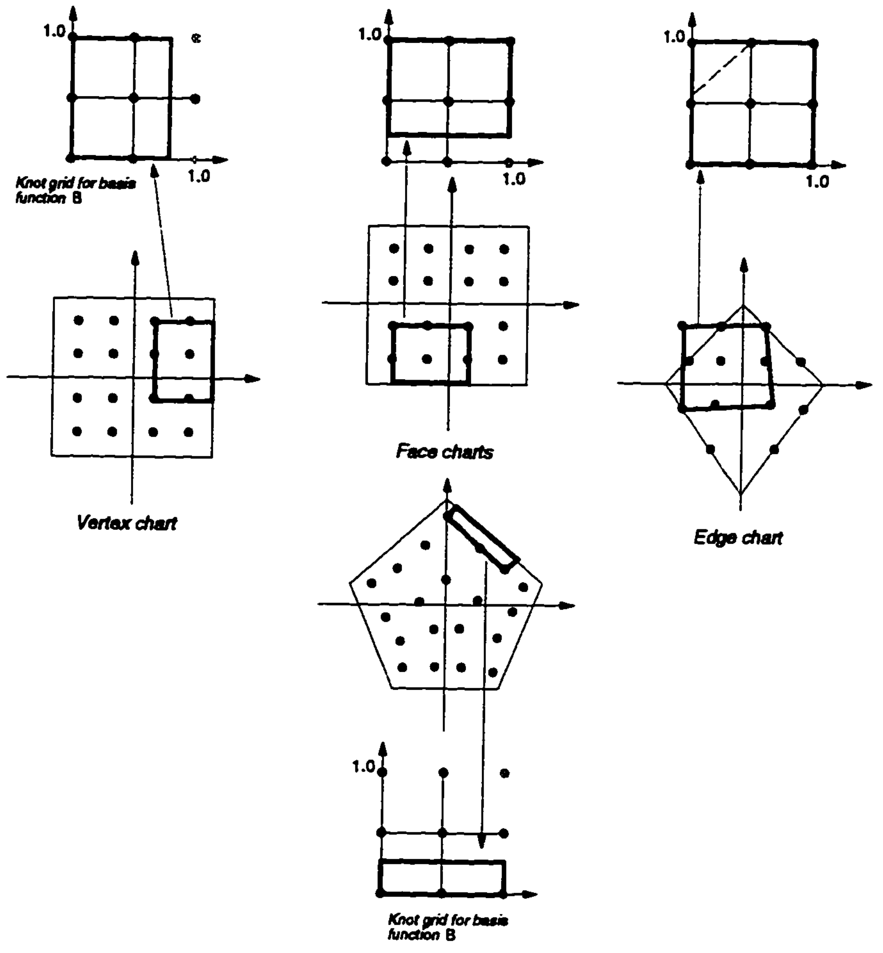
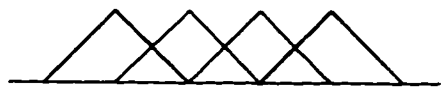

1.0
1.0
Knot grid for basis function B
Vertex chart
1.0
1.0
Face charts
1.0
1.0
Edge chart
1.0
Knot grid for basis function B

Figure 9.7: Defining a single basis function across two face charts, an edge chart, and a vertex chart. The support of the function is spread across the four charts; no singe chart contains the entire support of the function.

There are two methods for dealing with this problem: either increase the number of functions overlapping at any given point or use more complicated geometry. Increasing the number of functions that overlap involves increasing the size of the support of the functions or increasing the total number of functions. If we are defining functions whose support lies in the image of a single chart then the size of the supports of those functions is limited, so we increase the total number of functions.

If we use functions which are not confined to a single chart then it might be possible to expand the supports. We have not tried this approach, in part because defining basis functions across multiple charts is complicated. The knot set used to define the basis functions in Section 6.3 might also be used to define these functions by dropping the requirement that all of the knots used be contained in a single chart. The chart functions used to build a given



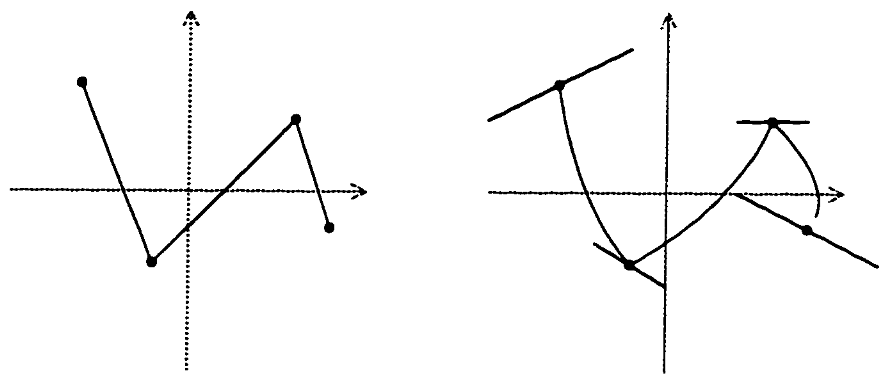


Figure 9.8: Top center: The basis functions used to create the two curves. Left: $g_i(t)$ is the constant function. Right: $g_i(t)$ is a linear function.

basis function would consist of a projective transform to part of the unit square, instead of the entire unit square (see Figure 9.7).

One other approach is to blend between some geometrical objects other than points, for example, lines or planes. Aside 10 describes this in some detail for a curve.

Aside 10 Splines with lines: *Consider the equation for a B-spline curve:*

$$\gamma(t) = \sum b_i(t) g_i$$

In this equation only the basis functions b_i depend upon the value of t; the control points g_i are constants and so do not depend upon t. We could, however, rewrite this equation so that the "control points" do *depend on t:*

$$\gamma(t) = \sum b_i(t) g_i(t)$$

where $g_i(t)$ is the constant function, i.e., *$g_i(t) = c$ for some constant c. Suppose that $g_i(t)$ is not a constant function but instead returns a point on a line. Let p be a point and v be a nonzero vector:*

$$g_i(t) = p + vt$$

Now the basis functions are blending between points on a line instead of just points. Figure 9.8 shows two B-spline curves, both having the same basis functions but different g_i functions.

The two difficulties with this approach are how to parameterize the geometrical object and what shape it should take. This is not too difficult for a plane (although some care must be taken with the orientation of neighboring planes) but extending this even to parabolas is difficult.

9.3.4 The shape of the basis functions

One problem with using division to create a partition of unity is that the shape of the final functions can be greatly distorted from the shape of the proto-functions. If the sum of the proto-functions is relatively constant, or if it changes slowly relative to the size of the supports of the proto-functions, then this problem is lessened.

Chapter 10

Implementation

A working interactive editor exists for the HP735, Sun SPARCstation 10, and the SGI Onyx. The editor consists of a polyhedral modeler that lets the user interactively construct a general polyhedron by creating vertices and connecting them together to form faces. A valence-four polyhedron and its corresponding surface are automatically constructed from this general polyhedron. The desired continuity and triangulation of the surface are specified at startup time. The user may manipulate the positions of the valence four polyhedron and the control points of the surface (initial positions are chosen using the subdivision surface of the general polyhedron). Existing model sizes range from 314 to 1112 polygons. Table 10.1 shows the times for calculating a C^k surface from scratch on a SPARCstation 10, triangulated with $(2d)^2$ triangles per vertex chart. The first sample surface is shown in Figure 10.1, the second in Figure 9.1.

As a benchmark, the image in Figure 1.2 was created in approximately six hours and the image in Figure 10.2 was created by a novice user in approximately four hours. The code is a straightforward implementation, without any attempt at optimization, of the equations presented here.

10.0.5 Results

Example control polyhedron and their corresponding surfaces are shown in Figures 10.3 through 10.5. All of these models were constructed entirely by hand. The initial control polyhedron for Figure 10.6 was constructed by hand. The locations of the vertices for the corresponding valence-four polyhedron were then adjusted, using a simple data-fitting algorithm, to bring the surface closer to the laser-scanned polyhedron [TL94] shown on the left of Figure 10.6. The valence-four polyhedron for Figure 10.7 was constructed automatically using the technique described in [GH95].

Figure 10.8 illustrates the effect of moving the vertices of the control polyhedron. Figure 10.9 illustrates the effect of moving individual control points on surfaces of varying continuity.

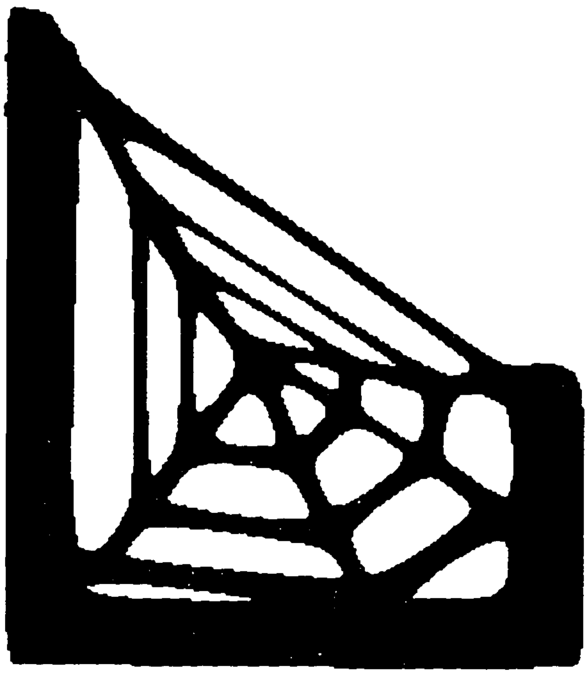


Figure 10.1: A spiderweb (764 polygons).

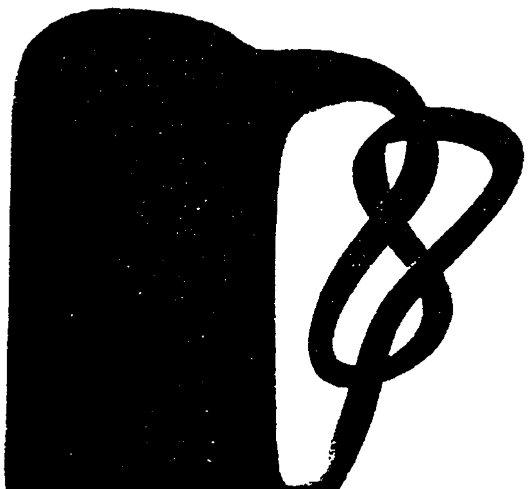


Figure 10.2: Alexander's two-holed torus (596 polygons).

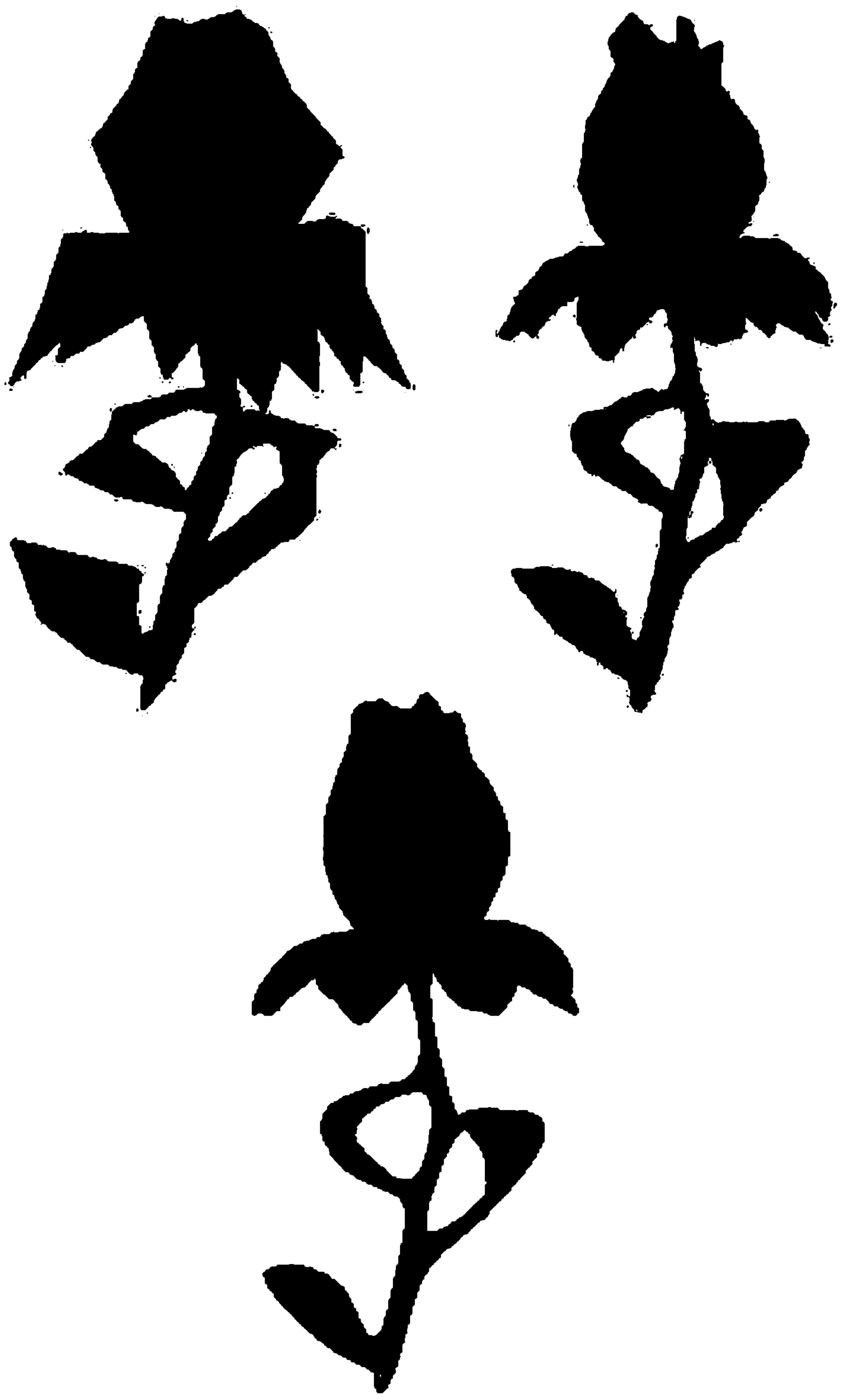


Figure 10.3: Top left: The initial polyhedron. Top right: The valence-four polyhedron. Bottom: The C^1 surface, tesselated with 5^2 triangles per vertex chart.

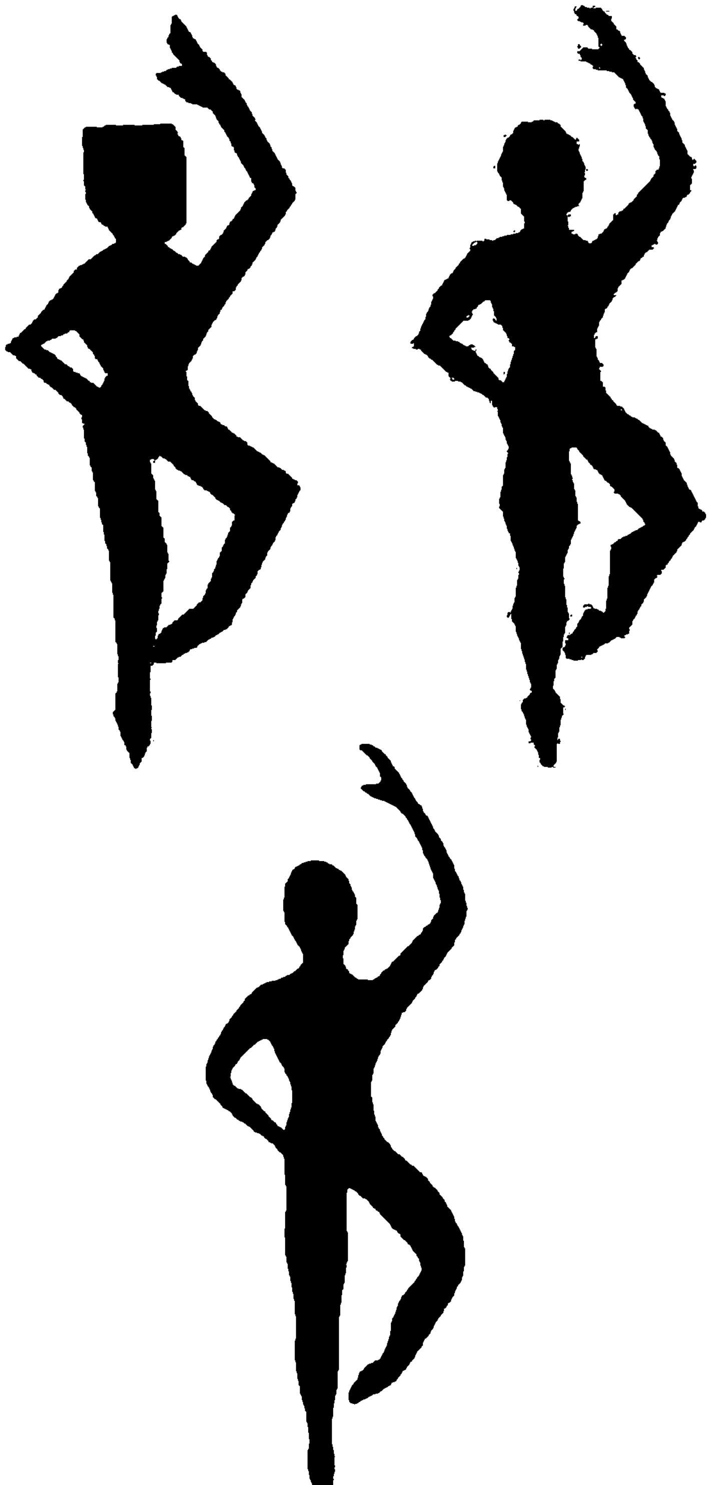


Figure 10.1: Top left: The initial polyhedron. Top right: The valence-four polyhedron. Bottom: The C^1 surface, tesselated with 5^2 triangles per vertex chart.

