Triangular Bézier Surfaces with Approximate Continuity

by

Yingbin Liu

A Thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Computer Science

Waterloo, Ontario, Canada, 2008

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

When interpolating a data mesh using triangular Bézier patches, the requirement of C^1 or G^1 continuity imposes strict constraints on the control points of adjacent patches. However, fulfilment of these continuity constraints cannot guarantee that the resulting surfaces have good shape. This thesis presents an approach to constructing surfaces with approximate C^1/G^1 continuity, where a small amount of discontinuity is allowed between surface normals of adjacent patches. For all the schemes presented in this thesis, although the resulting surface has C^1/G^1 continuity at the data vertices, I only require approximate C^1/G^1 continuity along data triangle boundaries so as to lower the patch degree.

For functional data, a cubic interpolating scheme with approximate C^1 continuity is presented. In this scheme, one cubic patch will be constructed for each data triangle and upper bounds are provided for the normal discontinuity across patch boundaries.

For a triangular mesh of arbitrary topology, two interpolating parametric schemes are devised. For each data triangle, the first scheme performs a domain split and constructs three cubic micro-patches; the second scheme constructs one quintic patch for each data triangle. To reduce the normal discontinuity, neighbouring patches across data triangle boundaries are adjusted to have identical normals at the middle point of the common boundary. The upper bounds for the normal discontinuity between two parametric patches are also derived for the resulting approximate G^1 surface.

In most cases, the resulting surfaces with approximate continuity have the same level of visual smoothness and in some cases better shape quality.

Acknowledgements

First I would like to thank my supervisor, Stephen Mann, for all his advices and support during my research work. I want to thank my reading committee: Justin Wan, Marshall Walker, Michael McCool and Tony Wirjanto for their help on my thesis. I also want to thank all my friends at University of Waterloo. Finally I would thank my mother for all her love and support.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The history of spline curves and surfaces can be traced back to the 1950's, when curves represented in Bézier form came into being and a new field called Computer Aided Geometric Design (CAGD) emerged. One CAGD problem is to construct a surface based on a data set. The resulting surface is often represented as a polynomial because polynomial surfaces are mathematically simple and easy to evaluate. However, using one polynomial for the whole data set requires high degree, which makes evaluation expensive. Therefore, most surface schemes construct a surface using multiple polynomial patches, where adjacent patches are required to share a common boundary curve and fulfill certain continuity constraints.

In this thesis, I use triangular Bézier patches to construct surfaces over arbitrary triangular meshes. Since the resulting surface is composed of many patches, it is important to ensure smoothness along patch boundaries. Researchers have paid considerable attention to creating surfaces that satisfy parametric or geometric continuity conditions. For most existing surface schemes, C^1/G^1 continuity is required along the boundary between two adjacent patches.

However, many industrial applications only require visual smoothness on the resulting surface, *i.e.*, the surface "looks" smooth, even though adjacent patches actually do not meet with C^1/G^1 continuity. On the other hand, these applications put an emphasis on the surface shape. Generally, the perception of a good surface shape is subjective. Different engineering applications and users may have different criteria to define a good surface shape. However, we usually have unanimous consent on some aspects of "bad" surface shape and wish to avoid things such as inflection points, cusps, dents, *etc.*

The primary objective of my work is to find a surface construction method that relaxes the C^1/G^1 continuity constraints so that I can use low degree patches (ideally cubic) to generate surfaces with visual smoothness and pleasing surface shape. In this dissertation, I will focus on the problem of constructing a surface with approximate continuity from a given triangular data mesh, using triangular Bézier patches.

In the design of surface schemes, there exists a gap between mathematics and practice. Even when the mathematical conditions are fulfilled, practical problems may occur and lead to unexpected results. For example, surfaces of many existing C^1/G^1 schemes have poor shape quality [27]. Another example is in the method introduce in chapters 3, 4 and 5, where mathematical derivation suggests that the more equal-normal points we have on the boundary curve, the lower the cross-boundary discontinuity. However, I found in experiments that the methods using two and three equal-normal points sometimes generate surfaces with higher discontinuity than the one-point method (Chapter 5). Therefore, not only developed mathematics for my work, but I also tested the results on different types of data meshes. When examining the surface shape quality, I used additional subjective criteria such as analyzing shaded images and compared the results to those of C^1/G^1 continuous schemes.

The primary contributions of this dissertation include:

1. I designed three surface fitting schemes using ϵ-C^1/ ϵ-G^1 continuity to construct interpolating surfaces with similar shape, but lower degree than comparable C^1/G^1 schemes.

2. I created an ϵ-G^1 method to reduce the normal discontinuity across patch boundaries and I conducted an analysis of this method. I used variations of this method in my parametric ϵ-G^1 surface fitting schemes.

In the remaining sections of this chapter, I will start introducing the idea of approximate continuity by reviewing some fundamentals of triangular Bézier surfaces and surface continuity. A surface interpolation scheme with approximate continuity for functional surfaces is introduced in Chapter 2, together with the analysis on upper bounds of the normal discontinuity angle. In Chapter 3, an interpolation scheme for parametric surfaces with a domain split is presented, and another scheme without a domain split is introduced in Chapter 4. Detailed analysis of upper bounds for the parametric surfaces with approximate continuity is discussed in Chapter 5. I will conclude my thesis with a summary and proposal for future research in Chapter 6.

1.1 Triangular Bézier patches

In 1911, *Sergei Bernstein*, a Russian mathematician, introduced *Bernstein Polynomials* as spline functions and gave a constructive proof of *Weierstrass's theorem*, namely that a continuous function on a finite subinterval of the real line can be uniformly approximated as closely as we wish by a polynomial.

A degree n binomial Bernstein polynomial is defined as

$$B_{ij}^{n}(u_0, u_1) = \frac{n!}{i!j!} u_0^i u_1^j, \tag{1.1}$$

where $u_0 + u_1 = 1$ and $i + j = n$. Equation 1.1 can be re-written as

$$B_i^n(t) = \frac{n!}{i!(n-i)!} t^i (1-t)^{n-i}. \tag{1.2}$$

Bernstein polynomials of degree n form a basis for all polynomials of degree equal to or less than n, hence each polynomial has a Bernstein representation.

Bézier curves, named after *Pierre Bézier*, an engineer from *Rénaut*, are a curve representation relative to Bernstein polynomials:

$$C(t) = \sum_{i=0}^{n} P_i B_i^n(t). \tag{1.3}$$

Here $B_i^n(t)$ are the Bernstein polynomials and the P_i are the control points. Figure 1.1 shows examples of two Bézier curves, with a quadratic curve on the left and a cubic curve on the right.

We can easily expand the definition of Bernstein polynomials to the trinomial case:

$$B_{ijk}^{n}(u_0, u_1, u_2) = \frac{n!}{i!j!k!} u_0^i u_1^j u_2^k,$$

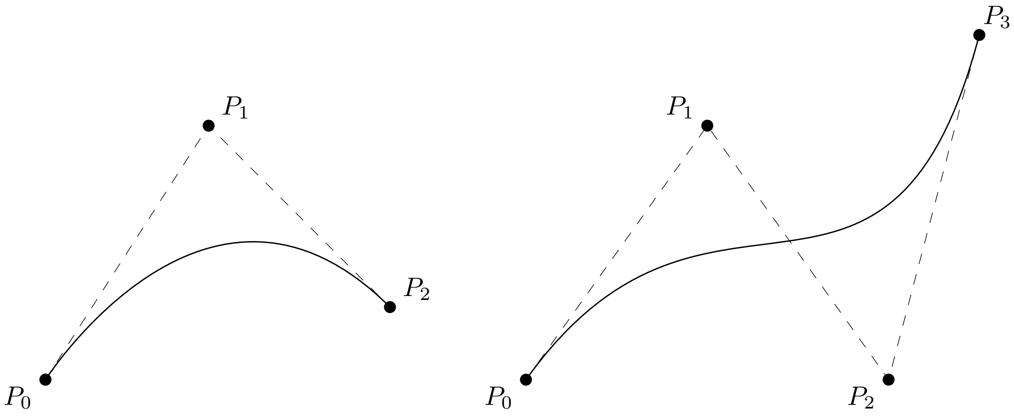


Figure 1.1: Bézier curves.

with $u_0 + u_1 + u_2 = 1$ and $i + j + k = n$. Here (u_0, u_1, u_2) are the Barycentric coordinates relative to the domain triangle. A degree n triangular Bézier patch is therefore defined over a triangular domain using Barycentric coordinates:

$$S(u_0, u_1, u_2) = \sum_{i+j+k=n} P_{ijk} B^n_{ijk}(u_0, u_1, u_2).$$

Here P_{ijk} are the control points. As shown in Figure 1.2, control points of a cubic patch are connected to form a *control mesh*. The triangular patch representation is a mapping from a two-dimensional triangular domain to a two-dimensional manifold.

Some of the nice properties of triangular Bézier patches include:

- Every point on the patch is an *affine combination* of the control points. [1]
- A triangular Bézier patch passes through its three corner control points.
- The tangent plane at each corner vertex is defined by the corresponding control panel.
- All three patch boundaries are Bézier curve, defined by the boundary control points of the patch.

---

[1] In CAGD, we like affine combinations since they have nice geometric interpretations. An affine combination is a linear combination whose coefficients sum to 1.

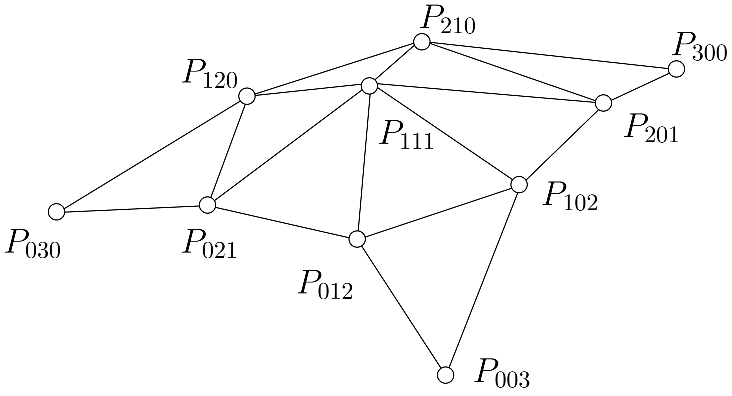


Figure 1.2: Control points of a cubic patch.

- It is efficient to evaluate a triangular Bézier patch using *de Casteljau*'s algorithm, especially when we need to calculate the surface normal.

These properties of triangular Bézier patches make them suitable for surface fitting on data meshes. All the spline surfaces discussed in this dissertation are constructed using triangular Bézier patches.

1.1.1 de Casteljau's algorithm

In 1959, *Paul de Faget de Casteljau*, an engineer from *Citroën*, invented a recursive algorithm to evaluate Bézier curves [5]. The de Casteljau's algorithm was later generalized to surfaces and used to evaluate triangular Bézier patches. The basic idea of this algorithm is that the desired curve or surface point can be found with repeated affine combinations of the control points.

Using de Casteljau's algorithm, a point P on a degree n patch with barycentric coordinates (u_0, u_1, u_2) can be calculated as

$$\begin{aligned}
P^0_{ijk} &= P_{ijk}, \\
P^{l+1}_{ijk} &= u_0 P^l_{ijk+100} + u_1 P^l_{ijk+010} + u_2 P^l_{ijk+001}, \qquad l = 0, 1, \ldots, n-1, \\
P &= P^n_{000},
\end{aligned}$$