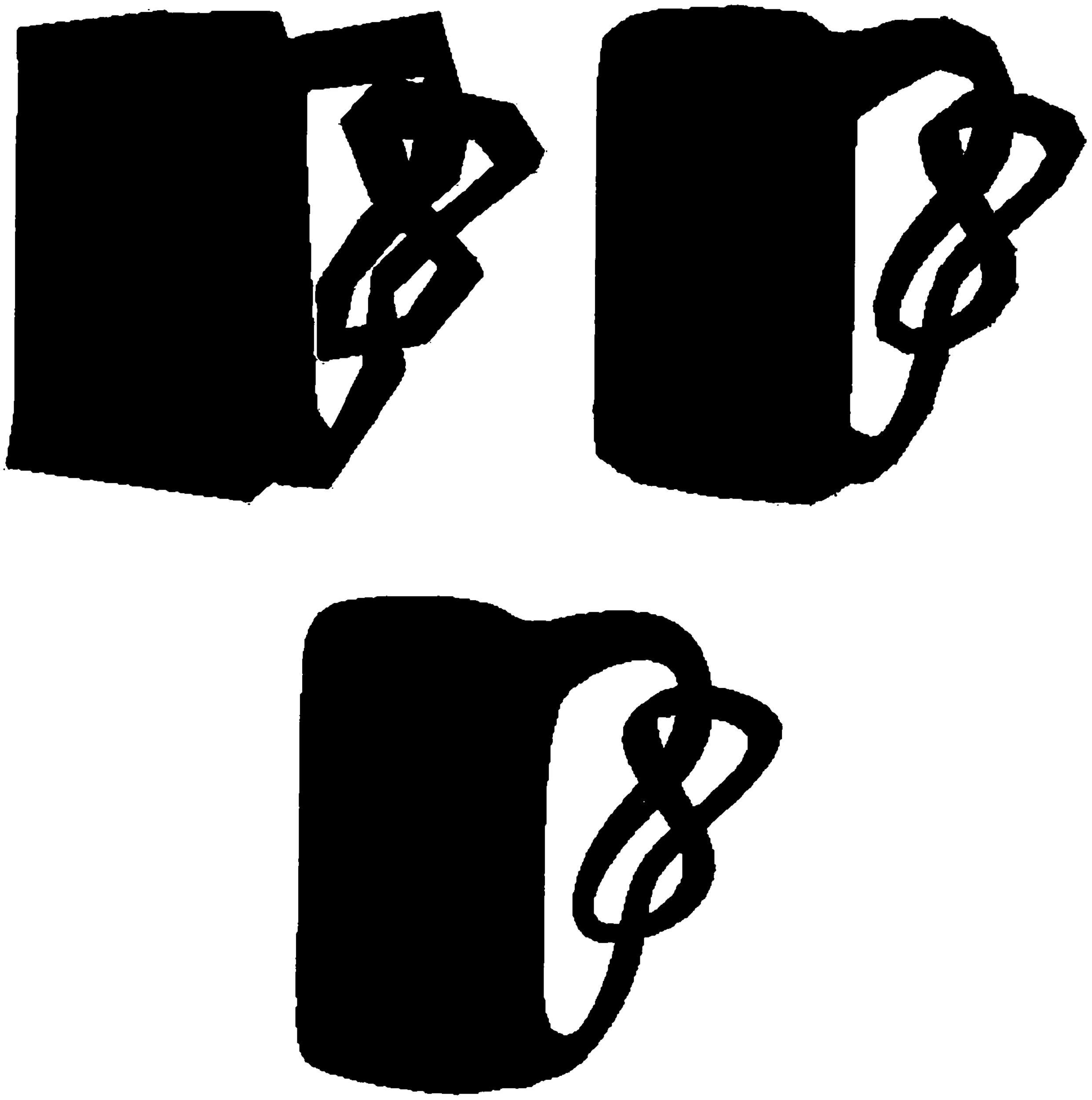


Figure 10.5: Top left: The initial polyhedron. Top right: The valence-four polyhedron. Bottom: The C^1 surface, tesselated with 5^2 triangles per vertex chart.

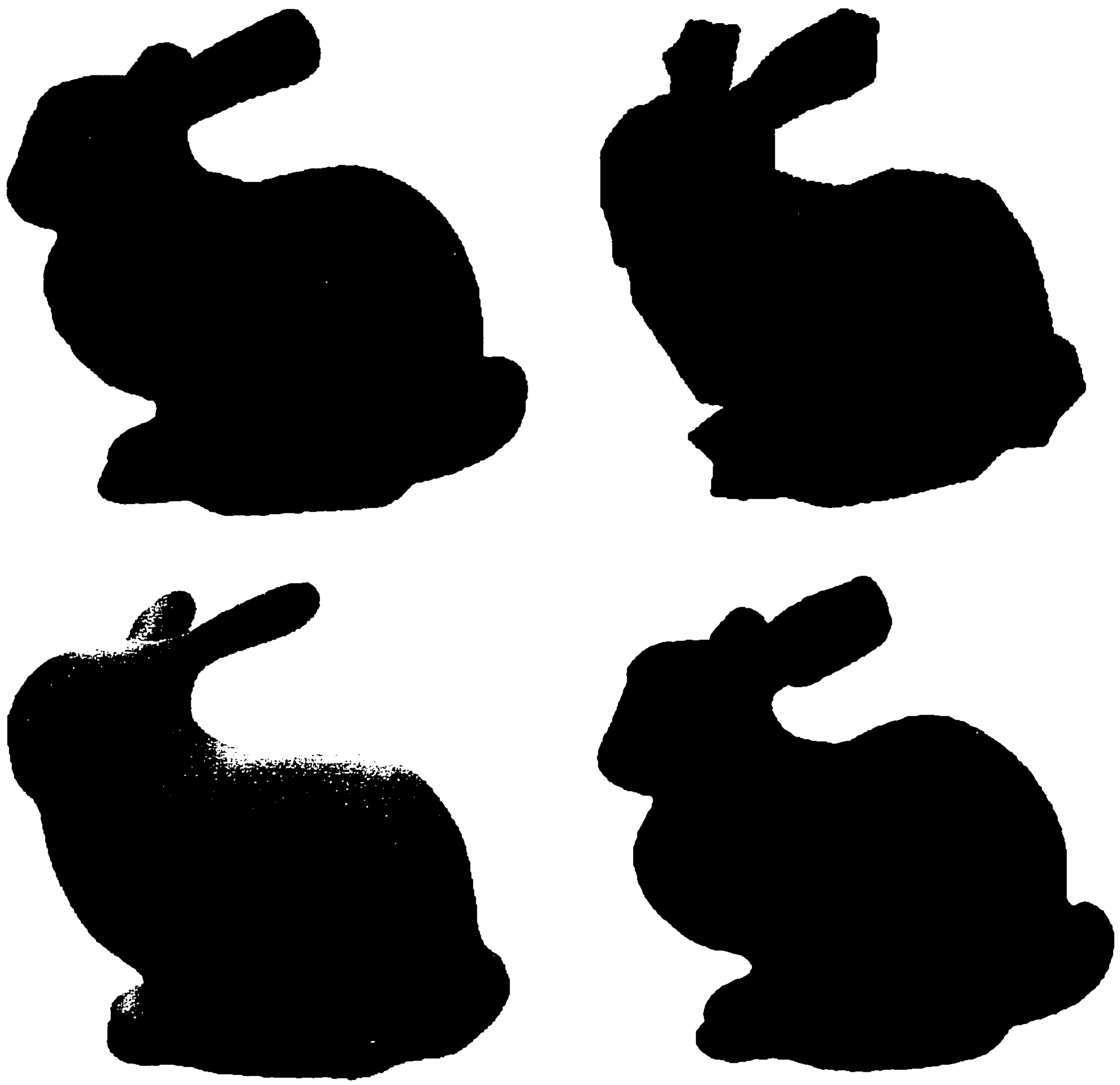


Figure 10.6: Upper left: The laser scanned data. Upper right: The control polyhedron (built by hand). Lower left: The surface before data fitting. Lower right: The surface after data fitting.

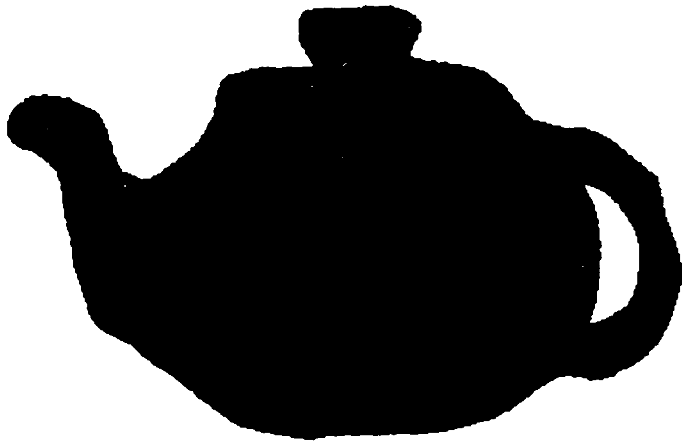


Figure 10.7: Top: The initial isosurface. Bottom: The C^1 surface, tesselated with 5^2 triangles per vertex chart.

Times for the spiderweb (764 polygons)			
	k = 0	k = 1	k = 2
d = 3	33:42	2:43:38	7:14:57
d = 5	37:56	3:10:07	10:13:53
d = 7	50:52	3:17:19	11:54:02

Times for the stick figure (264 polygons)			
	k = 0	k = 1	k = 2
d = 3	7:13	23:38	1:05:51
d = 5	8:29	28:17	1:17:01
d = 7	10:24	33:10	1:29:54

Table 10.1: Times (in minutes) for constructing a surface of continuity C^k with d^2 triangles per vertex chart. Tests were run on a SPARCstation 10.

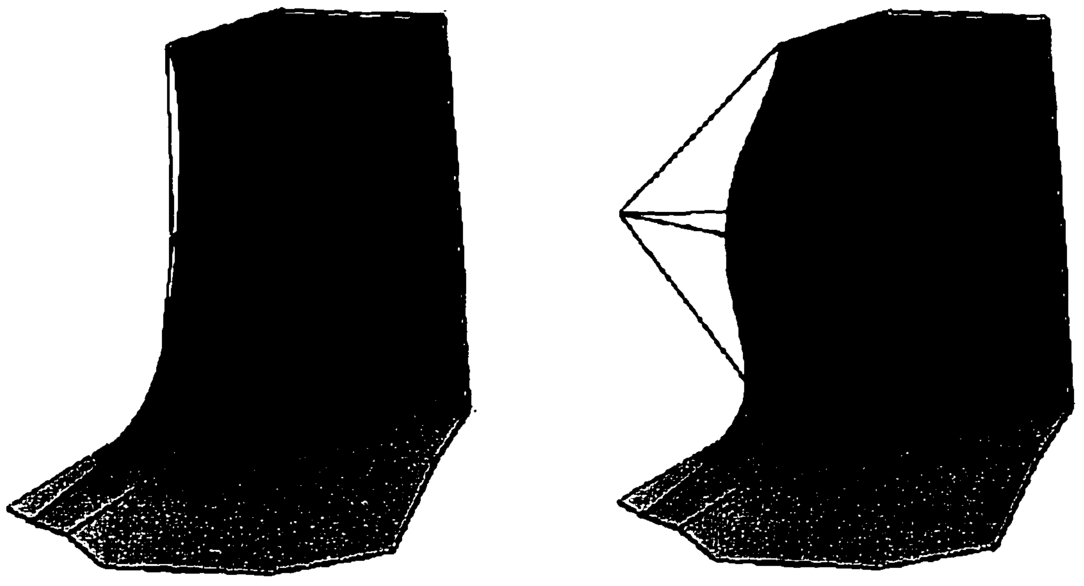


Figure 10.8: Left: Before moving the vertex. Right: After moving the vertex.

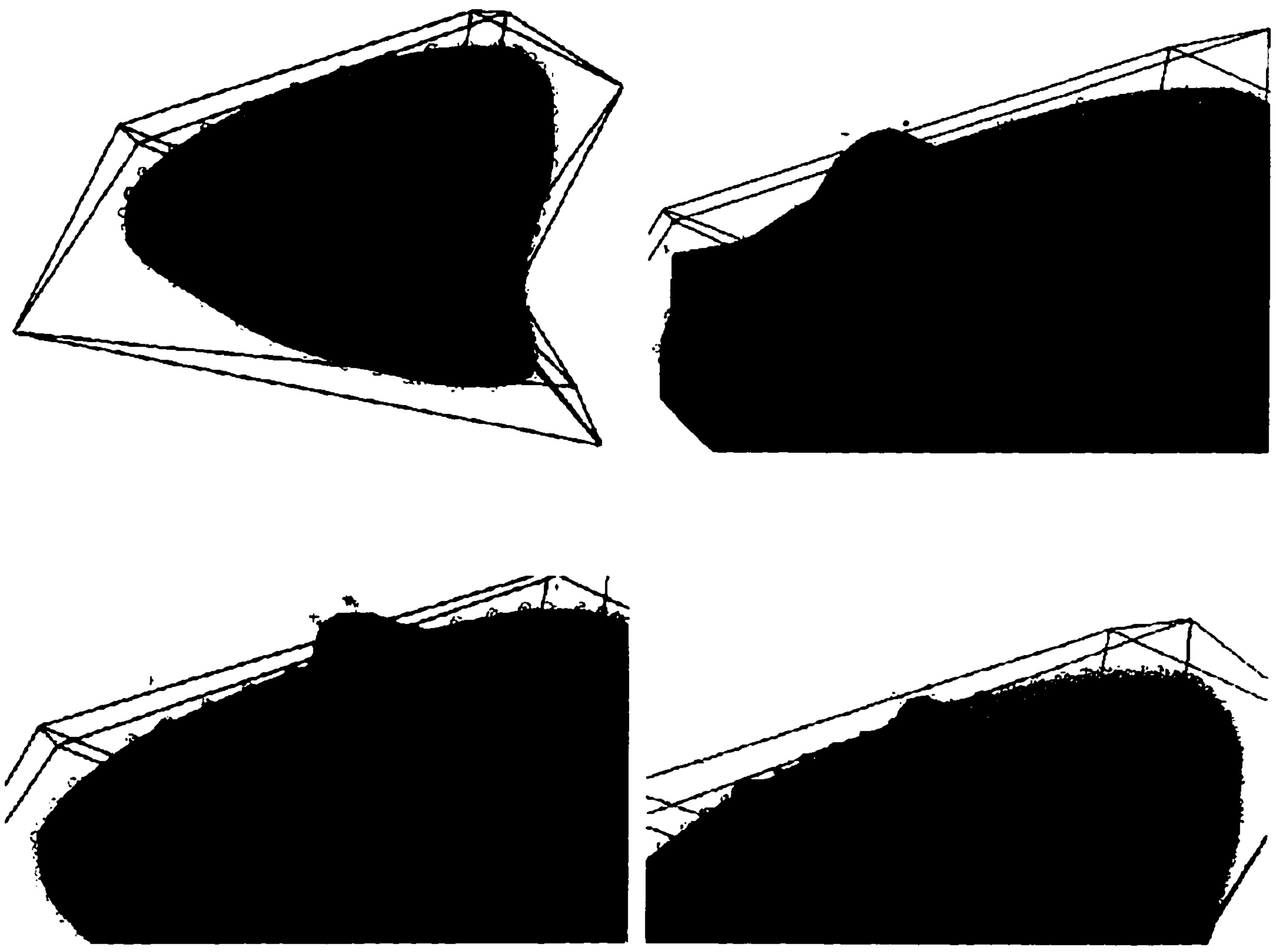


Figure 10.9: Top left: Initial C^0 surface. Top right: Moving control points on the C^0 surface. Bottom left: Moving control points on the C^1 surface. Bottom right: Moving control points on the C^2 surface.

Chapter 11

Future work

The technique presented here produces a C^k surface of arbitrary topology that has many of the properties of a spline surface. Several techniques exist for manipulating spline surfaces, however, that we have yet to develop for manifolds. The existence of these techniques is part of what makes splines a powerful modeling tool; developing similar techniques for manifolds is a necessary next step for introducing manifolds into the mainstream.

In this section we first discuss several spline manipulation techniques, how they are implemented for splines, and what the difficulties are in developing them for manifolds (Section 11.1). Following this we discuss other, more general directions for this research (Section 11.2).

11.1 Spline manipulation techniques

We discuss four basic spline manipulation techniques. These four do not form an exhaustive list; they do, however, illustrate some of the differences between manipulating functions defined on $\Re^n$ and functions defined on a manifold. For each technique we describe the technique for traditional splines, how one might approach the problem for manifolds, and what the difficulties are.

11.1.1 Refinement and subdivision

Refinement is best illustrated using spline curves. Descriptions of refinement and further references can be found in [BBB87] and [Far88].