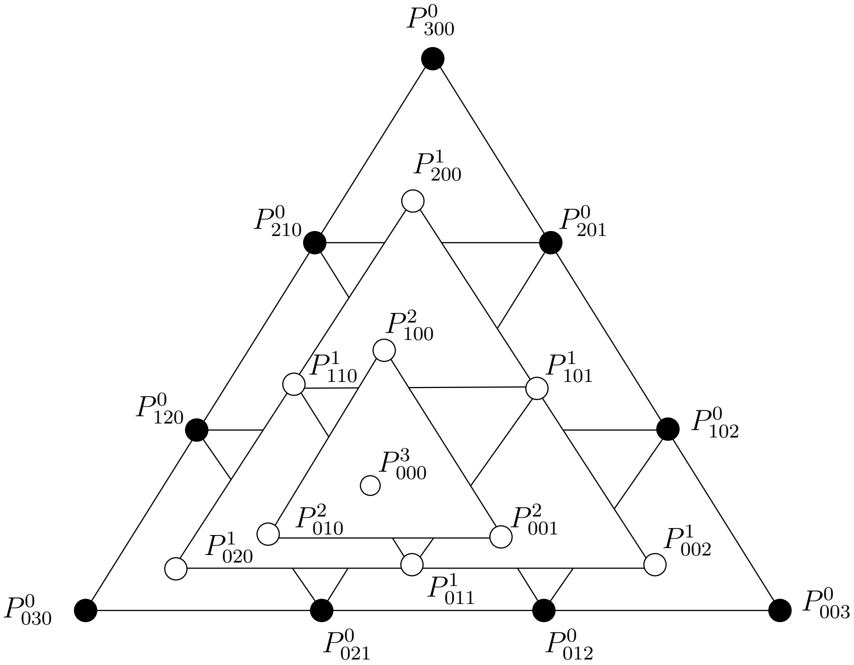


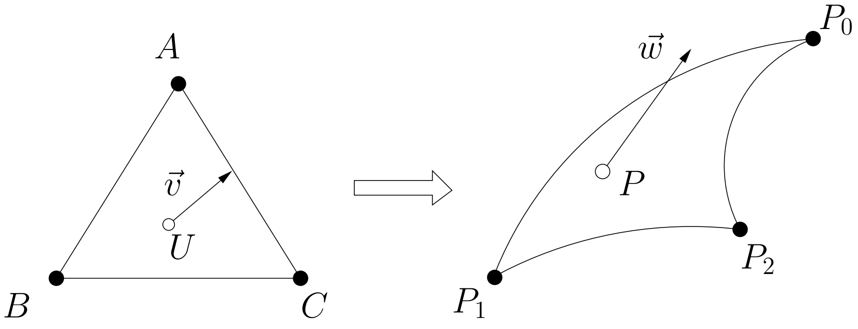
Figure 1.3: Patch evaluation using de Casteljau's algorithm.

where P_{ijk} are the control points of the patch. Figure 1.3 presents the evaluation process of a cubic patch, with the initial control points shown as black dots and the intermediate points in white. All the points in Figure 1.3 are labeled as in Equation 1.4, and the resulting points of the same step are connected. In Figure 1.3, the upward control point triangles are known as *control panels.*

At the last step of the evaluation, P is located on the plane of $\triangle P^2_{100}P^2_{010}P^2_{001}$, which defines the surface tangent plane at P (Figure 1.3). The normal vector at P is calculated as the cross product of two difference vectors in $\triangle P^2_{100}P^2_{010}P^2_{001}$:

$$\vec{N}_P = (P^2_{100} - P^2_{010}) \times (P^2_{001} - P^2_{010})$$

de Casteljau's algorithm can also be used to calculate the derivative in an arbitrary direction. As shown in Figure 1.4, letting $\vec{v}$ be a direction vector and U be the preimage of P relative to the domain triangle $\triangle ABC$, there are two ways to generate the derivative:



$$S(U) = P \qquad \frac{\partial S}{\partial \vec{v}}(U) = \vec{w}$$

Figure 1.4: Mapping of a Bézier patch.

1. Run de Casteljau's algorithm at U for $(n-1)$ times, then map $\vec{v}$ to the remaining triangle.
2. Map $\vec{v}$ in each control panel independently, then evaluate $(n-1)$ times at U to find the final result.

Higher order derivatives can be obtained by applying another derivative to the first derivative function to generate the second derivative, and so on. Note that the higher order derivatives can be mixed partial derivatives.

1.1.2 Patch subdivision and extrapolation

Other than evaluation, de Casteljau's algorithm also subdivides a triangular Bézier patch by evaluating the patch at a specified split point. For the same cubic patch shown in Figure 1.3, we evaluate point P on the cubic patch using de Casteljau's algorithm. As shown in the left of Figure 1.5, U is the preimage of P, relative to the domain triangle $\triangle ABC$. After the evaluation, ABC is spit into three sub-triangles with U as the new vertex; the intermediate results generated by the evaluation can be divided into three sub-patches. As shown in Figure 1.5, the three sub-patches are composed with the points

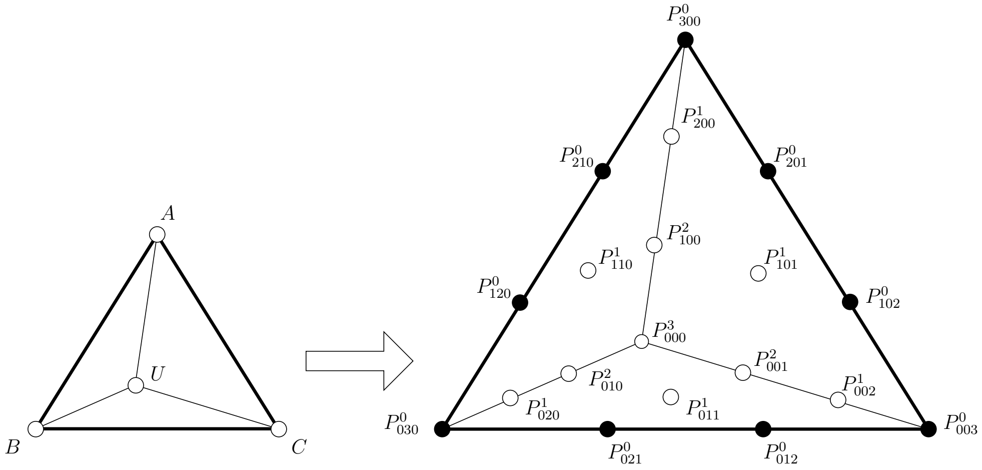


Figure 1.5: Patch split.