Let $\gamma : \Re \to \Re^2$ be a NUBS curve of degree k with basis functions defined by the knot vector r (refer to Section 6.2.1). We can *refine* γ to produce a new curve γ' with one more degree of freedom (i.e., one more control point) than γ. This is accomplished by adding a new knot into the knot vector r to produce a new knot vector r' from which the basis functions for γ' are constructed. Since r' has one more knot than r, γ' has one more control point than γ. This would not be terribly interesting except that we can chose locations for the control points of γ' such that $\gamma'(t) = \gamma(t)$ for all $t \in \Re$ (and, in fact, these locations are unique). This is possible because the basis functions for γ' span the basis functions for γ. By this we mean that each basis function for γ can be written as a linear combination of the basis functions for γ'. Exactly how this is done can be found in [Far88].

Subdivision is a similar process used to produce two curves from a single curve, such that the two curves together form the original curve.

The heart of refinement is being able to create a nested set of spaces of curves, $\mathcal{W}^0 \subset \mathcal{W}^1, \ldots$, where each space $\mathcal{W}^i$ consists of an infinite number of curves. An example of such a space is the space consisting of all piece-linear curves with two segments. A curve in $\mathcal{W}^i$ is a linear combination of the n_i functions that form a basis for $\mathcal{W}^i$. Let $\{w_j^i\}_{1 \leq i \leq n_i}$ be a basis for the space $\mathcal{W}^i$. An element γ in $\mathcal{W}^i$ is expressed as

$$\gamma(t) \in \mathcal{W}^i = \sum_{j=1}^{n_i} a_j w_j^i(t)$$

where $a_j \in \Re^n$. Given a basis for the space $\mathcal{W}^i$, an element of $\mathcal{W}^i$ can be defined by the vector $a_j \in \Re^n$.

Since $\mathcal{W}^i$ is a subspace of $\mathcal{W}^{i+1}$ each element in the basis for $\mathcal{W}^i$ can be expressed in the basis for $\mathcal{W}^{i+1}$; e.g., for all j such that $1 \leq j \leq n_i$ there exists $a_j \in \Re$ such that

$$w_k^i(t) \quad = \quad \sum_{j=1}^{n_{i+1}} a_j^k w_j^{i+1}(t)$$

This implies that any element in $\mathcal{W}^i$ can be expressed in the basis for $\mathcal{W}^{i+1}$. This is exactly what happened in the spline example above. Let $\{w_j^i\}_{1 \leq i \leq n_i}$ be the basis functions for γ and $\{w_j^{i+1}\}_{1 \leq i \leq n_{i+1}}$ be the basis functions for γ'. Then each basis function w_j^i can be expressed as a linear combination of the $\{w_j^{i+1}\}_{1 \leq i \leq n_{i+1}}$; hence we can find a linear combination of the $\{w_j^{i+1}\}_{1 \leq i \leq n_{i+1}}$ which gives us γ:

$$\begin{aligned}
w_k^i(t) &= \sum_{j=1}^{n_{i+1}} a_j^k w_j^{i+1}(t) \\
\gamma(t) &= \sum_{k=1}^{n_i} g_k w_k^i(t) \\
&= \sum_{k=1}^{n_i} g_k \sum_{j=1}^{n^{i+1}} a_j^k w_j^{i+1}(t) \\
&= \sum_{j=1}^{n_{i+1}} \hat{g}_j w_j^{i+1}(t) \\
&= \gamma'(t)
\end{aligned}$$

It is a property of NUBS curves that each of the basis functions w_j^i created from the knot vector r can be expressed as linear combinations of the basis functions w_j^{i+1} created from the knot vector r'.

Developing a similar property for manifolds involves developing basis functions which also have this nesting property. In particular, given a set of n basis functions $\{B_j^i\}_{1\leq j\leq n}$ we need to find a new set $\{B_j^{i+1}\}_{1\leq j\leq m}$ of m basis functions with $m > n$ such that each of the B_j^i can be written as a linear combination of the $\{B_j^{i+1}\}_{1\leq j\leq m}$ basis functions.

Recall that the manifold basis function B_j^i is the result of promoting and normalizing a chart function $(b_j^i)_c$ defined in some chart c ($(b_j^i)_c : c \rightarrow \Re$). Let $\hat{B}_j^i$ be the chart function $(b_j^i)_c$ promoted to the manifold:

$$\hat{B}_k^i(p) = \mathcal{I}_c(p)(b_k^i)_c(\alpha_c^{-1}(p))$$

Instead of directly defining the basis functions $\{B_k^{i+1}\}_{k\in[1,m]}$, we define a new set of chart functions $\{(b_k^{i+1})_c : c \rightarrow \Re\}_{k\in[1,m]}$ such that each of the old chart functions can be expressed as a linear combination of the new ones defined in that same chart, i.e.,

$$(b_k^i)_c(t) = \sum_{\substack{1\leq j\leq m \\ b_j^{i+1} : c\rightarrow \Re}} a_j^k (b_j^{i+1})_{c'}(t)$$

and the corresponding functions on the manifold (the a_j^k not defined by the above equation are set to zero):

$$\hat{B}_k^i = \sum_{j=1}^{m} a_j^k \hat{B}_j^{i+1}$$

To define the new basis functions B_k^{i+1} we expand the following:

$$\begin{aligned}
B_k^i(p) &= \frac{\hat{B}_k^i(p)}{\sum_{l=1}^n \hat{B}_l^i(p)} \\
&= \frac{\sum_{j=1}^m a_j^k \hat{B}_j^{i+1}(p)}{\sum_{l=1}^n \sum_{r=1}^m a_r^l \hat{B}_r^{i+1}(p)} \\
&= \sum_{j=1}^m a_j^k \frac{\hat{B}_j^{i+1}(p)}{\sum_{l=1}^n \sum_{r=1}^m a_r^l \hat{B}_r^{i+1}(p)}
\end{aligned}$$

which means that B_k^{i+1} must be

$$\frac{\hat{B}_k^{i+1}}{\sum_{l=1}^n \sum_{r=1}^m a_r^l \hat{B}_r^{i+1}} = \frac{\hat{B}_k^{i+1}}{\sum_{l=1}^n \hat{B}_l^i}$$

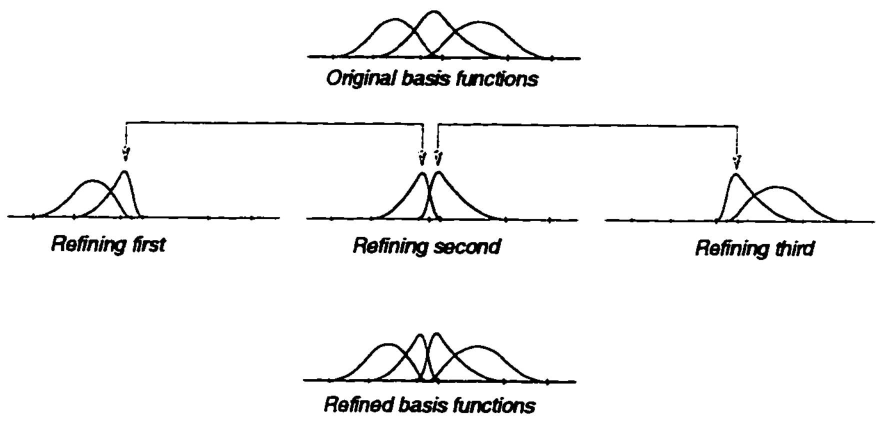


Figure 11.1: Individually refining the basis functions of a curve. Note that pairs of functions are identical.

We can therefore build the new set of basis functions by building a new set of chart functions that forms a basis for the old chart basis functions. Note that the new set of functions is not a partition of unity unless the denominator of the above equation is one, which happens only when the original proto-basis functions summed to one at that point.

Recall that each chart function $(b_k^i)_c$ is defined by a projective mapping ζ composed with a NUBS basis function b defined on the unit square (see Equation 6.5). Suppose we refine b using standard spline techniques to form a set of n functions b_i' such that

$$b(s,t)\sum_{j=1}^{n} a_j b_j'(s,t)$$

Then the new basis functions

$$\{b_j'(\zeta(s,t))\}_{1\leq j\leq n}$$

span the old chart function

$$\begin{aligned} b(s,t) &= \sum_{j=1}^{n} a_j b_j'(s,t) \\ \rightarrow b(\zeta(s,t)) &= \sum_{j=1}^{n} a_j b_j'(\zeta(s,t)) \end{aligned}$$

Suppose we refine each of the old chart basis functions in this manner. In the rectangular regions of the manifold many of the new basis functions will be duplicates of each other: to

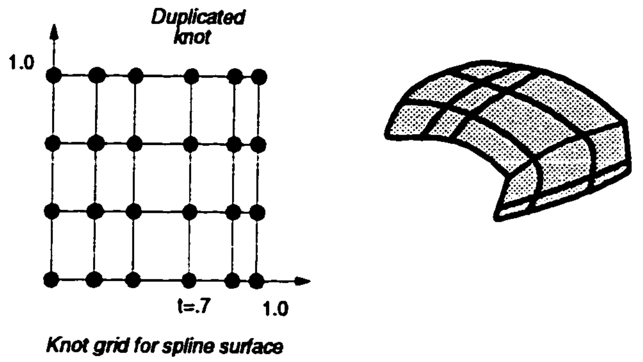


Figure 11.2: Introducing a discontinuity into one of the knot vectors used to build the knot grid for a a spline surface. This results in a duplicated knot line in the knot grid (the red line in the knot grid) and a line of discontinuities in the surface (shown on the right).

see this, imagine refining the basis functions for a curve by refining each of the basis functions individually (see Figure 11.1). This is because the knot lines in the supports of the old chart functions are aligned with the knot lines of overlapping chart functions. In the non-rectangular regions this is not the case and hence the number of new functions is dramatically increased.

11.1.2 Introducing discontinuities

Discontinuities are useful for modeling creases in curves and surfaces. One method for introducing a discontinuity into a curve is to duplicate a knot a sufficient number of times [BBB87]. Each time a knot is duplicated the continuity of the curve at that knot drops by one. Another approach is to duplicate adjacent control points; the corresponding technique for subdivision surfaces alters the blend functions to introduce discontinuities along edges or at vertices [HDD+94].

Introducing a random discontinuity into a spline surface is not simple. One way to do it is to duplicate the knots of one of the curves making up the spline surface. This produces a discontinuity in the surface along a constant parameter, i.e., the curve $S(s, t_0)$ where t_0 is constant (see Figure 11.2).

This approach can be used in manifold surfaces in the four-sided regions by introducing a discontinuity in each of the basis functions. Note that these discontinuities can lie only along lines parallel to the axes of the charts.

11.1.3 Curves immersed in a surface

A curve $\gamma : \Re \to \Re^3$ is immersed in a surface $S : \Re^2 \to \Re^3$ if every point of the curve lies on the surface, i.e., $\gamma(t) = S(u, v)$ for all t in the domain of γ.

The ability to define a curve immersed in a surface is useful for a variety of techniques, such as trimming surfaces, blending between surfaces, and constraining the surface to pass through a curve. One method for immersing a curve in a spline surface is to define a curve $\tilde{\gamma} : \Re \to \Re^2$ in the domain of the surface. The composition of this curve with the surface ($\gamma = S \circ \tilde{\gamma}$) is then a curve immersed in that surface. If $\tilde{\gamma}$ is C^{k_0} and S is C^{k_1} then the immersed curve γ is of continuity $(k_0 + 1)(k_1 + 1) - 1$.

We can use the same method to construct a curve immersed in a manifold surface, but building the curve and determining its continuity are more complicated. Let $\gamma : \Re \to M^K$ be a curve lying in the manifold, and let $\mathcal{E}_M : M^K \to \Re^3$ be an immersion function for the manifold. Then the immersed curve is

$$\mathcal{E}_M \circ \gamma : \Re \to M^K \to \Re^3$$

To construct γ we build it in consecutive segments, each of which lies in a chart. The end of segment i is blended with the beginning of the following segment, so as we move along the curve we switch slowly from the definition in one chart to the definition in an adjacent chart.

More formally, define a set of n functions, $\{\gamma_i : [i-1, i+1] \to c_i\}_{i \in [1,\ldots,n]}$, where $c_i \in \mathcal{A}$. For all $i \in [1, \ldots, n-1]$ define δ_i such that $i < \delta_i < i + 1$. The set of functions must have the following properties:

- The charts for adjacent functions must overlap, i.e., $\forall i \in [1, \ldots, n-1]$, $U_{c_i c_{i+1}} \neq \emptyset$.
- For each pair of functions γ_i and γ_{i+1}, the end of γ_i and the beginning of γ_{i+1} must lie in the overlap region of the respective pair of charts, i.e., $\forall t \in [i, \delta_i]$, $\gamma_i(t) \in U_{c_i c_{i+1}}$ and $\gamma_{i+1}(t) \in U_{c_{i+1} c_i}$.
- γ_i is C^k, where the continuity of the manifold is at least C^k (γ_i can be of greater continuity than the manifold but the blended function γ will have at best the continuity of the manifold).

To blend between the two functions γ_i and γ_{i+1} we need a blend function $\beta_i : \Re \to [0, 1]$ with the following properties:

- β_i is C^k.
- $\beta_i(t) = 0$ for $t \leq i$.
- $\beta_i(t) = 1$ for $t \geq \delta_i$.

Then we can define $\gamma : [0, n] \to M^K$ as follows ($1 \leq i \leq n$):

$$\gamma(t) = \begin{cases} \alpha_{c_1}(\gamma_1(t)) & t < 1 \\ \alpha_{c_i}\left((1 - \beta_i(t))\gamma_i(t) + \beta_i(t)\varphi_{c_i c_{i+1}}(\gamma_{i+1}(t))\right) & i < t \leq i + 1 \\ \alpha_{c_n}(\gamma_n(t)) & t \geq n \end{cases}$$

The continuity of γ is the continuity of

$$\mathcal{E}_M \circ \gamma$$

which, by the definition of continuity on the manifold (Definition 5), is the continuity of one of the following:

$$\begin{cases} \gamma_1(t) & t < 1 \\ (1-\beta_i(t))\gamma_i(t) + \beta_i(t)\varphi_{c_i c_{i+1}}(\gamma_{i+1}(t)) & i < t \leq i+1 \\ \gamma_n(t) & t \geq n \end{cases}$$

all of which are C^k by definition.

There remains the question of building the individual functions γ_i. Ideally, the end of the curve γ_i, when mapped into the chart c_{i+1} via the transition function, should look as much like the beginning of the curve γ_{i+1} as possible, or even be the same curve. One way to do this is to map the end of γ_i into the chart c_{i+1} and use that as the beginning of the curve γ_{i+1}.

Suppose we define γ_i to be a C^k NURB with its endpoint in the overlap region $U_{c_i c_{i+1}}$. Then we refine γ_i so that the last segment of the curve is contained in the overlap region. Then a good choice for the first $k+2$ control points of γ_{i+1} is $\varphi_{c_i c_{i+1}}$ of the last $k+2$ control points of γ_i. Additionally, set the first $k+3$ knots to be a shifted copy of the last $k+3$ knots of γ (so that the $(k+2)$th knot lies at the origin). Set δ_k to the first knot of the last segment of γ_i. Some care must be taken to ensure that the first segment of γ_{i+1} actually lies within the overlap region $U_{c_{i+1} c_i}$; this can usually be guaranteed using the convex hull property for NURBS.

Note that if the transition function $\varphi_{c_i c_{i+1}}$ is a projective transform then the last segment of γ_i will be identical to the first segment of γ_{i+1} (which is why we chose NURBS instead of the simpler NUBS). If this is the case then the blend function is redundant and, practically speaking, we can simply string the curve segments together by setting the first $k+1$ control points of γ_{i+1} to be the projective transform of the last $k+1$ control points of γ_i. γ in this case is just $\alpha_{c_i}(\gamma_i)$ followed by $\alpha_{c_{i+1}}(\gamma_{i+1})$.

11.1.4 Parameter-space-based texture mapping

One way to introduce visual complexity into a scene is to color it by some texture. The texture T is usually defined to be a map from a part of the plane, say the unit square, to a color triple:

$$T : [0,1] \times [0,1] \to \Re^3 : (u,v) \to ((r(u,v), g(u,v), b(u,v)))$$

To apply this texture to a surface we need to establish a correspondence between points on the surface and points in the texture: we need to assign a (u,v), $u, v \in [0,1]$ value to each point in the part of the surface to be covered by the texture. The color of the surface at that point is then $T(u, v)$. Alternatively, if a surface S is already parameterized ($S : D \rightarrow \Re^3$) then we can achieve the same effect by mapping the parameter values D into the domain of T. Let $T_D : D \rightarrow [0, 1] \times [0, 1]$. Then the color of the surface at a point $S(d), d \in D$, is $T(T_D(d))$.

For spline surfaces (and a few other surfaces such as cylinders and tori) there is a natural map from the domain of the surface to the domain of the texture. In some other cases, such as spheres and surfaces comprised of multiple patches, there is a mapping that *almost* works. The existence of these cases has resulted in the assumption that the domain of the surface should always be used when doing texture mapping. In fact, it may be better to define a mapping from the surface itself to the texture (and for surfaces without a parameterization, e.g., implicit surfaces, it is necessary). Creating such a parameterization for arbitrary surfaces, however, can be very difficult [Ped95].

Defining a single mapping for the entire surface is usually unnecessary (and sometimes impossible); we really need only define a parameterization for the part of the surface to be covered by the texture map.

One advantage of manifolds is that their parameterization is very flexible, which simplifies defining a map from part of the surface to the texture domain. If the relevant part of the surface is in the image of a single chart then we can define a map from part of the image of the chart to the texture domain. In most cases, however, the texture will cover a larger portion of the surface. The domain of the mapping will therefore contain the image of multiple charts. Because the charts overlap, we can define a mapping for each individual chart and then blend between the individual mappings.

11.2 Other directions

There are other research directions to explore in addition to those related to splines. First, there may be alternative methods for constructing the manifold itself as well as the immersion. Secondly, there are certainly better methods for specifying the desired surface than building a polyhedron by hand. Finally, there is the interesting, open question of how to fit a surface to a data set automatically or semi-automatically.

11.2.1 Building the manifold and the immersion

We have already discussed several other possible immersion functions. For example, an immersion function based on wavelets would be ideal for surfaces having multiple levels of detail. For a spline-based immersion function, is there a better set of basis functions than the ones currently used? The current set of functions rely on division to produce a partition of unity and do not have a natural refinement algorithm.

The transition functions are largely projective transforms, which have some nice properties. Ideally, all of the transition functions would be projective transforms. It is not clear, though, whether there is a reasonable set of charts and overlaps that allow all of the transition functions to be projective transforms.

11.2.2 Sketching models

Currently surfaces are constructed using a polyhedral modeler. Although this is sufficient, deciding what set of polygons are appropriate to sketch the shape of the model can be time-consuming and difficult. A better interface would be one that lets the user sketch the surface in a natural way, for example by building the surface out of "clay."

11.2.3 Data fitting

A different approach to acquiring a surface is to scan in a real object, using, for example, a laser scanner [TL94] or an MRI scan [MBL+91a][MBL+91b][Lai95]. The resulting model is typically a large collection of triangles or a volume data set (which can be converted to a set of triangles). Although the data is often useful in its original form, there are many reasons for converting it to a more structured representation such as patches or solids models: in order to reverse engineer a part, include the object in an existing model, or to animate the data. This is the subject of data fitting.

One advantage of manifolds is that the local topology of the sketch can be made to approximate the final desired geometry. This makes data fitting much simpler because the individual regions of the surface do not have to "stretch" themselves out of shape to fit the data. Choosing the correct local topology, i.e., the polyhedron, to begin with can be difficult to do automatically, especially if the final model will be animated. Creating the model by hand is usually fairly simple but automating the process is difficult for many of the reasons that artificial intelligence has proven difficult.

A standard way to reduce a model is to perform triangle reduction [SZL92][HDD93]. If the desired effect is to reduce the *geometry* of the model, i.e., the number of triangles, then this is an appropriate method. If, however, the desire is to capture the basic structure or topology of the model then we need a technique that simplifies or reduces the topology and does not rely on triangles. For example, consider reducing a cylinder, which can be modeled using a single patch but which reduces at best to twelve or so triangles.

Appendix A

Projective Transformations

We construct a projective transformation that takes four non-collinear points in the plane to any four non-collinear points in the plane (see Figure 5.17).

A.1 Computing the transform

Let $B_1 = (1,0,0)^t, B_2 = (0,1,0)^t, B_3 = (0,0,1)^t$, and $B_4 = (1,1,1)^t$ and let $P_1 \ldots P_4$ be the (column vectors of) homogeneous coordinates of four points p_i in the plane, no three of which are collinear. Let M_P be the 3×3 matrix whose columns are P_1, P_2, and P_3. Note that M_P is invertible because the three vectors P_1, P_2, and P_3 are non-collinear and hence form a basis for $\Re^3$. Since P_4 is also an element of $\Re^3$, we can write P_4 as a linear combination of P_1, P_2, and P_3. Let

$$P_4 = \lambda_1 P_1 + \lambda_2 P_2 + \lambda_3 P_3 = M_P(\lambda_1 \lambda_2 \lambda_3)^t \tag{A.1}$$

Since P_4 is non-collinear with any pair of P_1, P_2, and P_3, all of the λ_i are nonzero. We can rewrite Equation A.1 in the form

$$\Lambda = M_P^{-1} P_4$$

where Λ is a column vector with entries λ_i.

Let T_P be the matrix whose columns are $\lambda_1 P_1, \lambda_2 P_2$, and $\lambda_3 P_3$; then $T_P(B_i)$ is a nonzero multiple of the vector P_i and $T_P^{-1} P_i$ is B_i. For example:

$$T_P(B_1) = T_P(1,0,0)^t = \lambda_1(P_1)$$

$$T_P(B_4) = T_P(1,1,1)^t = \sum_{i=1}^{3} \lambda_i(P_i) = P_4$$

To find a projective transformation on the plane taking any set of four points $\{p_i\}$, no three collinear, to any other such set $\{q_i\}$, compute the matrix $L = T_Q T_P^{-1}$. To show that LP_i is a nonzero multiple of Q_i, note that L first takes the point P_i to the point B_i, then maps B_i to a nonzero multiple of Q_i.

Multiplying L by the vector $(x, y, 1)^t$ gives a vector

$$(X(x,y), Y(x,y), W(x,y))$$

The projective transformation we seek is just

$$\zeta_{PQ}(x,y) \mapsto (\frac{X(x,y)}{W(x,y)}, \frac{Y(x,y)}{W(x,y)})$$

This transform is unique for each distinct pair of polygons and maps lines to lines (note that the parameterization of the lines is not, in general, preserved). If the edges of the polygon $\{p_i\}$ do not intersect themselves (and similarly for $\{q_i\}$) then the transformation restricted to the polygon $\{p_i\}$ is one-to-one, onto, and C^∞ continuous.

A.2 The discontinuities of the transform

This transformation is undefined when $W(x, y) = 0$, i.e., when the last element of $L(x, y, 1)^t$ is 0. If $(L_{2,0}, L_{2,1}, L_{2,2})$ is the last row of L, this becomes

$$L_{2,0}x + L_{2,1}y + L_{2,2} = 0$$

which is a line provided the last row of L is nonzero. If the last row of L was zero then

$$L = T_Q T_P^1$$
$$LT_P = T_Q T_P^{-1} = T_Q$$

The last row of $LT_P = T_Q$ would also be zero, implying that T_Q was not invertible (which it is). Therefore the discontinuity is a line.

Figure A.1 shows an example transform and its corresponding line of discontinuity. The function, in general, is of the form

$$\{\frac{L_{00}x + L_{01}y + L_{02}}{L_{20}x + L_{21}y + L_{22}}, \frac{L_{10}x + L_{11}y + L_{12}}{L_{20}x + L_{21}y + L_{22}}\}$$

which is C^∞ everywhere that the denominator is nonzero.

A.3 Inverting the transform

To invert the transform we invert the matrix $L = T_Q T_P^{-1}$:

$$L^{-1} = (T_Q T_P^{-1})^{-1} = T_P T_Q^{-1}$$

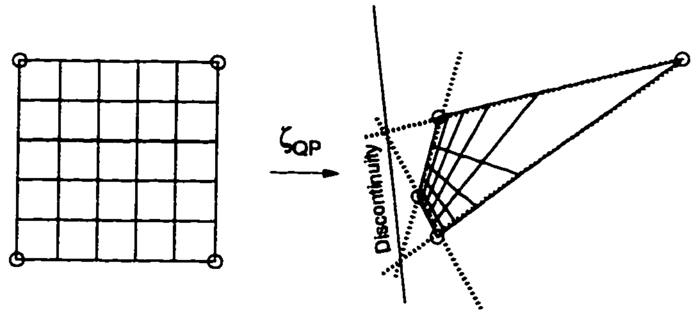


Figure A.1: A projective transform and its line of discontinuity.

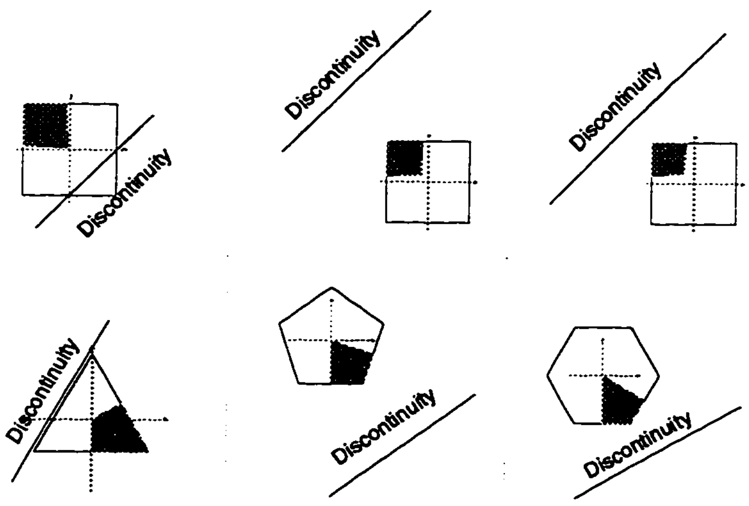


Figure A.2: The lines where the transition functions φ_{VF} (top) and φ_{FV} (bottom) are not defined. From left to right: The number of sides is three, five, and six. (For four-sided face charts the line is at infinity.)

A.4 The transition functions φ_{FV} and φ_{VF}

The transition function φ_{FV} is the restriction to U_{FV} of the projective transform taking a corner of a unit polygon to a quadrant of the unit square. The lines of discontinuity for this projective transform and its inverse are shown in Figure A.2.

Appendix B

The polyhedron

The user specifies the rough shape of the surface with a polyhedron, providing as much detail as desired. This sketch contains two separate types of information – topological and geometrical. The topological information is contained in the interrelations between the elements, e.g., which vertex is contained in which face. The geometrical information is contained in the locations of the vertices in 3-space.

Informally, the sketch is an oriented polyhedron with two restrictions placed on it; the faces meet four at a vertex, and the faces have three to six sides. The first restriction simplifies building a domain from the polyhedron without limiting possible topologies. (In fact, the dual of the first subdivision surface of any general polyhedron meets this restriction.) The second restriction is simply for implementation convenience.

We now provide a formal definition of the polyhedron. This definition is divided into two steps; we first define a *restricted complex*, which contains only topological information and then an abstract realization of the complex. The abstract realization allows us to immerse the restricted complex into three-space, thus adding geometrical information. This immersed object is what we call the *polyhedron*.

B.1 The restricted complex

This section defines a *cell complex* and gives the additional restrictions placed on the cell complex definition to produce a *restricted complex*. The cell complex definition given here is a simplification of that in. [CF67][1] A cell complex C is a triple $(\mathcal{V}, \mathcal{E}, \mathcal{F})$, where

1. $\mathcal{V}$ is a finite subset of $\mathcal{Z}$. An element of $\mathcal{V}$ is called a *vertex*. We denote the elements of $\mathcal{V}$ by the symbols $v_0 \dots v_{n-1}$.

[1] We are essentially defining an abstract two dimensional surface.

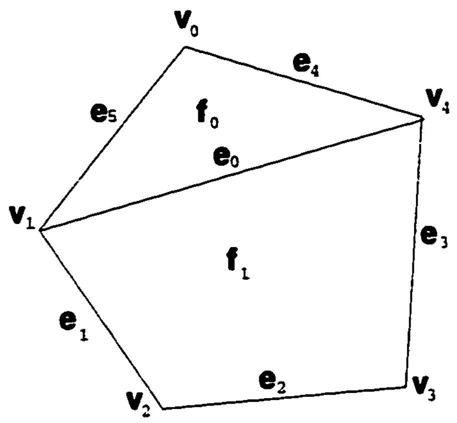


Figure B.1: A visualization of a complex $C = (\{v_0, \ldots, v_4\}, \{e_0, \ldots, e_5\}, \{f_0, f_1\})$ where $f_0 = [v_0, v_1, v_4]$ and $f_1 = [v_1, v_2, v_3, v_4]$.

2. $\mathcal{E}$ is a set of subsets of $\mathcal{V}$, each containing two distinct vertices. An element $e \in \mathcal{E}$ is called an *edge* and is written $e = \{v_j, v_k\}$.

3. $\mathcal{F}$ is a set of ordered subsets of $\mathcal{V}$, each containing three or more distinct vertices. An element $f \in \mathcal{F}$ is called a *face*. We write $f = [v_{i_1}, \ldots, v_{i_p}]$ to indicate the vertices of f and their order. If $f = [v_{i_1}, \ldots, v_{i_p}]$ then $\{v_{i_1}, v_{i_2}\}$, ..., $\{v_{i_{p-1}}, v_{i_p}\}$ and $\{v_{i_p}, v_{i_1}\}$ must all be in $\mathcal{E}$. These elements are the *edges* of f. If f is a face, let $\overline{f}$ be the unordered set with the same vertices as f.

4. $\mathcal{V} \subset \bigcup_{f \in \mathcal{F}} \overline{f}$ (i.e., every vertex is in some face).

5. $\forall e \in \mathcal{E}, \exists f \in \mathcal{F}$ such that e is an edge of f.

Additionally, we require that two faces that meet do so in a vertex or an edge, i.e.,

6. $\forall f_1, f_2 \in \mathcal{F}$, either

   (a) $\overline{f_1} \cap \overline{f_2} = \emptyset$

   (b) $\overline{f_1} \cap \overline{f_2}$ is a vertex of both f_1 and f_2.

   (c) $\overline{f_1} \cap \overline{f_2}$ is an edge of both f_1 and f_2.

Figure B.1 shows a visualization of a particular complex. Note that no geometry is associated with this definition: a complex is just a set of symbols and subsets of symbols.

The previous definition allows multiple faces per edge. An oriented 2D surface is a complex that also satisfies:

7. If $e = \{v_j, v_k\} \in \mathcal{E}$ then there is at most one face containing the sequence $\ldots, v_j, v_k, \ldots$ and at most one face containing the sequence $\ldots, v_k, v_j, \ldots$

A *boundary edge* of a complex is an edge that does not lie in the intersection of any two faces. A *boundary vertex* of a complex is a vertex that is contained in exactly two boundary edges.[2] If a vertex (or an edge) is not a boundary vertex (or edge) then it is an *interior* vertex (or edge).

The definition of a restricted complex is based on this definition with three additional restrictions:

8. If v is an interior vertex then there are exactly four distinct edges e and four distinct faces f such that $e \cap \{v\} \neq \emptyset$ and $\overline{f} \cap \{v\} \neq \emptyset$.
9. If v is a boundary vertex then there are one, two, or three distinct faces f (and two, three, or four distinct edges e) such that $e \cap \{v\} \neq \emptyset$ and $\overline{f} \cap \{v\} \neq \emptyset$.
10. Each face has three, four, five, or six edges.

B.2 Immersing the restricted complex in $\Re^3$

To immerse the restricted complex into 3-space we first need to define an *abstract realization* of the restricted complex. We then define a mapping from the abstract realization to $\Re^3$.

Definition 1 *An abstract realization of a complex $C = (\mathcal{V}, \mathcal{E}, \mathcal{F})$ is a set $E(C)$ of formal linear combinations of the symbols $v_0 \dots v_{n-1}$. The elements of the set are of three forms:*

1. *For each $v_i \in \mathcal{V}$ the combination $0v_0 + \dots + 1v_i + \dots 0v_{n-1} \in E(C)$.*
2. *For each $e = \{v_i, v_j\} \in \mathcal{E}$ the combination $0v_0 + \dots + \alpha_1 v_i + \alpha_2 v_j + \dots + 0v_{n-1} \in E(C)$ for every $\alpha_1, \alpha_2 \in \Re$ such that $\alpha_1 + \alpha_2 = 1$ and $0 \leq \alpha_1, \alpha_2 \leq 1$.*
3. *For each $f = [v_{i_1}, \dots, v_{i_m}] \in \mathcal{F}$ the combination $0v_0 + \dots + \alpha_1 v_{i_1} + \dots + \alpha_m v_{i_m} + \dots 0v_{n-1} \in E(C)$ for every $\alpha_i \in \Re$ where $\sum_1^m \alpha_i = 1$ and $0 \leq \alpha_i \leq 1$.*

We immerse this set $E(C)$ into $\Re^3$ by choosing locations for the vertices and then immersing the edges and faces on the basis of these choices:

1. Define a function $P : \mathcal{V} \to \Re^3$ which associates a point in $\Re^3$ to each vertex in C.
2. An element of the form $\sum_{j=0}^{n-1} \alpha_j v_j$ is taken to the point $\sum_{j=0}^{n-1} \alpha_j P(v_j)$.

Informally, the vertices go to points in 3-space, the edges go to straight lines between the vertices, and the faces to a "sheet" stretched between the edges. In practice, we triangulate the face using the centroid of the immersed vertices of the face and then immerse the triangles.

Note that there are no restrictions on the locations of the vertices.

[2] If a vertex is contained in any boundary edge, it must be contained in exactly two boundary edges.

We now have three different structures that are all related; the polyhedron (built by the user), the restricted complex (a geometry-free formalization of the topology of the polyhedron) and this abstract realization of the restricted complex. The latter is the link between the informal concept of a polyhedron and the abstract notation of the restricted complex. In this work we use the more informal terms *polyhedron*, *vertex*, *edge*, and *face* to refer to both the structure and geometry of the abstract realization of the restricted complex, disambiguating the two where necessary.