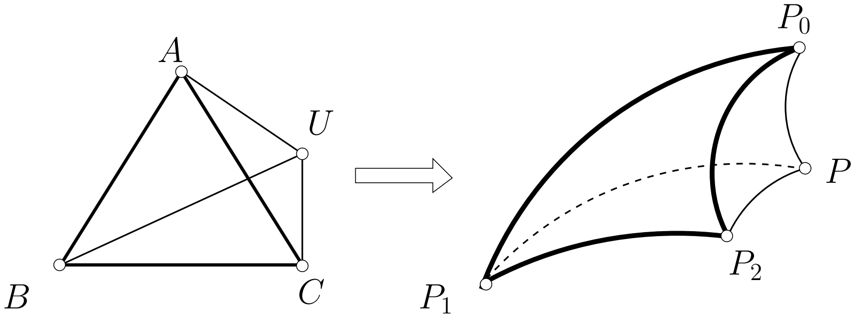


Figure 1.6: Patch extrapolation.

labeled as P^l_{ij0}, P^l_{0jk} and P^l_{i0k} respectively, with $l = 0, 1, 2, 3$. These three sub-patches define the same surface patch as before the subdivision.

If U, the preimage of P, is outside the domain triangle $\triangle ABC$, as shown in Figure 1.6, evaluating at U becomes an extrapolation and the intermediate control points expand the surface patch to P. As shown in Figure 1.6, we can divide the intermediate control points generated by the extrapolation into three micro-patches as well, with domain triangles $\triangle ABC$, $\triangle ABU$ and $\triangle UBC$ respectively.

1.1.3 Degree elevation

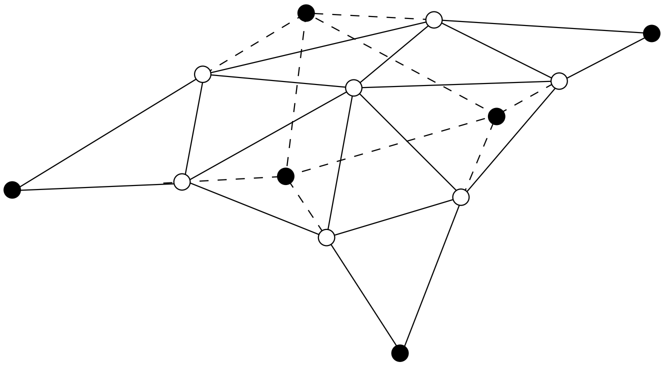


Figure 1.7: Degree elevation of a quadratic patch.

For a degree n triangular Bézier patch, we can compute the Bézier control points of the same polynomial relative to degree $n+1$ Berstein polynomials [11]. This transformation is called *degree elevation* or *degree raising*. For a degree n Bézier patch with control points P_{ijk}, the same polynomial can be represented relative to the degree $n+1$ Berstein polynomials with control points Q_{rst} :

$$\sum_{ijk}^{n} P_{ijk} B_{ijk}^{n}(u_0, u_1, u_2) = \sum_{rst}^{n+1} Q_{rst} B_{rst}^{n+1}(u_0, u_1, u_2),$$

$$Q_{rst} = \frac{1}{n+1}\left(rP_{rst-(1,0,0)} + sP_{rst-(0,1,0)} + tP_{rst-(0,0,1)}\right), \tag{1.4}$$

where $i+j+k=n$ and $r+s+t=n+1$. For the control points of a quadratic patch shown as black dots in Figure 1.7, the patch degree is elevated to cubic and the newly generated control points are shown as white dots. Note that the cubic representation has the same corner control points as the quadratic representation. We can raise the patch degree further by applying Equation 1.4 multiple times on the resulting patch.

1.2 Surface continuity

Many CAD/CAM applications have aesthetic demands (such as a car body) and physical requirements deduced from aerodynamic or hydrodynamic laws (like air crafts or ship hulls). For all such applications, a certain order of continuity is required between adjacent patches. In mathematics, the continuity of a curve can be analyzed by differentiating the curve with respect to the parameter. For a surface, continuity is still analyzed by differentiation of the surface, after taking partial derivatives into account.

Basically, there are two kinds of continuity conditions associated with spline surfaces. The first is called *parametric continuity* (C^k or k^{th} order parametric continuity), also known as algebraic continuity, which requires partial derivatives to agree where the surface patches meet. The second category of the surface continuity is *geometric continuity* (G^k or k^{th} order geometric continuity), which is parameterization independent.

1.2.1 Parametric continuity

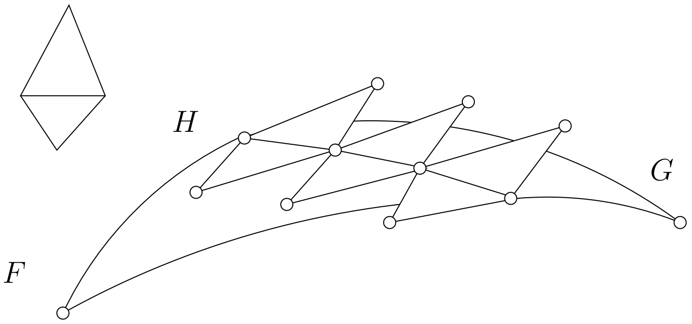


Figure 1.8: Two functional patches meeting with C^1 continuity.

Parametric continuity places emphasis on parameterization and has a close relationship with algebra, but it does not necessarily reflect the smoothness of the surfaces. For triangular Bézier surfaces, k^{th} order of parametric continuity means that for any point on the surface, the k^{th} order of derivatives in an arbitrary direction are continuous. Note that the derivatives discussed here are directional derivatives, and a direction in the domain is required to specify a vector in the tangent plane. A surface with C^k continuity must also be C^{k-1} continuous, $k \geq 1$. For example, a C^2 surface must be C^1 and C^0 continuous. At the interior of a Bézier patch, derivatives are infinitely continuous, hence the patch is C^∞ everywhere in its interior.

Generally, there are geometric or physical meanings associated with C^k continuity.

- C^0: There is no gap, break or hole on the surface, which is "watertight".
- C^1: The first derivative is continuous along any direction on the surface. In physics, the first derivative denotes the velocity, and a C^1 surface means that if we are moving along the surface, then we get a continuous (not necessarily constant) velocity. In geometric design, if the surface is not C^1 continuous at a point, we will have a non-continuous feature, such as a crease (suppose the control points do not overlap and no singularities in the data mesh).