Appendix C

Theorems

C.1 $\sim$ is an equivalence relation

Theorem 3 *Let $K = (A, \Psi)$ be a proto manifold. Define a relation $\sim$ on $\sqcup_{c \in A} c$ such that if $x \in c_i$, $y \in c_j$, then $x \sim y$ iff $\psi_{ij}(x) = y$. Then $\sim$ is an equivalence relation.*

Proof:

1. $x \sim x$ for $x \in c$ since $x = \psi_{cc}(x)$.
2. Let $x \in c$, $y \in c'$. If

   $$x \sim y$$

   then

   $$\psi_{cc'}(x) = y$$

   take $\psi_{c'c}$ of both sides

   $$\begin{aligned} \psi_{c'c}(\psi_{cc'}(x)) &= \psi_{c'c}(y) \\ \psi_{cc'}^{-1}(\psi_{cc'}(x)) &= \psi_{c'c}(y) \\ x &= \psi_{c'c}(y) \\ \Rightarrow y &\sim x \end{aligned}$$

3. Let $x \in c_x$, $y \in c_y$, and $z \in c_z$. If $x \sim y$ and $y \sim z$ then

   $$\psi_{c_x c_y}(x) = y$$

$$\begin{aligned}
\psi_{c_y c_z}(\psi_{c_x c_y}(x)) &= \psi_{c_y c_z}(y) \\
(\psi_{c_y c_z} \circ \psi_{c_x c_y})(x) &= z \\
\psi_{c_x c_z}(x) &= z \\
\Rightarrow x &\sim z
\end{aligned}$$

□

C.2 M^K is a Hausdorff space

Theorem 4 *Let $K = (A, \Phi)$ and M^K be the quotient of $\sqcup_{(c\in A)} c$ by $\sim$. Then M^K is a Hausdorff space.*

Proof:

Recall that X is a Hausdorff space if:

- For every $p \in X$ there exists an open set containing p.
- For every $p, q \in X$, $p \neq q$, there exists an open set b_p containing p and an open set b_q containing q such that $b_p \bigcap b_q = \emptyset$.

Construct a ball b_p around $p \in M^K$ as follows. Let $\mathcal{B}$ be the set of charts c for which $\exists x \in c : \Pi_K(x) = p$. For each of these charts construct a neighborhood $b^c \in c$ around x (c is an open set in $\Re^n$ so b exists) and a corresponding set $b_c = \Pi_K(b^c)$ on the manifold. Recall that, by definition, the b_c are open sets. Let $b_p = \bigcap_{c\in\mathcal{B}} b_c$. Since b_p is a finite intersection of open sets, each of which contains p, b_p is an open set containing p.

Next we show that if $p, q \in M^K$, $p \neq q$, we can chose neighborhoods $b_p, b_q \subset M^K$ such that $p \in b_p$, $q \in b_q$, $b_p \bigcap b_q = \emptyset$.

If $\exists c : x, y \in c$, $\Pi_K(x) = p$, $\Pi_K(y) = q$, then we chose two neighborhoods $b_x, b_y \in c$ such that $b_x \bigcap b_y = \emptyset$. The corresponding subsets in M^K will also be disjoint, i.e., $\Pi_K(b_x) \bigcap \Pi_K(b_y) = \emptyset$ because the map α_c is injective.

If $\nexists c : x, y \in c$, $\Pi_K(x) = p$, $\Pi_K(y) = q$, then $\exists c_1, c_2 \in A$, $c_1 \neq c_2$ with $p = \Pi_K(x_p)$, $x_p \in c_1$, and $q = \Pi_K(x_q)$, $x_q \in c_2$. Recall that there exists an embedding $\mathcal{E}_{c_1c_2} : K_{cc'} \to \Re^n$. Let $e_p = \mathcal{E}_{c_1c_2}(x_p)$ and similarly for e_q. Construct two neighborhoods $\hat{b}_p \subset \mathcal{E}_{c_1c_2}(c_1)$ and $\hat{b}_q \subset \mathcal{E}_{c_1c_2}(c_2)$ around e_p and e_q such that $\hat{b}_p \bigcap \hat{b}_q = \emptyset$. Let $b_p = \hat{b}_p \bigcap \mathcal{E}_{c_1c_2}(c_1)$ and similarly for b_q (b_p and b_q exist and are both open sets because $\mathcal{E}_{c_1c_2}$ is an embedding and $\Re^n$ is Hausdorff). Then $\Pi_K(\mathcal{E}^{-1}_{c_1c_2}(b_p))$ and $\Pi_K(\mathcal{E}^{-1}_{c_1c_2}(b_q))$ are both open sets in M^K and their intersection is empty by construction. □

Item 4 of Definition 2, the requirement that the charts be pairwise embeddable, simplifies the last step of this proof. It is, however, a stronger requirement then necessary because it excludes some cases that *are* Hausdorff spaces. It is possible that a less stringent condition

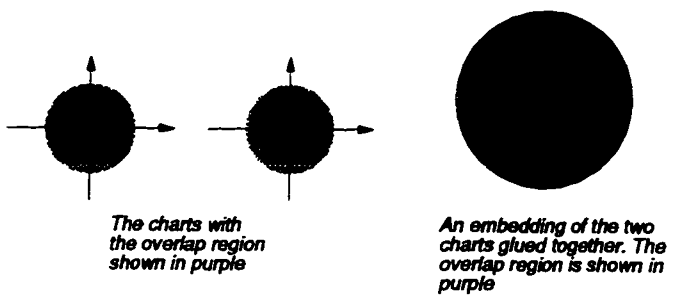

The charts with the overlap region shown in purple
An embedding of the two charts glued together. The overlap region is shown in purple

Figure C.1: An example embedding of a sphere built from two charts.

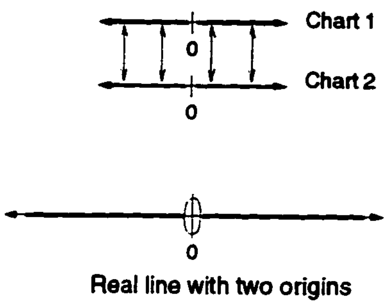


Figure C.2: The real line with two origins. The charts are glued together everywhere but at their origins.

would suffice; the following are some positive and negative examples that should be considered when trying to derive such a condition.

- Hausdorff space excluded by the condition: The atlas consists of two charts, both disks of radius 2. The overlap region for both is the annular region $r > 1$, where r is the radius. The transition function for both is $(x, y) \to (\frac{x}{x^2+y^2}, \frac{y}{x^2+y^2})$. This object is topologically a sphere (which is clearly a Hausdorff space). Figure C.1 shows an example embedding of this object.

- A non-Hausdorff space: The real line with two origins (see Figure C.2). The atlas consists of two charts, both consisting of the real line. The overlap region for both charts is $\Re - \{0\}$. The transition function is $x \to x$. To show this region is not a Hausdorff space, let p be the origin of the first chart, and q be the origin of the second chart. Then there is no neighborhood of p that does not also contain a part of the neighborhood of q.

- A non-Hausdorff space: A Y-shape (see Figure C.3). The atlas consists of two charts, both consisting of the real line. The overlap region for both charts is $x < 0$. The transition function is $x \to x$. To show this region is not a Hausdorff space, let p be the origin of the first chart and q be the origin of the second chart. Then there is no neighborhood of p that does not also contain some of the neighborhood of q.

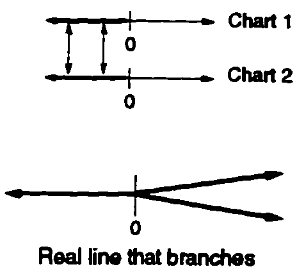


Figure C.3: A Y-shape formed by gluing the negative part of the charts together.

Note that if we take the overlap region to be $x \leq 0$ then the object is a Hausdorff space. but it is not locally $\Re$ at the origin.

Appendix D

φ_{EV} is one-to-one and invertible

In this appendix we prove that the edge-to-vertex function is one-to-one and onto and therefore invertible. The function is onto by definition; it remains to show that it is one-to-one. We first describe a general technique for proving that a function is one-to-one, given certain conditions, then show that the edge-to-vertex function meets these conditions.

D.1 Proving a function is one-to-one

We discuss the technique informally and then provide the theorem and its proof. Let $f : D \subset \Re^2 \rightarrow \Re^2$ be the function we want to prove is one-to-one, i.e., we want to show that if $(x, y), (x', y') \in D$ are two district points then $f(x, y) \neq f(x', y')$. Obviously, not every function is one-to-one. We require that f meet the following conditions:

- All of the partial derivatives of f with respect to x point in roughly the same direction.
- All of the partial derivatives of f with respect to y point in roughly the same direction.
- The set of x derivatives and their negatives do not overlap with the set of y derivatives and their negatives.

For example, let $X = \{\frac{\partial f}{\partial x}(x, y)\}_{(x,y \in D)}$ and similarly for Y. Suppose that X consists of vectors that point mostly to the right and Y consists of vectors that point mostly up (see Figure D.1). Let (x, y) be a point in the domain and let $(x + \Delta x, y)$ be a nearby point. If $\Delta x > 0$ then the approximate location of $f(x + \Delta x, y)$ is $f(x, y) + \Delta x \frac{\partial f}{\partial x}(x, y)$, which is a point to the right of $f(x, y)$. This is true at every point in D, since all of the x derivatives point to the right. Therefore, if we take any two points (x, y), $(x', y) \in D$, such that $x < x'$, then $f(x, y)_x < f(x', y)$. We can do the same analysis in the y direction; the points $f(x, y)$ and $f(x, y')$ are separated in the y direction.

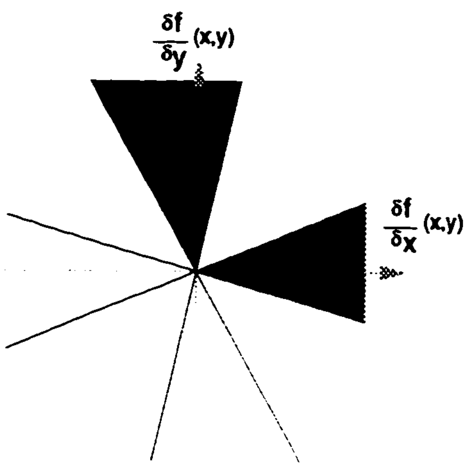

$\frac{\delta f}{\delta y}(x,y)$
$\frac{\delta f}{\delta x}(x,y)$

Figure D.1: The derivatives of f in the x direction all lie within the blue cone and the derivatives of f in the y direction all lie within the red cone.

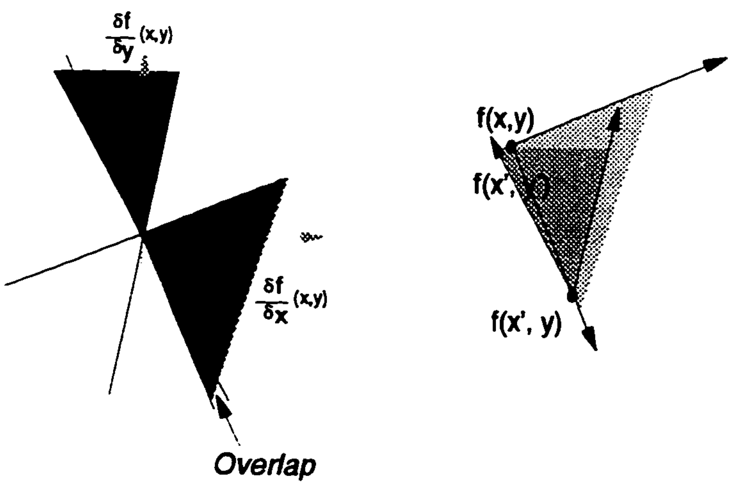

$\frac{\delta f}{\delta y}(x,y)$
$\frac{\delta f}{\delta x}(x,y)$
Overlap
f(x,y)
f(x',y')
f(x', y)

Figure D.2: The relative locations of $f(x, y)$, $f(x', y)$ and $f(x', y')$ where the x and y derivatives overlap. Note that $f(x, y) = f(x', y')$.

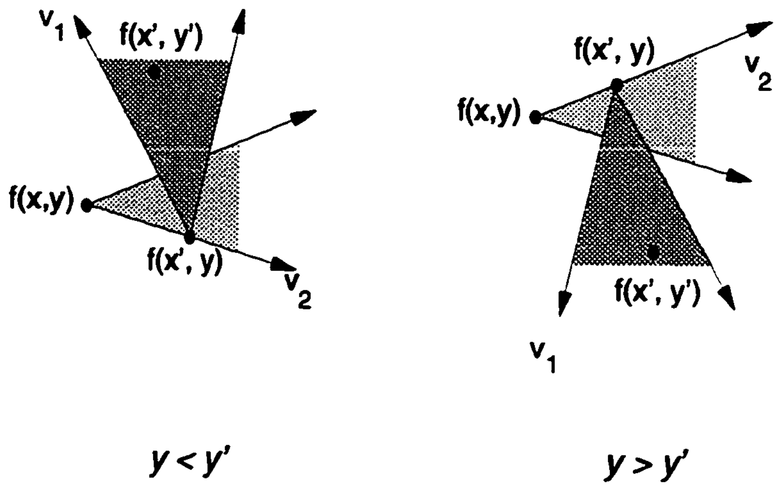


Figure D.3: The relative locations of $f(x,y)$, $f(x',y)$, and $f(x',y')$. The slope of v_1 is greater than the slope of v_2 because the x and y derivatives do not overlap.

Now let (x,y), $(x',y') \in D$ be two points such that $x < x'$ and $y < y'$. We know that the point $f(x',y)$ lies to the right of $f(x,y)$, and that $f(x',y')$ lies above $f(x',y)$. This alone does not, in general, guarantee that $f(x',y')$ lies above and to the right of $f(x,y)$ (see Figure D.2). If, however, the x and y derivatives meet the intersection condition given above we can guarantee that $f(x',y') \neq f(x,y)$.

More specifically, suppose each of the sets X and Y is contained within a cone, as shown in Figure D.1, and that these cones, when extended to the opposite direction, do not intersect. If this is true then we can bound the possible locations of $f(x',y')$ with respect to $f(x,y)$. We first note that

$$f(x',y) = f(x,y) + \int_x^{x'} \frac{\partial f}{\partial s}(s,y)ds$$

If $x < x'$ then the point $f(x',y)$ must lie somewhere to the right of $f(x,y)$, or in the cone containing X (see Figure D.3). To get from $f(x',y)$ to $f(x',y')$ we integrate from y to y' using an analogous formula. (If $y < y'$ then $f(x',y')$ lies somewhere in the Y cone placed at the point $f(x',y)$.) If $y' < y$ then we place the inverse cone at the point $f(x',y)$ instead. In either case, the absolute slope of the left edge of the second cone is greater than the slope of the top or bottom edge of the first cone; therefore the second cone cannot contain the point $f(x',y')$.

To see this more clearly, shear the cones so that one boundary of each of the cones lies along one of the axes (see Figure D.4). Each of the cones is now contained in a single quadrant; therefore the derivative in the x direction has a positive x-component (and a non-positive y-component). Similarly, the derivative in the y direction has a non-negative x-component. Therefore the image of the point (x',y') must lie to the right of the image of the point (x,y) (for $x < x'$).

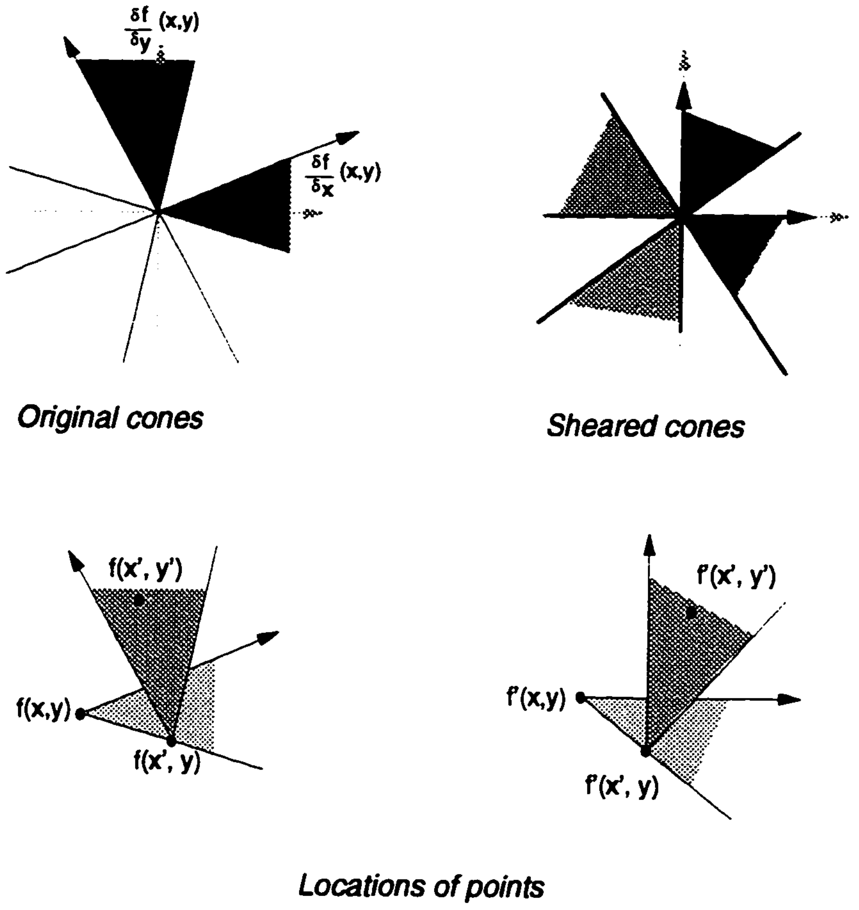


Figure D.4: Shearing the cones so that one boundary of each cone lies along an axis.

We formalize this idea in the following theorem.

Theorem 5 *Let D be a rectangle, $f : D \subset \Re^2 \to \Re^2$ be a C^1 function, and let*

$$\begin{aligned}
X_p &= \left\{\alpha\frac{\partial f}{\partial x}(x,y)\right\}_{(x,y)\in D,\alpha\in\Re^+} \\
Y_p &= \left\{\alpha\frac{\partial f}{\partial y}(x,y)\right\}_{(x,y)\in D,\alpha\in\Re^+} \\
X_n &= \left\{\alpha\frac{\partial f}{\partial x}(x,y)\right\}_{(x,y)\in D,\alpha\in\Re^-} \\
Y_n &= \left\{\alpha\frac{\partial f}{\partial y}(x,y)\right\}_{(x,y)\in D,\alpha\in\Re^-} \\
Y &= Y_p \cup Y_n \\
X &= X_p \cup X_n
\end{aligned}$$

If the following hold:

$$\begin{aligned}
X \cap Y &= \emptyset \\
X_p \cap X_n &= \emptyset \\
Y_p \cap X_n &= \emptyset
\end{aligned}$$

then f is one-to-one on the region D.

Proof: Observe that X_p, X_n, Y_p, and Y_n are all cones since f is C^1 and $(0,0)$ is not in X or Y. Let $T : \Re^2 \to \Re^2$ be the unique linear map taking the counter-clockwise boundary of X_p to the x-axis and the counter-clockwise boundary of Y_p to the y-axis. We show that $T \circ f$ is one-to-one, and since T is one-to-one and onto, then $T^{-1} \circ T \circ f = f$ must also be one-to-one and onto.

Let $(x,y),(x',y') \in D$ be two distinct points in D; without loss of generality let $x \leq x'$.
Case 1: If $x = x'$ then

$$(T\circ f)(x',y') = (T\circ f)(x,y') = (T\circ f)(x,y) + \int_y^{y'} \frac{\partial(T\circ f)}{\partial t}(x,t)dt$$

Since $(\frac{\partial(T\circ f)}{\partial y}(x,y))_y > 0$ for all points in D the images of the two points must differ in their y-coordinates:

$$(T \circ f)(x, y)_y < (T \circ f)(x', y')_y \quad \text{if } y < y'$$
$$(T \circ f)(x, y)_y > (T \circ f)(x', y')_y \quad \text{if } y > y'$$

Case 2: If $x < x'$ then

$$\begin{aligned}(T \circ f)(x', y) &= (T \circ f)(x, y) + \int_x^{x'} \frac{\partial (T \circ f)}{\partial s}(s, y)ds \\ (T \circ f)(x', y') &= (T \circ f)(x', y) + \int_y^{y'} \frac{\partial (T \circ f)}{\partial t}(x, t)dt\end{aligned}$$

Since $(\frac{\partial (T \circ f)}{\partial y}(x, y))_x \geq 0$ and $(\frac{\partial T \circ f}{\partial x}(x, y))_x > 0$ for all points in D the x the images of the two points must differ in their x-coordinates:

$$(T \circ f)(x, y)_x < (T \circ f)(x', y)_x \leq (T \circ f)(x', y')_x$$

Therefore $(T \circ f)$ is one-to-one. □

D.2 Proving φ_{EV} is one-to-one

In this section we show that the edge-to-vertex function meets the conditions given in Theorem 5. The steps of the proof are as follows:

- Partition the domain U_{EV} of φ_{EV} into three sections D_u, D_b, and D_l as follows: let D_b be the blend region in the middle of U_{EV} and let D_u and D_l be the two non-blended regions adjacent to it (see Figure 5.18).
- Establish that the images of the three sections do not intersect, e.g., $\varphi_{EV}(D_u) \bigcap \varphi_{EV}(D_b) = \emptyset$.
- Show that φ_{EV} is one-to-one on each of D_u, D_b, and D_l. This is trivially true for D_u and D_l; for D_b we apply Theorem 5.

For the following discussion let U_{EV} be the right half of an edge chart and U_{VE} be the left half of the vertex chart (i.e., the configuration shown in Figure 5.18). It suffices to prove invertibility for this case since the other cases are simply rotations and translations of this one.

Recall that there are four different φ_u functions (the overlapping upper face has three, four, five, or six sides) and four different φ_l functions, for a total of sixteen different φ_{EV} functions. The discussion here refers to any one of those sixteen functions. The specific numerical bounds generated for the different cases may be different, but the technique for generating them is the same.

D.2.1 The domain

The function is clearly one-to-one on the non-blended regions, since it is the composition of two projective transforms. We therefore restrict the discussion to the blend region between them:

$$D_b = \{(x,y) \in U_{EV} : -h\cot(\frac{\pi}{n_l}) \le y \le h\cot(\frac{\pi}{n_u})\}$$

To simplify the proof we use the slightly larger domain

$$D_{EV} = [0, 0.5] \times [-m(h), m(h)] \tag{D.1}$$

where $m(h)$ is the smallest number such that $D_b \subset D_{EV}$:

$$\begin{aligned} m(h) &= \max(h\cot(\frac{\pi}{3}), h\cot(\frac{\pi}{4}), h\cot(\frac{\pi}{5}), h\cot(\frac{\pi}{6})) \\ &= \max(0.57735\,h, 1.\,h, 1.37638\,h, 1.73205\,h) \\ &= h\cot(\frac{\pi}{6}) \end{aligned}$$

The function φ_{EV} is not defined on all of D_{EV}. Recall that the definition of φ_{EV} is

$$\varphi_{EV}(x,y) = (1-\beta(y))\varphi_{F_lV} \circ \varphi_{EF_l} + \beta(y)\varphi_{F_uV} \circ \varphi_{EF_u}$$

Clearly, we can extend the definition of φ_{EV} to the region D_{EV}. If the extended definition is one-to-one then φ_{EV} is also one-to-one. In the following proof we will show that this extended definition is one-to-one on D_{EV}; to simplify the notation, however, we will use φ_{EV} to refer to the extened definition.

D.2.2 The image of the blended region

We show that $\varphi_{EV}(D_b)$, the image of the blended region, does not intersect $\varphi_{EV}(D_u)$ or $\varphi_{EV}(D_l)$, the images of the non-blended regions. The region D_b is contained within the four lines

$$\begin{array}{ccc} & y = h\cot(\frac{\pi}{n_u}) & \\ x = 0 & & x = 0.5 \\ & y = h\cot(\frac{\pi}{n_l}) & \end{array}$$

The left boundary, $x = 0$, is mapped to the line $x = -0.5$ in the vertex chart. The upper (and lower) boundaries are each mapped to a line in the vertex chart (see Figure D.5). If the function is one-to-one on D_{EV} then $\varphi_{EV}(D_b)$ must lie between the two lines in the vertex chart. Since $\varphi_{EV}(D_u)$ and $\varphi_{EV}(D_l)$ lie entirely either above or below these lines the three regions do not intersect.

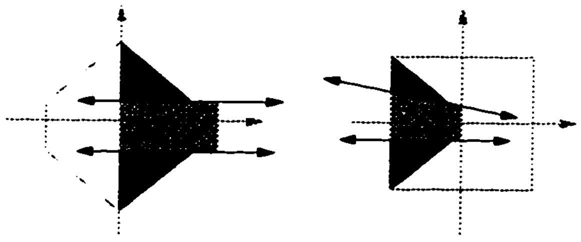


Figure D.5: Bounds on the image of the blended region.

D.2.3 Abbreviations and definitions

We abbreviate the upper and lower composed functions from which φ_{EV} is defined as follows:

$$\begin{aligned}\varphi_u &= \varphi_{F_uV} \circ \varphi_{EF_u}\\ \varphi_l &= \varphi_{F_lV} \circ \varphi_{EF_l}\end{aligned}$$

Let $f : D \subset \Re^2 \to \Re^2$ be a C^1 function and let

$$X_f = \{\frac{\partial f}{\partial x}(x,y)\}_{\alpha \in \Re^+,(x,y)\in D}$$
$$Y_f = \{\frac{\partial f}{\partial y}(x,y)\}_{\alpha \in \Re^+,(x,y)\in D}$$

If the x and y derivatives of f are never the zero vector then X_f and Y_f are cones. We define the *extremes* of the x derivatives of f to be those vectors that lie on either boundary of the cone X_f, and similarly for the y derivatives. We note that the slope of the line containing the vector $\{v_x, v_y\}$ is v_y/v_x; to find the vectors lying on the boundary of the cone it suffices to compare the ratios v_y/v_x of the derivative vectors. If the cone is contained within a half plane then the smallest and largest ratios are the slopes of the boundaries of the cone.

In summary, to find the extremes of the x derivatives of f we must show the following:

- For all $(x,y) \in D$, $\frac{\partial f}{\partial x}(x,y) \neq \{0,0\}$.
- X_f is contained in a half plane.
- Let r_l and r_s be the slopes of the left and right boundaries of the cone X_f. Then r_l and r_s must be the largest and smallest possible ratios v_y/v_x, where v is an x derivative vector.

Finding the extremes of the y derivatives of f is a similar process.

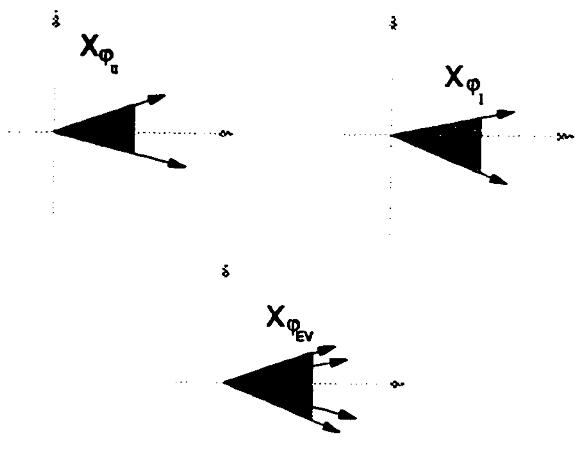


Figure D.6: The relationships between the x derivatives of φ_u, φ_l, and φ_{EV}.

D.2.4 φ_{EV} is one-to-one on D_{EV}

We first bound the derivative in the x direction, then the derivative in the y direction. We then show that these bounded sets of derivatives do no intersect.

Bounding the derivative of φ_{EV} in the x direction

The derivatives of φ_{EV} in the x direction are a blend of the derivatives in the x direction of the two composed functions:

$$\frac{\partial \varphi_{EV}}{\partial x}(x,y) = (1-\beta(y))\frac{\partial \varphi_l}{\partial x}(x,y) + \beta(y)\frac{\partial \varphi_u}{\partial x}(x,y)$$

Since the x derivative of φ_{EV} is a convex combination of the x derivatives of φ_u and φ_l the extremes of its x derivative must lie between the extremes of the x derivatives of φ_u and φ_l (see Figure D.6). It suffices to find the x derivative extremes of both φ_u and φ_l and take the greater. To find the extremes we use the technique described in Section D.2.3. Both φ_u and φ_l are projective transforms with x derivatives of the form:

$$\{\frac{a_0 + a_1 y}{(b_0 + b_1 x + b_2 y)^2}, \frac{\pm a_1 y}{(b_0 + b_1 x + b_2 y)^2}\}$$

where

$a_0 = 18$	$a_1 = 12\sqrt{3}$	if $n = 3$
$a_0 = 1$	$a_1 = 0$	if $n = 4$
$a_0 = .763932$	$a_1 = 0.496433$	if $n = 5$
$a_0 = 6$	$a_1 = 4\sqrt{3}$	if $n = 6$

The functions φ_u and φ_l are both C^1 on D_{EV}. The derivative is never the zero vector because a_0 is non-zero. The x-component of the derivative is positive provided $a_0+a_1y > 0$, or $a_0 > -a_1y$. Since $|y| < m(h) < 1$ and $a_1 < a_0$ this is clearly true and therefore the cones X_{f_u} and X_{f_l} exist and lie in the $x > 0$ half plane. The extremes are the points at which the ratio $(a_1y)/(a_0+a_1y)$ is biggest and smallest. Since (a_0+a_1y) is always positive and $-m(h) < y < m(h)$ the extreme values occur at $y = \pm m(h)$.

A note: in the 4-sided case the derivative is everywhere $\{1, 0\}$

Bounding the derivative of φ_{EV} in the y direction

The derivative of φ_{EV} in the y direction is a blend of the derivatives in the y direction of the two composed functions plus the difference in value of the two functions, scaled by β':

$$\begin{aligned}
df_{ul}(x,y) &= (1-\beta(y))\frac{\partial\varphi_l}{\partial y} + \beta(y)\frac{\partial\varphi_u}{\partial y} \\
f_{ul}(x,y) &= \frac{\partial\beta}{\partial y}(y)(\varphi_u - \varphi_l)(x,y) \\
\frac{\partial\varphi_{EV}}{\partial y}(x,y) &= df_{ul}(x,y) + f_{ul}(x,y)
\end{aligned}$$

We bound this derivative by bounding df_{ul} and f_{ul} individually. Bounding df_{ul} is accomplished by finding the extremes of the y derivatives of φ_u and φ_l. f_{ul} is bounded by extremizing the derivative of β.

Since df_{ul} is a convex combination of the y derivatives of φ_u and φ_l the extremes of its y derivative must lie between the extremes of the y derivatives of φ_u and φ_l. It suffices to find the y derivative extremes of both φ_u and φ_l and take the greater. To find the extremes we apply the technique in Section D.2.3. The y derivatives of φ_u and φ_l are of the form:

$$\left\{\frac{\pm a_1x}{(b_0+b_1x+b_2y)^2}, \frac{a_0+a_1x}{(b_0+b_1x+b_2y)^2}\right\}$$

where

$$\begin{array}{lll}
a_0 = 6\sqrt{3} & a_1 = 12\sqrt{3} & \text{if } n=3 \\
a_0 = 1 & a_1 = 0 & \text{if } n=4 \\
a_0 = 4.71693 & a_1 = 2.22703 & \text{if } n=5 \\
a_0 = 6\sqrt{3} & a_1 = 4\sqrt{3} & \text{if } n=6
\end{array}$$

The functions φ_u and φ_l are both C^1 on D_{EV}. The derivative is never the zero vector because a_0 is non-zero. The y-component of the derivative is positive provided $a_0+a_1x > 0$, or $a_0 > -a_1x$. Since $x < 0.5$ and $a_0 < a_1$ this is clearly true and therefore the cones Y_{f_u} and Y_{f_l} exist and lie in the $y > 0$ half plane. The extremes are the points at which the ratio $(a_1x)/(a_0+a_1x)$ is biggest and smallest. Since (a_0+a_1x) is always positive and $0 < x < .5$ the extreme values occur at $x = 0$ and $x = .5$.

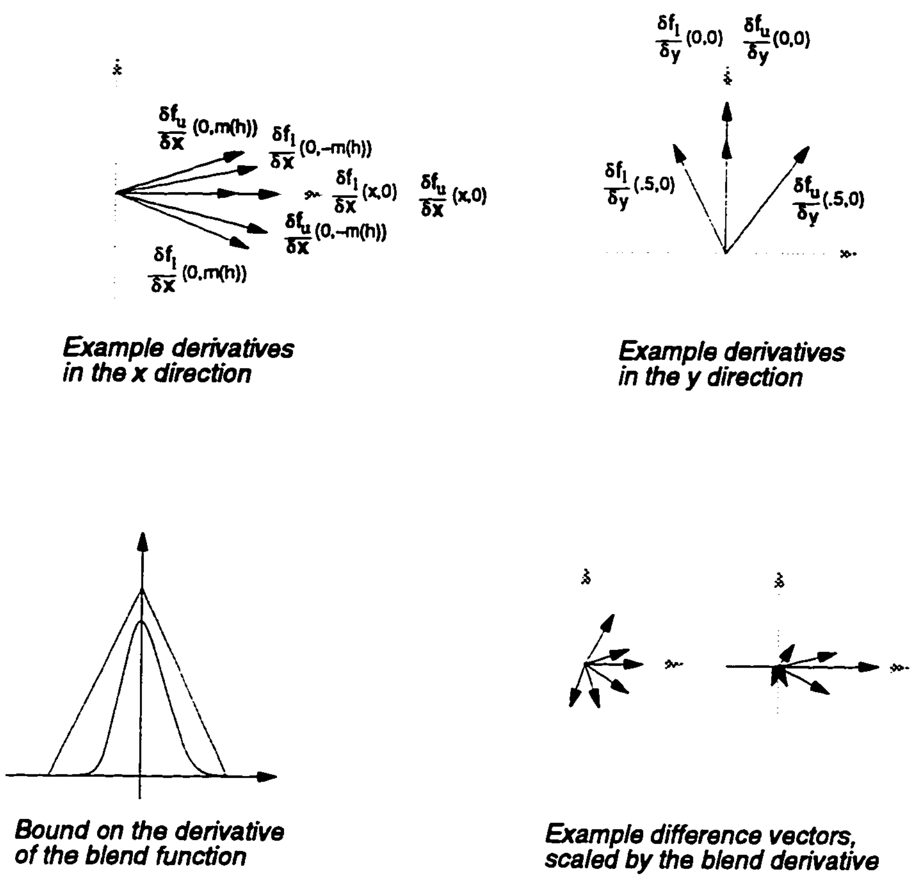

δf_u/δx (0,m(h))
δf_l/δx (0,-m(h))
δf_l/δx (x,0)
δf_u/δx (x,0)
δf_u/δx (0,-m(h))
δf_l/δx (0,m(h))
Example derivatives in the x direction
δf_l/δy (0,0)
δf_u/δy (0,0)
δf_l/δy (.5,0)
δf_u/δy (.5,0)
Example derivatives in the y direction
Bound on the derivative of the blend function
Example difference vectors, scaled by the blend derivative

Figure D.7: The general locations of the derivatives making up the derivatives of φ_{EV}.

In all cases, the y derivative at the points $(0, y)$ is of the form $\{0, t\}$. In the 4-sided case the y derivative is everywhere $\{0, 1\}$.