For two adjacent patches to join with C^1 continuity, every pair of boundary control panels must be affine images of the domain triangles [21, 30]. As shown in Figure 1.8, patches F and G meet at a common boundary curve H, with the boundary control panels outlined and the domain triangles shown in the upper left corner. The geometric interpretation of the C^1 continuity condition is that we can re-parameterize one patch over the domain of the neighbouring patch and compare the boundary layer of control points. Using the extrapolation method introduced in Section 1.1.2, we first re-parameterize F (Figure 1.8) over the domain of G and generate a set of new control points. Patches F and G meet with C^1 continuity if the first two rows of new points from the boundary match with the corresponding control points of G.

The C^1 conditions are simplified for functional surfaces (*i.e.*, where $z = f(x, y)$). Since all the domain triangles are identical in the functional case, the C^1 continuity conditions are met if each pair of the control panels along the boundary is coplanar [21, 30].

1.2.2 Geometric continuity

Since parametric continuity require the domains to generate the derivatives, it is inconvenient to use parametric continuity for surfaces with arbitrary topology. Furthermore, parametric continuity disallows many parameterizations that would generate geometrically smooth surfaces. As a relaxed form of parametric continuity, geometric continuity (G^k or k^{th} order geometric continuity) was introduced. The constraints imposed by geometric continuity accommodate the differences between different parameterizations of adjacent surface patches.

Geometric continuity, in essence, is the existence of a (local) re-parameterization. DeRose [6] and Hahn [20] discussed the detailed mathematical meanings of geometric continuity. Generally, the arc element, *i.e.*, the first fundamental form, is considered an invariant parameter as used in differential geometry [11]. Then a G^k continuous surface is up to the k^{th} order differentiable with respect to the arc length, although not necessarily k^{th} order differentiable with respect to the current parameterization.

For specific meanings of G^k continuity used in my work, we have

- G^0 : Same as C^0 continuity, there is no break/gap/hole on the surface.
- G^1 : There exists a continuous tangent plane across the surface.

Currently, G^1 continuity is the most widely used form of continuity in CAGD. A C^1 continuous non-degenerate surface is naturally G^1 continuous (with non-vanishing derivatives), but not vice versa. Geometrically continuous surfaces are useful in practice, particularly for modelling various situations where ordinary C^1 continuous surfaces cannot be constructed, for example, a star-shaped patch. They are also useful for joining various kinds of patches, such as triangles and rectangles [19].

For triangular Bézier surfaces, two patches meeting with G^0 continuity share the same sequence of control points along the boundary. If the two patches meet with G^1 continuity, their tangent planes must be co-planar along the common boundary. For patches F and G (whose domains are shown in Figure 1.9), we evaluate them symbolically at a parameter value on the boundary H to yield a point P. If the two normal vectors calculated in F and G are identical (after normalization, and they point to the same side of the surface) at P, then F and G have the same tangent plane at P, *i.e.*, they meet with G^1 continuity across the boundary.

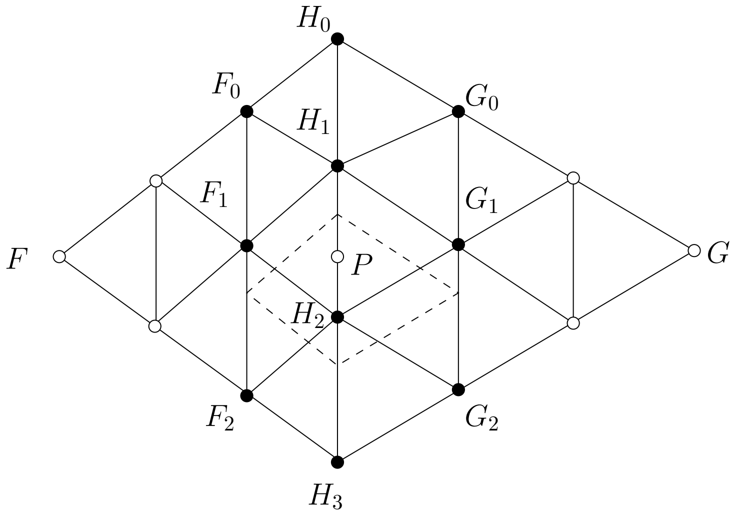


Figure 1.9: Domains of two neighboring cubic patches.

1.2.3 Approximate continuity

Although geometric continuity is already a relaxed form of parametric continuity, it still requires considerable calculation to find a solution to fulfill the constraints. For example, resolving the cross boundary G^1 continuity constraints of two abutting cubic triangular patches involves solving four quadratic equations. Further, satisfying geometric continuity conditions often lead to surfaces with poor shapes. The object of my work is to find a solution to relax the continuity conditions that provides a trade-off between continuity and shape quality.

Instead of satisfying the continuity constraints of C^1 or G^1, I will trade them off for better overall geometric quality and lower cost. By using *approximate continuity*, small discontinuities in the surface normals will be allowed across the patch boundaries. Approximate continuity is a relaxation of C^1/G^1 continuity where patches meet with C^0 continuity, but the discontinuity in the normals between adjacent patches is small. The definition of approximate continuity is as follows [27]:

> *A piecewise, C^0 surface is defined to be ε-C^1 or ε-G^1 if the maximum angle between two surface normals across the patch boundary is bounded by ε.*

Here ε is an upper bound on the normal discontinuity, a metric to estimate the level of smoothness of a surface constructed with ε-C^1/ε-G^1 continuity. As stated, the definition of ε-C^1/ε-G^1 allows a surface to have a "razor edge"; these surfaces are not considered to be ε-C^1 for $\varepsilon < 90$ degrees. For functional surfaces, C^1 continuity conditions are relaxed to be ε-C^1; for parametric surfaces, G^1 conditions are relaxed to be ε-G^1.

Although surfaces with approximate continuity allow some normal discontinuity, it does not mean that any surfaces with discontinuities are ε-C^1/ε-G^1 surfaces. The value of ε should be limited to such a small amount that the discontinuity is not observable on resulting surfaces. When setting the control points that define the cross boundary continuity, ε-C^1/ε-G^1 schemes use simpler methods than the schemes with C^1/G^1 continuity. The goal of designing ε-C^1/ε-G^1 schemes is to use lower degree patches to construct functional or parametric surfaces, yet still generate visually smooth surfaces with good shape quality. Moreover, we hope the overall surface shape can be improved by applying the idea of approximate continuity.