The function f_{ul} can be bounded by bounding the derivative of β. Recall that the one of the conditions on the blend function is that it be bounded (Equation 5.2):

$$f_{ul}(x,y) \leq \begin{cases} (\varphi_u(x,y) - \varphi_l(x,y))\frac{m(h)+y}{m(h)} & \text{if } y \leq 0 \\ (\varphi_u(x,y) - \varphi_l(x,y))\frac{m(h)-y}{m(h)} & \text{if } y > 0 \end{cases}$$

Summary

Figure D.7 summarizes the locations of the various derivatives.

The extremes of the x derivative are two of the following four vectors, each of which is a function of $m(h)$:

$$\frac{\partial \varphi_u}{\partial x}(0, \pm m(h)), \quad \frac{\partial \varphi_l}{\partial x}(0, \pm m(h))$$

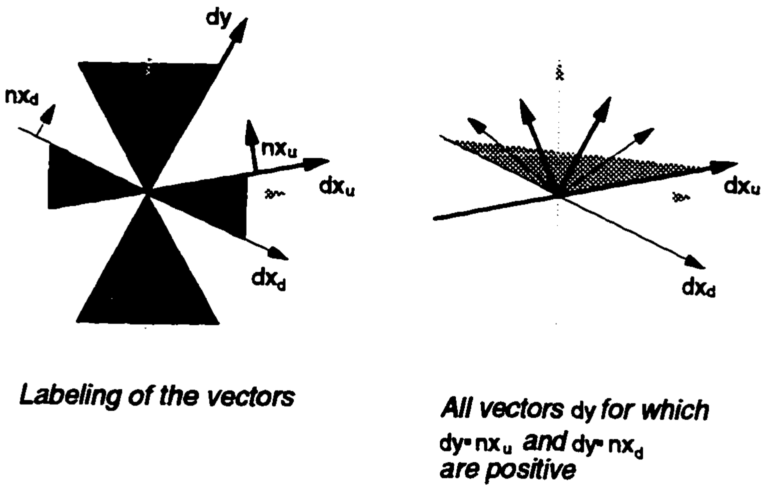


Figure D.8: Showing that the two cones do not intersect.

Which two of the four vectors are the extremes depends only upon the number of sides of f_u and f_l; it is independent of the actual value of h. If $\frac{\partial\varphi_u}{\partial x}(0, m(h))$ is an extreme for $h = h_0$ it is also an extreme for $h = h_1$.

The extremes of the y derivative are the sum of the extremes of df_{ul} and f_{ul}. The extremes of df_{ul} are two of the following four vectors, each of which is constant:

$$\frac{\partial\varphi_u}{\partial y}(0,0),\quad \frac{\partial\varphi_l}{\partial y}(0,0),\quad \frac{\partial\varphi_u}{\partial y}(0.5,0),\quad \frac{\partial\varphi_l}{\partial y}(0.5,0)$$

Which two of the four vectors are the extremes depends upon the number of sides of f_u and f_l.

The extremes of f_{ul} are split into two cases, each of which is a function of the three variables s, t, and $m(h)$.

$$\begin{array}{ll} (\varphi_u(x,y) - \varphi_l(x,y))\frac{m(h)+y}{m(h)} & \text{if } y \leq 0 \\ (\varphi_u(x,y) - \varphi_l(x,y))\frac{m(h)-y}{m(h)} & \text{if } y > 0 \end{array}$$

D.2.5 Proving the cones do not intersect

To show that the cones do not intersect we show that the angular distance from the boundary of one cone to the boundary of the other is positive. This analysis breaks up into eight cases: both boundaries of the x derivative cone, both boundaries of the y derivative cone, $t \leq 0$, and $t > 0$.

We measure the angular distance between two vectors by taking the dot product of one vector with the normal to the other vector (see Figure D.8).

Note that the gap h is at most $\frac{0.0416666}{4/3+0}$ (see Section 8.1); therefore $m(h) \leq 0.03125 \cot \pi/6 < 0.0541266$). Let

$$\begin{aligned}
dx_1(m) &: [0, 0.0541266] \rightarrow \Re^2 \\
dx_2(m) &: [0, 0.0541266] \rightarrow \Re^2
\end{aligned}$$

be the two extremes of the x derivative and let

$$\begin{aligned}
dyp_1(x,y,m) &: [0,0.5] \times [0,m] \times [0, 0.0541266] \rightarrow \Re^2 \\
dyp_2(x,y,m) &: [0,0.5] \times [0,m] \times [0, 0.0541266] \rightarrow \Re^2 \\
dyn_1(x,y,m) &: [0,0.5] \times [0,-m] \times [0, 0.0541266] \rightarrow \Re^2 \\
dyn_1(x,y,m) &: [0,0.5] \times [0,-m] \times [0, 0.0541266] \rightarrow \Re^2
\end{aligned}$$

be the four extremes of the y derivative. Let

$$\begin{aligned}
nx_1(m) &= (-dx_1(m)_y, dx_1(m)_x) \\
nx_2(m) &= (-dx_2(m)_y, dx_2(m)_x)
\end{aligned}$$

be the normals to the boundaries of the x derivative cone. If the dot product of each of the two normals with the four y derivative extremes is positive, e.g.,

$$dist(x,y,m) = nx_1(m) \cdot dyp_1(x,y,m) > 0$$

for all eight variations then the two cones do not intersect. (Note that we do not have to normalize these vectors since we are only interested in their relative orientation.) The function *dist* is a rational polynomial in three variables. To prove that it is positive on the domain we show that the numerator and denominator have the same sign. To prove that the numerator (or denominator) is of a particular sign we gather terms of the polynomial and show that groups of coefficients are of the correct sign. For example, if all of the coefficients of m are positive then the m terms must be positive, since m is positive.

D.2.6 A specific example

We now trace through the calculations for a particular example. These equations were generated using Mathematica [Wol91]. Refer to Figure D.7.

Let the upper face have three sides and the lower face have six sides. Then φ_u is

$$\varphi_u(x,y) = \{\frac{3\left(-0.5 + x + \frac{y}{\sqrt{3}}\right)}{3 + 6x - 2\sqrt{3}\,y}, \frac{-2\sqrt{3}\,y}{-3 - 6x + 2\sqrt{3}\,y}\}$$

and φ_l is

$$\varphi_l(x,y) = \{\frac{3\left(0.5 - x - \frac{y}{\sqrt{3}}\right)}{-3+2x+2\sqrt{3}\,y}, \frac{-2\sqrt{3}\,y}{-3+2x+2\sqrt{3}\,y}\}$$

The x derivatives of f_u and f_l:

$$\frac{\partial\varphi_u}{\partial x}(x,y) = \{\frac{12\,(1.5-\sqrt{3}y)}{(3+6x-2\sqrt{3}y)^2}, \frac{-12\sqrt{3}\,y}{(3+6x-2\sqrt{3}y)^2}\}$$
$$\frac{\partial\varphi_l}{\partial x}(x,y) = \{\frac{4\,(1.5-\sqrt{3}y)}{(-3+2x+2\sqrt{3}y)^2}, \frac{4\sqrt{3}\,y}{(-3+2x+2\sqrt{3}y)^2}\}$$

which are extremized at $(0, \pm m(h))$. Recall that the extremes are found by finding the vectors with the largest and smallest slope. To determine which two of the four vectors are the extremes we evaluate the derivative vectors at $(0, \pm 0.0541266)$, the largest and smallest possible values of $m(h)$, and see which of the four are extremes:

$$\begin{aligned}
\frac{\partial\varphi_u}{\partial x}(0,-0.0541266) &= \{1.88235, 0.110727\}\\
\frac{\partial\varphi_u}{\partial x}(0,0.0541266) &= \{2.13333, -0.142222\}\\
\frac{\partial\varphi_l}{\partial x}(0,-0.0541266) &= \{0.627451, -0.0369089\}\\
\frac{\partial\varphi_l}{\partial x}(0,0.0541266) &= \{0.711111, 0.0474074\}
\end{aligned}$$

Comparing the two vectors with positive slopes:

$$\arctan(\tfrac{\partial\varphi_u}{\partial x}(0,-0.0541266)) = 0.0587558 \;>\; 0.0665682 = \arctan(\tfrac{\partial\varphi_l}{\partial x}(0,0.0541266))$$

Comparing the two vectors with negative slopes:

$$\arctan(\tfrac{\partial\varphi_l}{\partial x}(0,-0.0541266)) = -0.0587558 \;<\; -0.0665682 = \arctan(\tfrac{\partial\varphi_u}{\partial x}(0,0.0541266))$$

The two extremes are therefore

$$dx_u(m) = \frac{\partial\varphi_u}{\partial x}(0,-m) = \{\frac{12\,(1.5+\sqrt{3}m)}{(3+2\sqrt{3}m)^2}, \frac{12\sqrt{3}m}{(3+2\sqrt{3}m)^2}\}$$

$$dx_l(m) = \frac{\partial\varphi_l}{\partial x}(0,-m) = \{\frac{4\,(1.5+\sqrt{3}m)}{(3+2\sqrt{3}m)^2}, \frac{-4\sqrt{3}m}{(3+2\sqrt{3}m)^2}\}$$

The derivatives in the y direction of φ_u and φ_l are

$$\frac{\partial\varphi_u}{\partial y}(x,y) = \{\frac{12\sqrt{3}\,x}{(-3-6\,x+2\sqrt{3}\,y)^2}, \frac{6\,(\sqrt{3}+2\sqrt{3}\,x)}{(-3-6\,x+2\sqrt{3}\,y)^2}\}$$

and

$$\frac{\partial\varphi_l}{\partial y}(x,y) = \{\frac{4\sqrt{3}\,x}{(-3+2\,x+2\sqrt{3}\,y)^2}, \frac{2\,(3\sqrt{3}-2\sqrt{3}\,x)}{(-3+2\,x+2\sqrt{3}\,y)^2}\}$$

which are extremized at $(0,0)$ and $(0.5,0)$. The vectors at $(0,0)$ both point straight up, while the other two vectors both have a positive x component.

$$\begin{aligned} dy_r &= \frac{\partial\varphi_u}{\partial y}(0.5,0) = \{0.288675, 0.57735\} \\ \arctan(dy_r) &= 1.10715 \end{aligned}$$

$$\begin{aligned} dy_l &= \frac{\partial\varphi_l}{\partial y}(0.5,0) = \{0.866025, 1.73205\} \\ \arctan(dy_l) &= 1.10715 \end{aligned}$$

Therefore, the left and right extreme of df_{ul} occur at

$$\begin{aligned} \frac{\partial\varphi_l}{\partial y}(0.5,0) &= \{0.866025, 1.73205\} \\ \frac{\partial\varphi_l}{\partial y}(0,0) &= \{0., 1.1547\} \end{aligned}$$

The equation f_{ul} for $t > 0$ is:

$$\begin{aligned} f_{ul}(x,y,m) &< \frac{m-y}{m}(\varphi_u(x,y) - \varphi_l(x,y)) \\ &= \frac{m-t}{m}\{\frac{24\left(0.5\,x - x^2 - \frac{xy}{\sqrt{3}}\right)}{(-3-6\,x+2\sqrt{3}y)\,(-3+2\,x+2\sqrt{3}\,y)}, \frac{16\sqrt{3}\,x\,y}{(3+6\,x-2\sqrt{3}y)\,(-3+2\,x+2\sqrt{3}\,y)}\} \\ &= dy_p(x,y,m) \end{aligned}$$

and for $t \leq 0$ is:

$$\begin{aligned} f_{ul}(x,y,m) &< \frac{m+y}{m}(\varphi_u(x,y) - \varphi_l(x,y)) \\ &= \frac{m+t}{m}\{\frac{24\left(0.5\,x - x^2 - \frac{xy}{\sqrt{3}}\right)}{(-3-6\,x+2\sqrt{3}y)\,(-3+2\,x+2\sqrt{3}\,y)}, \frac{16\sqrt{3}\,x\,y}{(3+6\,x-2\sqrt{3}y)\,(-3+2\,x+2\sqrt{3}\,y)}\} \\ &= dy_n(x.y,m) \end{aligned}$$

These are the normals to the boundary of the x derivative cone.

$$\begin{aligned} nx_u(m) &= \{-(dx_u(m))_y, (dx_u(m))_x\} \\ nx_l(m) &= \{-(dx_l(m))_y, (dx_l(m))_x\} \end{aligned}$$

We must show that the following eight functions are positive on one of the regions $[0, 0.5] \times [0, \pm m] \times [0, 0.0541266]$. To fit the equations on the paper the numbers have been rounded to the third decimal place.

$$\begin{aligned}
nx_u(m) \cdot (dy_r + dy_p(s,t,h)) &= \frac{12.\ (1.5 + 1.73\, m) \left(0.612 + \frac{27.7 x\, (m-1.\, y)\, y}{m^2\, (3.+6.\, x-3.46 y)\, (-3.+2.\, x+3.46 y)}\right)}{(3. + 3.46\, m)^2} + \\
&\quad \frac{-20.8\, m \left(0.306 + \frac{24.\, (m-1.\, y)\, (0.5 x-1.\, x^2-0.577 x\, y)}{m^2\, (-3.-6.\, x+3.46 y)\, (-3.+2.\, x+3.46 y)}\right)}{(3. + 3.46\, m)^2} \\
nx_u(m) \cdot (dy_r + dy_n(s,t,h)) &= \frac{12.\ (1.5 + 1.73\, m) \left(0.612 + \frac{27.7 x\, y\, (m+y)}{m^2\, (3.+6.\, x-3.46 y)\, (-3.+2.\, x+3.46 y)}\right)}{(3. + 3.46\, m)^2} + \\
&\quad \frac{-20.8\, m \left(0.306 + \frac{24.\, (m+y)\, (0.5 x-1.\, x^2-0.577 x\, y)}{m^2\, (-3.-6.\, x+3.46 y)\, (-3.+2.\, x+3.46 y)}\right)}{(3. + 3.46\, m)^2} \\
nx_u(m) \cdot (dy_l + dy_p(s,t,h)) &= \frac{12.\ (1.5 + 1.73\, m) \left(2.08 + \frac{27.7 x\, (m-1.\, y)\, y}{m^2\, (3.+6.\, x-3.46 y)\, (-3.+2.\, x+3.46 y)}\right)}{(3. + 3.46\, m)^2} + \\
&\quad \frac{-20.8\, m \left(1.04 + \frac{24.\, (m-1.\, y)\, (0.5 x-1.\, x^2-0.577 x\, y)}{m^2\, (-3.-6.\, x+3.46 y)\, (-3.+2.\, x+3.46 y)}\right)}{(3. + 3.46\, m)^2} \\
nx_u(m) \cdot (dy_l + dy_n(s,t,h)) &= \frac{12.\ (1.5 + 1.73\, m) \left(2.08 + \frac{27.7 x\, y\, (m+y)}{m^2\, (3.+6.\, x-3.46 y)\, (-3.+2.\, x+3.46 y)}\right)}{(3. + 3.46\, m)^2} + \\
&\quad \frac{-20.8\, m \left(1.04 + \frac{24.\, (m+y)\, (0.5 x-1.\, x^2-0.577 x\, y)}{m^2\, (-3.-6.\, x+3.46 y)\, (-3.+2.\, x+3.46 y)}\right)}{(3. + 3.46\, m)^2} \\
nx_l(m) \cdot (dy_r + dy_p(s,t,h)) &= \frac{12.\ (1.5 - 1.73\, m) \left(0.612 + \frac{27.7 x\, (m-1.\, y)\, y}{m^2\, (3.+6.\, x-3.46 y)\, (-3.+2.\, x+3.46 y)}\right)}{(-3. + 3.46\, m)^2} + \\
&\quad \frac{2.31\, m \left(0.306 + \frac{24.\, (m-1.\, y)\, (0.5 x-1.\, x^2-0.577 x\, y)}{m^2\, (-3.-6.\, x+3.46 y)\, (-3.+2.\, x+3.46 y)}\right)}{(1. - 1.15\, m)^2} \\
nx_l(m) \cdot (dy_r + dy_n(s,t,h)) &= \frac{12.\ (1.5 - 1.73\, m) \left(0.612 + \frac{27.7 x\, y\, (m+y)}{m^2\, (3.+6.\, x-3.46 y)\, (-3.+2.\, x+3.46 y)}\right)}{(-3. + 3.46\, m)^2} +
\end{aligned}$$

$$\frac{2.31\,m\left(0.306+\frac{24.\,(m+y)\left(0.5x-1.\,x^2-0.577x\,y\right)}{m^2\,(-3.-6.\,x+3.46y)\,(-3.+2.\,x+3.46y)}\right)}{(1.-1.15\,m)^2}$$

$$nx_l(m)\cdot(dy_l+dy_p(s,t,h)) = \frac{12.\,(1.5-1.73\,m)\left(2.08+\frac{27.7x\,(m-1.\,y)\,y}{m^2\,(3.+6.\,x-3.46y)\,(-3.+2.\,x+3.46y)}\right)}{(-3.+3.46\,m)^2}+\frac{2.31\,m\left(1.04+\frac{24.\,(m-1.\,y)\left(0.5x-1.\,x^2-0.577x\,y\right)}{m^2\,(-3.-6.\,x+3.46y)\,(-3.+2.\,x+3.46y)}\right)}{(1.-1.15\,m)^2}$$

$$nx_l(m)\cdot(dy_l+dy_n(s,t,h)) = \frac{12.\,(1.5-1.73\,m)\left(2.08+\frac{27.7x\,y\,(m+y)}{m^2\,(3.+6.\,x-3.46y)\,(-3.+2.\,x+3.46y)}\right)}{(-3.+3.46\,m)^2}+\frac{2.31\,m\left(1.04+\frac{24.\,(m+y)\left(0.5x-1.\,x^2-0.577x\,y\right)}{m^2\,(-3.-6.\,x+3.46y)\,(-3.+2.\,x+3.46y)}\right)}{(1.-1.15\,m)^2}$$

To analyze these functions we show both the numerator and the denominator have the same sign. For example, the first equation is:

$$\frac{12.\,(1.5+1.73\,m)\left(0.612+\frac{27.7x\,(m-1.\,y)\,y}{m^2\,(3.+6.\,x-3.46y)\,(-3.+2.\,x+3.46y)}\right)}{(3.+3.46\,m)^2}+\frac{-20.8\,m\left(0.306+\frac{24.\,(m-1.\,y)\left(0.5x-1.\,x^2-0.577x\,y\right)}{m^2\,(-3.-6.\,x+3.46y)\,(-3.+2.\,x+3.46y)}\right)}{(3.+3.46\,m)^2}$$

We can put this over a common denominator, which is:

$$m^2\,(0.866025+1.\,m)^2\,(0.5+1.\,x-0.57735\,y)\,(-1.5+1.\,x+1.73205\,y)$$

The first two terms, m^2 and $(0.866025+1.\,m)^2$, are clearly positive. The third term, $0.5+1.\,x-0.57735\,y$, is positive because $|-.57737y|<|-.57737(0.0541266)|<.5$. The last term is negative because $1(0.5)+1.73025(0.0541266)<1.5$. Therefore the denominator is always negative.
The numerator is:

$$-2.33557\,m^2-1.34844\,m^3-1.38205\,m^2x-1.79793\,m^3x-0.350004\,m^2x^2+1.79793\,m^3x^2+$$
$$5.39378\,m^2y+3.1141\,m^3y+1.73205\,m\,x\,y+5.59585\,m^2x\,y+2.07607\,m^3x\,y-3.1141\,m^2y^2+$$
$$2.07607\,m^3x\,y-1.79793\,m^3y^2-3.4641\,x\,y^2-2.\,m\,x\,y^2$$

We split the numerator into four parts by separating the terms into the different powers of m. Starting with m^0, these coefficients are

$$\begin{aligned}
m_0(x,y) &= -3.4641\, x\, y^2 \\
m_1(x,y) &= 1.73205\, x\, y + 3.4641\, x^2\, y - 2.\, x\, y^2 \\
m_2(x,y) &= -2.33557 - 1.38205\, x - 0.350004\, x^2 + 5.39378\, y + 5.59585\, x\, y - 3.1141\, y^2
\end{aligned}$$

The coefficient m_0 is clearly negative, since x and y are both positive. The coefficient m_1 is positive, since we can factor out the positive term xy leaving us with

$$1.73205 + 3.4641\, s - 2.\, t > 1.6238$$

which is positive. The coefficient m_2 is negative; setting x and y of the positive terms to their maximum values:

$$\begin{aligned}
m2(x,y) &\leq -2.33557 + 5.39378(0.0541266) + 5.59585(0.5)(0.0541266) \\
&= -1.89218
\end{aligned}$$

The coefficient m_3 is negative; setting x and y of the positive terms to their maximum values:

$$m3(x,y) \leq -1.34844{+}1.79793(0.5^2){+}3.1141(0.0541266){+}2.07607(0.5)(0.0541266) = -0.674222$$

All of the coefficients of m except for m_1 are negative. We now show that

$$\begin{aligned}
numer(x,y,m) &> m_0(x,y) + m m_1(x,y) + m^2 m_2(x,y) \\
&= -2.33557\, m^2 - 1.38205\, m^2\, x - 0.350004\, m^2\, x^2 + 5.39378\, m^2\, y + 1.73205\, m\, x\, y + \\
&\quad 5.59585\, m^2\, x\, y + 3.4641\, m\, x^2\, y - 3.1141\, m^2\, y^2 - 3.4641\, x\, y^2 - 2.\, m\, x\, y^2
\end{aligned}$$

For the positive coefficients, set $x = 0.5$ and $t = m$, which are the maximum values they can take.

$$-0.603523\, m^2 + 8.1917\, m^3 - 1.38205\, m^2\, x - 0.350004\, m^2\, x^2 - 3.1141\, m^2\, y - 3.4641\, x\, y - 2.\, m\, x\, y$$

The only positive term left is $8.1917\, m^3$, and since

$$|-0.603523| > 8.1917(0.0541266) = 0.168556$$

The entire numerator is negative. Therefore, since the denominator is also negative, the original equation is positive.
The other cases can be dealt with in a similar manner.

Bibliography

[BB83] R.E. Barnhill and W. Boehm, editors. *Surfaces in Computer-Aided Geometric Design*. North-Holland, Amsterdam, 1983.

[BBB87] R. Bartels, J. Beatty, and B. Barsky. *An Introduction to Splines for Use in Computer Graphics and Geometric Modeling*. Morgan Kaufmann, 1987.

[BBX95] Chanderjit L. Bajaj, Fausto Bernardini, and Guoliang Xu. Automatic Reconstruction of Surfaces and Scalar Fields from 3D Scans. *SIGGRAPH*, 29(2), August 1995.

[BFK84] W. Boehm, G. Farin, and J. Kahmann. A Survey of Curve and Surface Methods in Computer-Aided Geometric Design. *Computer-Aided Geometric Design*, 1(1):1–60, 1984.

[BI92a] Chanderjit L. Bajaj and Insung Ihm. Algebraic Surface Design with Hermite Interpolation. *ACM Transactions on Graphics*, 11(1):61–91, January 1992.

[BI92b] Chanderjit L. Bajaj and Insung Ihm. Smoothing Polyhedra Using Implicit Algebraic Splines. *Computer Graphics*, 26(2):79–88, 1992.

[BR91] Mark Bloomenthal and Richard Riesenfeld. Approximation of Sweep Surfaces by Tensor Product NURBS. In *Proceedings of SPIE Conference*, volume 1610, pages 132–144, November 1991.

[BS88] A. A. Ball and D. J. T. Storry. Conditions for Tangent Plane Continuity Over Recursively Generated B-spline Surfaces. *ACM Transactions on Graphics*, 7(2):83–102, April 1988.

[BS91] Jules Bloomenthal and Ken Shoemake. Convolution Surfaces. *Computer Graphics*, 25(4):251–256, July 1991. Proceedings of SIGGRAPH '91.

[CC78] E. Catmull and J. Clark. Recursively Generated B-spline Surfaces on Arbitrary Topological Meshes. *Computer-Aided Design*, 10(6):350–355, November 1978.

[CF67] G. Cooke and L. Finney. *Homology of Cell Complexes*. Princeton University Press, Princeton, New Jersey, 1967.

[CG91] G. Celniker and D. Gossard. Deformable Curve and Surface Finite-Elements for Free Form Shape Design. *Computer Graphics*, 25(4):257-266, July 1991.

[CK83] H. Chiyokura and F. Kimura. Design of Solids with Free-form Surfaces. *Computer Graphics*, 17(3):289–298, 1983.

[CL91] C. Chui and M. J. Lai. Algorithms for Generating B-nets and Graphically Displaying Spline Surfaces On Three- and Four-directional Meshes. *Computer-Aided Geometric Design*, 8:479–493, 1991.

[CW93] Michael Cohen and John Wallace. *Radiosity and Realistic Image Synthesis*. Academic Press Professional, Cambridge, Massachusetts, 1993.

[Dah89] W. Dahmen. Smooth Piecewise Quadratic Surfaces. *Mathematical Methods in Computer-Aided Geometric Design*, pages 181–193, 1989.

[DK93] Tony DeRose and Michael Kass. Efficient, Fair Interpolation Using Catmull-Clark Surfaces. *Computer Graphics*, 27(2):35–44, July 1993.

[DM84] W. Dahmen and C. Micchelli. Subdivision Algorithms for the Generation of Box Spline Surfaces. *Computer-Aided Geometric Design*, 18(2):115–129, 1984.

[DS78] D. Doo and M. Sabin. Behaviour of Recursive Division Surfaces Near Extraordinary Points. *Computer-Aided Design*, 10(6):356–360, November 1978.

[Far88] G. Farin. *Curves and Surfaces for Computer-Aided Geometric Design*. Academic Press, 1988.

[FB88] David Forsey and Richard Bartels. Hierarchical B-Spline Refinement. *Computer Graphics*, 22(2):205–212, July 1988.

[FS92] P. Fong and H. P. Seidel. An Implementation of Multivariate B-spline Surfaces Over Arbitrary Triangulations. In *Proc. Graphics Interface '92*, pages 1–10, 1992.

[GH95] Cindy Grimm and John Hughes. Smooth Isosurface Approximation. In *Eurographics Workshop on Implicit Surfaces*, pages 57–77, April 1995.

[Gre83] J. A. Gregory. N-sided Surface Patches. *Mathematics of Surfaces*, pages 217–232, 1983.

[Hah89] J. M. Hahn. Filling Polygonal Holes with Rectangular Patches. *Theory and Practice of Geometric Modeling*, pages 81–91, 1989.

[HB94] John Hughes and T. Banchoff. Smooth Subdivision Surfaces. *Unpublished*, 1994.

[HDD93] H. Hoppe, T. DeRose, and T. Duchamp. Mesh Optimization. *Computer Graphics*, 27(4):19–25, July 1993. Proceedings of SIGGRAPH '93.

[HDD+94] Hugues Hoppe, Tony DeRose, T. Duchamp, M. Halstead, H. Jin, J. McDonald, J. Schweitzer, and W. Stuetzle. Piecewise Smooth Surface Reconstruction. *Computer Graphics*, 28(2):295–305, July 1994. Proceedings of SIGGRAPH '94.

[HK84] M. Hosaka and F. Kimura. Non-four-sided Patch Expressions with Control Points. *Computer-Aided Geometric Design*, 1(1):75–86, January 1984.

[HM90] K. Hollig and H. Mogerle. G-splines. *Computer-Aided Geometric Design*, 7:197–207, 1990.

[Hof89] Christoph M. Hoffmann. *Geometric and Solid Modeling: An Introduction.* Morgan Kaufmann Pub, 1989.

[Jen87] T. Jensen. Assembling Triangular and Rectangular Patches and Multivariate Splines. In G. Farin, editor, *Geometric Modeling: Algorithms and New Trends, SIAM*, pages 203–222, Philadelphia, PA, 1987.

[Kje83] J. A. Kjellander. Smoothing of Bicubic Parametric Surfaces. *Computer-Aided Design*, 15:288–293, 1983.

[KR90] M. Kallay and B. Ravani. Optimal Twist Vectors as a Tool for Interpolation of a Network of Curves with a Minimal Energy Surface. *Computer-Aided Geometric Design*, 1990.

[Lai95] David H. Laidlaw. *Geometric Model Extraction from Magnetic Resonance Volume Data.* PhD thesis, California Institute of Technology, 1995.

[LC87] W.E. Lorenson and H.E. Cline. Marching Cubes: A High Resolution 3D Surface Construction Algorithm. *Computer Graphics*, 21(4):163–169, July 1987.

[LD89] Charles Loop and Tony DeRose. A Multisided Generalization of Bézier Surfaces. *ACM Transactions on Graphics*, 8(3):204–234, July 1989.

[LD90] Charles Loop and Tony DeRose. Generalized B-spline Surfaces of Arbitrary Topology. *Computer Graphics*, 24(4):347–357, August 1990.

[Loo87] Charles Loop. Smooth Subdivision Surfaces Based on Triangles. Master's thesis, University of Utah, 1987.

[Loo94a] Charles Loop. A G^1 Triangular Spline Surface of Arbitrary Topological Type. *Computer-Aided Geometric Design*, 11:303–330, 1994.

[Loo94b] Charles Loop. Smooth Spline Surfaces Over Irregular Meshes. *Computer Graphics*, 28(2):303–310, July 1994.

[Man88] Martti Mantyla. *Introduction to Solid Modeling.* W H Freeman and Co., 1988.

[MBL+91a] J. Miller, D. Breen, W. Lorensen, R. O'Bara, and M. Wozny. Geometrically Deformed Models: A Method for Extracting Closed Geometric Models from Volume-Data. *Computer Graphics*, 25(4):217–225, July 1991. Proceedings of SIGGRAPH '91.

[MBL+91b] J.V. Miller, D.E. Breen, W.E. Lorensen, R.M. O'Bara, and M.J. Wozny. Geometrically Deformed Models: A Method for Extracting Closed Geometric Models from Volume Data. *Computer Graphics*, 25(4):217–226, July 1991. Proceedings of SIGGRAPH '91.

[MLL+92] S. Mann, C. Loop, M. Lounsbery, D. Meyers, J. Painter, T. DeRose, and K. Sloan. A Survey of Parametric Scattered Data Fitting Using Triangular Interpolants. In H. Hagen, editor, *Curve and Surface Modeling, SIAM*, pages 145–172, Philadelphia, PA, 1992.

[MP77] Richard Millman and George Parker. *Elements of Differential Geometry.* Prentice-Hall, Inc., 1977.

[MS92] Henry Moreton and Carlo Séquin. Functional Optimization for Fair Surface Design. *Computer Graphics*, 26(4):167–176, July 1992.

[Mun75] James R. Munkres. *Topology, a First Course.* Prentice-Hall, Inc., 1975.

[Mur91] S. Muraki. Volumetric Shape Description of Range Data using "Blobby Model". *Computer Graphics*, 25(4):227–235, July 1991. Proceedings of SIGGRAPH '91.

[MYV93] J. Maillot, H. Yahia, and A. Verroust. Interactive Texture Mapping. *Computer Graphics*, 27(4):27–35, July 1993. Proceedings of SIGGRAPH '93.

[Nas87] Ahmad H. Nasri. Polyhedral Subdivision Methods for Free-form Surfaces. *ACM Transactions on Graphics*, 6(1):29–73, January 1987.

[NP94] M. Neamtu and P. R. Pfluger. Degenerate Polynomial Patches of Degree 4 and 5 Used for Geometrically Smooth Interpolation in $\Re^3$. *Computer-Aided Geometric Design*, 11:451–474, 1994.

[Ped95] Hans Köhling Pedersen. Decorating Implicit Surfaces. *Computer Graphics*, 29(2):291–300, August 1995. Proceedings of SIGGRAPH '95.

[Pet92] Jörg Peters. Smooth Free-form Surfaces Over Irregular Meshes Generalizing Quadratic Splines. *Computer-Aided Geometric Design*, 10:347–361, 1992.

[Pet93] Jörg Peters. Constructing C^1 Surfaces of Arbitrary Topology Using Biquadratic and Bicubic Splines. *Desigining Fair Curves and Surfaces*, 1993.

[Sab83] M. Sabin. Non-rectangular Surface Patches Suitable for Inclusion in B-spline Surface. In *Eurographics*, pages 57–69, 1983.

[Sed85] T. W. Sederberg. Piecewise Algebraic Surface Patches. *Computer-Aided Geometric Design*, 2(1-3):53–59, 1985.

[Sed90] T. W. Sederberg. Techniques for Cubic Algebraic Surfaces. *IEEE Computer Graphics and Applications*, 10(5):12–21, 1990.

[Spi70] Michael Spivak. *Differential Geometry Volume 1.* Publish or Perish Inc., 1970.

[SS95] Peter Schröder and Wim Sweldens. Spherical Wavelets: Efficiently Representing Functions on the Sphere. *Computer Graphics*, 29(2):161–169, August 1995. Proceedings of SIGGRAPH '95.

[ST67] I.M. Singer and John A. Thorpe. *Lecture Notes on Elementary Topology and Geometry.* Scott, Foresman and Company, Glenview, Illinois, 1967.

[SZL92] W.J. Schroeder, J.A. Zarge, and W.E. Lorenson. Decimation of Triangle Meshes. *Computer Graphics*, 26(2):65–70, July 1992.

[TC86] S. T. Tan and K. C. Chan. Generation of Higher Order Surfaces Over Arbitrary Polyhedral Meshes. *CAD*, 18(8):411–423, 1986.

[TL94] Greg Turk and Marc Levoy. Zippered Polygon Meshes from Range Data. *Computer Graphics*, 28(2):311–319, July 1994. Proceedings of SIGGRAPH '94.

[War89] Joe Warren. Blending Algebraic Surfaces. *ACM Transactions on Graphics*, 8(4):263–278, 1989.

[War92] Joe Warren. Creating Multisided Rational Bézier Surfaces Using Base Points. *ACM Transactions on Graphics*, 11(2):127–139, April 1992.

[WH94] Andrew Witkin and Paul Heckbert. Using Particles to Sample and control Implicit Surfaces. *Computer Graphics*, 28(2):269–279, July 1994. Proceedings of SIGGRAPH '94.

[Wij86] J. Van Wijk. Bicubic Patches for Approximating Non-rectangular Control Point Meshes. *Computer-Aided Geometric Design*, 3(1-13):1–13, 1986.

[WMW86] G. Wyvill, C. McPheeters, and B. Wyvill. Data Structure for Soft Objects. *Visual Computer*, 2(4):227–234, August 1986.

[Wol91] Stephen Wolfram. *Mathematica: A System for Doing Mathematics by Computer.* Addison-Wesley, 1991.

[WW92] William Welch and Andrew Witkin. Variational Surface Modeling. *Computer Graphics*, 22(2):157–166, July 1992.

[WW94] William Welch and Andrew Witkin. Free Form Shape Design Using Triangulated Surfaces. *Computer Graphics*, 28(2):247–256, July 1994.