1.3 Polynomial surface interpolation

In this dissertation, I have applied the idea of approximate continuity to surface fitting. For a set of scattered data points, we need to construct a surface that interpolates the data points and possibly meets other constraints. Different surface methods demand different amounts of computational time and memory, as well as resulting in varied quality of the constructed surfaces. There is no one method that works well in all cases.

Existing surface fitting schemes include methods of *polynomial/rational polynomial*, *transfinite*, *subdivision/wavelet*, *radial basis function*, *etc.* With each category of method, we can construct surfaces either interpolating or approximating the input data mesh. Each method has its own strength and drawbacks, suitable for different applications. Since polynomial methods have simplest explicit representation, all the schemes I designed in this thesis are polynomial methods.

Polynomial schemes are further classified into functional and parametric schemes, because of the different patch domains. Although functional surface schemes can handle many problems, they are incapable of modelling surfaces with arbitrary topology, such as a sphere. Another type of interpolation using parametric surfaces was introduced for data sets with arbitrary topology. The popular schemes in this field include Shirman-Séquin [38], Loop [25], Hahmann [17]. All such schemes use at least quartic patches, and most of them only guarantee G^1 continuity across the patch boundaries [31].

Surface fitting methods can also be categorized into *global methods* and *local methods.* The global methods create the surface using all the data points at once. If any one of the data points changes, the problem has to be solved over again and the entire surface changes. Global methods can generate surfaces with more uniform curvature distribution, but they are computationally slow. The local methods perform the interpolation locally. If any one of the data points changes, only a part of the interpolated surface has to be re-computed. Generally, local methods do not generate results as good as the global methods. Sometimes, global and local methods can be combined [12][32]. In my work, all the interpolation schemes are local methods, and they are limited to triangular Bézier patches. I assume that all the scattered data points have already been properly triangulated, and the intersection of the interior of every pair of triangles is empty. Each data point is a vertex of at least one non-degenerate triangle, and every triangle is defined by exactly three points.

Most local interpolation methods usually have three steps:

1. Construct boundary curves connecting the input vertices.
2. Set the interior control points along boundaries to achieve cross-boundary continuity.
3. Set any remaining interior points.

The second step has to handle the *vertex consistency* problem, which involves finding a consistent mixed partial terms for each patch meeting at a vertex [26]. Based on how the vertex consistency problem is solved, surface fitting methods can also be classified into two categories. The first kind of method performs a domain split for each data triangle, such as the Clough-Tocher scheme [3]. Another category of schemes create only one patch for each data triangle, such as Loop's scheme [25]. I will further discuss the vertex consistency problem and Loop's scheme in Chaper 4.

1.4 Surface interrogation

For most applications, the analysis of surface shape quality is equally important as the surface construction process. As an intrinsic property, the continuity conditions on a piecewise polynomial surface provides a straightforward measurement for the surface quality. We can evaluate a surface mathematically to determine its continuity condition, such as C^1, G^1, C^2, *etc.* Although such a mathematical evaluation is efficient and accurate, it does not allow designers to examine the surface shape quality interactively.

The process of extracting and visualizing the geometric information from a surface is called *surface interrogation* [16]. I use surface interrogation methods to locate regions with poor shape, especially along the patch boundaries where approximate continuity is employed. There are various visualization techniques available with the goal of identifying unwanted shapes on the surface. For surfaces in this dissertation, I used three methods to visualize the geometric properties:

1. Shaded images: Diffuse images of the constructed surfaces.
2. Gaussian curvature plots: Colored image of the Gaussian curvature values.
3. Isophotes: Curves with equal light intensity above the surface.

The shaded images presents the information about the C^1/G^1 continuity of a surface, while Gaussian curvature plots and isophotes curves describe the C^2/G^2 conditions. The primary interrogation method I used is rendering shaded images. However, even if a surface looks visually smooth in the shaded images, it may still have unwanted curvature regions. To further examine the surface shape quality, Gaussian curvature plots and isophotes curves are generated.

1.4.1 Shaded images

To visualize the surface shape, shaded images of a surface are rendered using standard OpenGL lighting with ambient, diffuse and specular materials. The result is what a real surface would look like under certain conditions.

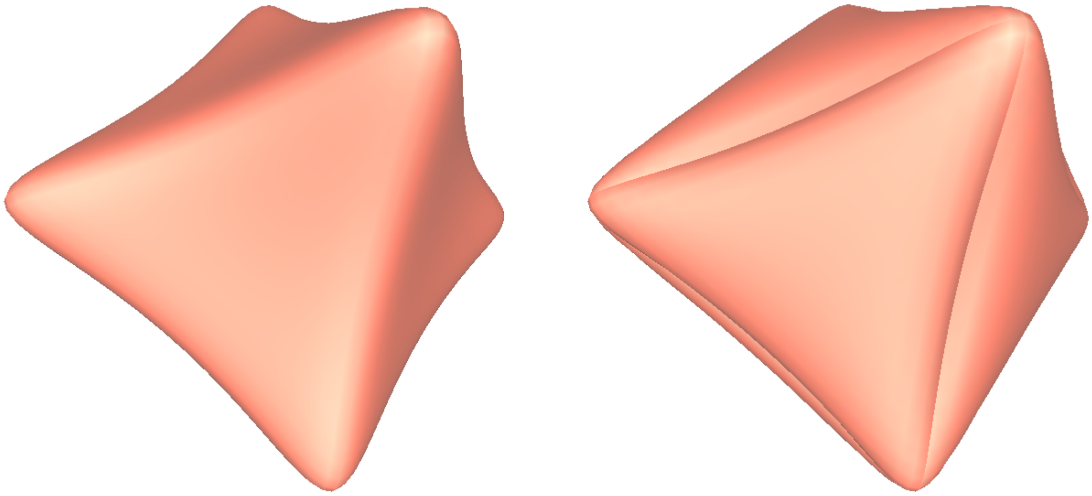


Figure 1.10: Examples of shaded images.

In Figure 1.10, both surfaces were constructed using the same octahedron data mesh. On the G^1 continuous surface shown on the left of Figure 1.10, we see no discontinuities along the patch boundaries. However, creases along boundaries are visible on the right surface of Figure 1.10, where patches meet with approximate continuity. One of the primary uses of shaded images in this thesis is to ensure poor shapes, such as creases, are not visible on the surfaces constructed with approximate continuity.

1.4.2 Gaussian curvature plots

The Gaussian curvature at a given surface point is an intrinsic measure of the surface shape. The value of Gaussian curvature depends only on how distances are measured on the surface, not on the way it is embedded in space. Gaussian curvature is formally defined as

$$K = \frac{eg - f^2}{EG - F^2}.$$

Here E, F and G are coefficients of the first fundamental form and e, f and g are coefficients of the second fundamental form [15].

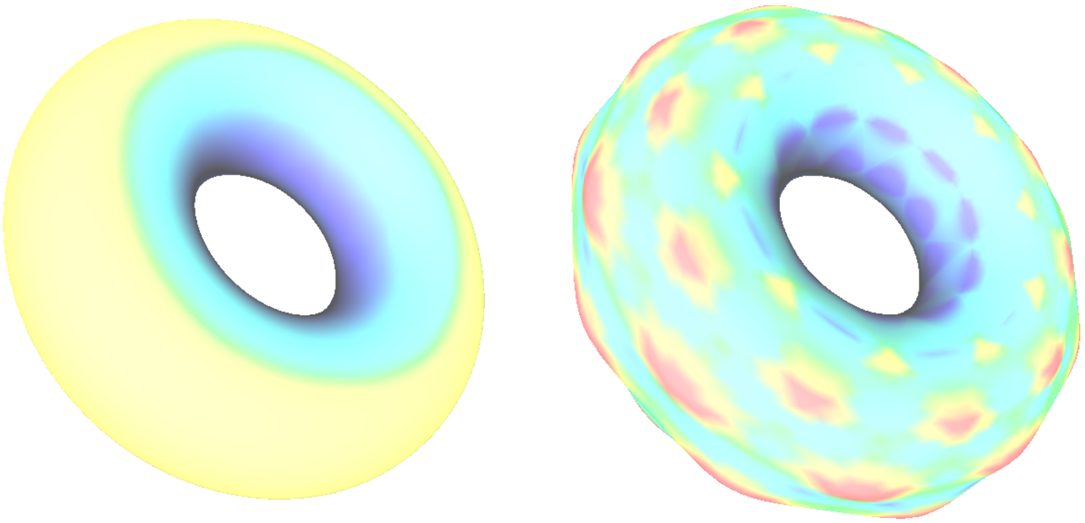


Figure 1.11: Examples of Gaussian curvature plots.

Gaussian curvature indicates the amount by which a geometric surface deviates from being flat. The value of Gaussian curvature is positive for spheres, negative for one-sheet hyperboloids and zero for planes. For a region on the surface constructed in this thesis, positive Gaussian curvatures correspond to areas that are locally convex; while negative Gaussian curvatures correspond to saddle points. When used for surface interrogation, Gaussian curvature values are mapped to colors and color plots are generated for the surface constructed.

To generate Gaussian curvature plots, each surface is tessellated into a set of small triangles. For each vertex of such a triangle, its Gaussian curvature value is calculated and mapped to a color. In my curvature plots, red regions indicate positive Gaussian curvature; blue regions indicate negative Gaussian curvature; and the green regions have zero Gaussian curvature. After each vertex is tagged with a material, a color image is generated for the surface. In general, the Gaussian curvature plot is very sensitive to small local changes and will detect types of defects not found by viewing the shaded images.

As shown in Figure 1.11, Gaussian curvature plots were generated for two surfaces based on the same torus model. Both surfaces are constructed with Loop's sextic scheme and both are G^1 continuous (Chapter 4). The left one is an approximating surface of the given data mesh, which shows a narrower range of curvature values and a smooth curvature distribution. For the interpolation surface on the right of Figure 1.11, we see some high curvature values (red) changing rapidly to small values (green), indicating abrupt changes of the surface shape in this region.

As a trade-off between the C^1/G^1 conditions and shape quality, approximate C^1/G^1 surfaces are expected to have better (at least not worse) shape than surfaces with C^1/G^1 continuity. To compare the shape quality between surfaces, I generated Gaussian curvature plots and identified the local shape imperfections. These local shape defects in turn determine the shape quality of the approximate C^1/G^1 surfaces.

1.4.3 Isophotes curves

The isophote method analyzes surface quality by determining lines of equal light intensity. For a parallel lighting $\vec{L}$, the isophotes condition is defined as

$$\vec{N}(u_0, u_1, u_2) \cdot \vec{L} = I. \tag{1.5}$$

Here I is a constant value; $\vec{N}(u_0, u_1, u_2)$ is the normal vector at the surface point of (u_0, u_1, u_2). All the points with the same value I in Equation 1.5 are connected and form isophotes curves (also called isophotes lines). Silhouettes are special isophotes for $I = 0$. In general, we can construct isophotes curves by testing various values of constant I in Equation 1.5. There also exist more efficient methods to generate isophotes curves with acceleration using GPU programming [2].

If a surface is C^k continuous, then the corresponding isophotes curves are C^{k-1} continuous [16]. For example, the isophotes curves of a C^1/G^1 continuous surface are C^0 continuous, *i.e.*, each curve should be connected. If a surface is constructed to be approximately C^1/G^1 continuous, its isophotes curves are not C^0 continuous, therefore gaps on the curves are visible. The lower the discontinuity we have on the surface, the smaller gaps are on the isophotes curves.

Isophotes curves are closely related to *reflection lines* [39]. Both isophotes and reflection lines give an indication of how reflections will look on the surface. For applications such as car body design and computer animation, it is important to have a surface with continuous

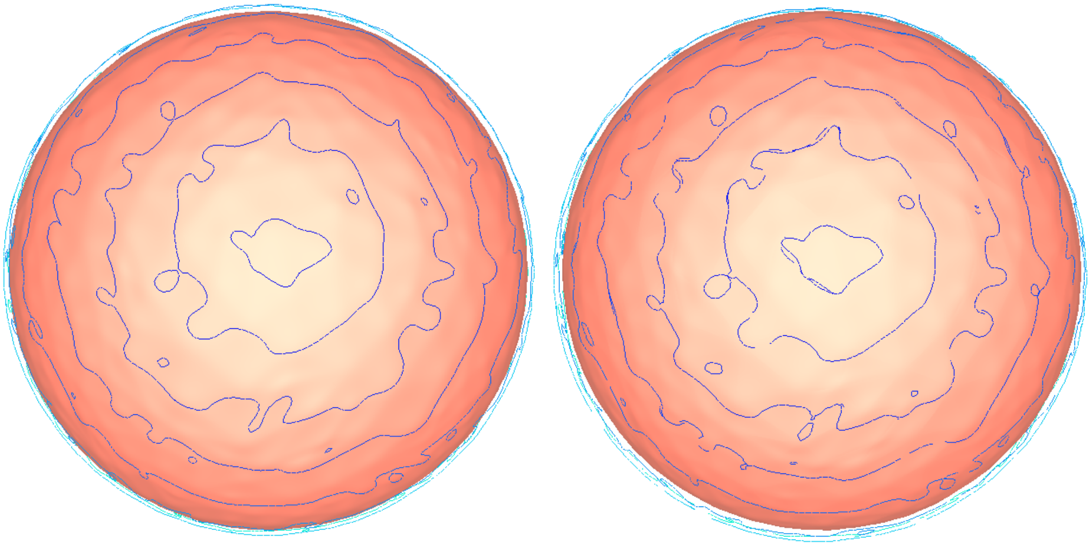


Figure 1.12: Examples of isophotoes curves.

reflections. In this case, isophotes and/or reflection lines are often applied as interrogation tools.

When interrogating the shape quality of ε-C^1/ε-G^1 continuous surfaces, I render isophotes curves together with the shaded surface image. To make the isophotes curves more discernible, they are placed with a small offset from the surface and rendered in different colors. Figure 1.12 shows two interpolating surfaces with isophotes curves (blue) attached. Both surfaces are constructed using the same data mesh sampled on a unit sphere. The left surface of Figure 1.12 is a G^1 continuous surface constructed using Loop's sextic interpolation scheme [25]; the right surface is a quartic approximately G^1 continuous surface (Section 4.3.1). Each of the isophotes curves on the left surface is connected (C^0 continuous), indicating that the surface is G^1 continuous. For the right surface of Figure 1.12, which is only approximately G^1 continuous, the gaps on the isophotes curves indicate that the isophotes curves are not C^0.

Chapter 2

Approximate continuity for functional surfaces

In this chapter, I will introduce the application of approximate continuity to the surface interpolation in the functional case. For functional surfaces, the xy-coordinates of all the control points are distributed uniformly in the xy-plane, with

$$R_{ijk}^{xy} = \frac{i}{n} R_{n00}^{xy} + \frac{j}{n} R_{0n0}^{xy} + \frac{k}{n} R_{00n}^{xy}, \qquad i + j + k = n.$$

Here (i, j, k) are barycentric coordinates; R_{n00}^{xy}, R_{0n0}^{xy} and R_{00n}^{xy} are the xy position of the three vertices. All the control points have different z-coordinates. The example of a cubic patch is shown in Figure 2.1, viewed along the z-axis. The boundary curves of these triangular Bézier patches are defined by the boundary control points, and the cross boundary derivatives are defined by the difference of the first two layers of control points.

For the cubic patch shown in Figure 2.1, the cross boundary derivatives along the boundary from R_{030} to R_{003} in the direction of $R_{003} - R_{030}$ are given by

$$R'(t) = 3 \sum_{j+k=2} \big(R_{0jk+100} - R_{0jk+010}\big) B_{0jk}^{2}(t).$$

As mentioned in Chapter 1, for two Bézier patches of the same degree to meet with C^0 continuity, they must share common boundary control points. If these two patches are functional and also meeting with C^1 continuity, their adjacent panels along the boundary must be coplanar.

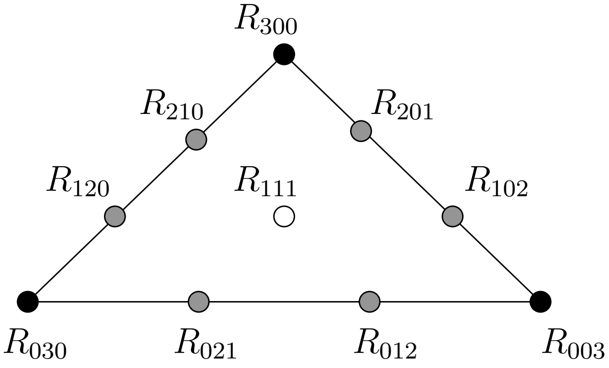


Figure 2.1: Control points of a functional Bézier patch.

2.1 Bounding the discontinuity

In earlier work, the discontinuity between two adjacent triangular Bézier patches was approximated by sampling the normals of the patches along the common boundary, and computing the maximum angle between corresponding pairs of normals [7, 27]. This method has two weaknesses: first, it can be computationally expensive; and second, in general the method calculates a lower bound than an upper bound. The difficulty in computing the true maximum is that the function for the discontinuity is non-polynomial.

In this section, two upper bounds are derived for the discontinuity in the surface normal based solely on the control points of the neighboring patches. To further simplify the upper bounds, the case of cubic patches is analyzed with equal normals at two end points of common boundaries. The case of equal normals at the end points is interesting, since by interpolating the position and derivatives at the corners of a data triangle, schemes can often achieve polynomial precision. Although this polynomial precision may not provide bounds on the discontinuity, it helps to minimize the discontinuity between adjacent patches and results in better shape.

Consider the functional case shown in Figure 2.2, where two cubic Bézier patches F and $\bar{F}$ share the common boundary H (this figure shows the orthographic projection of the control points into the xy-plane; the control points themselves have arbitrary z values). If F and $\bar{F}$ meet with C^1 continuity, then the control points of a pair of boundary control

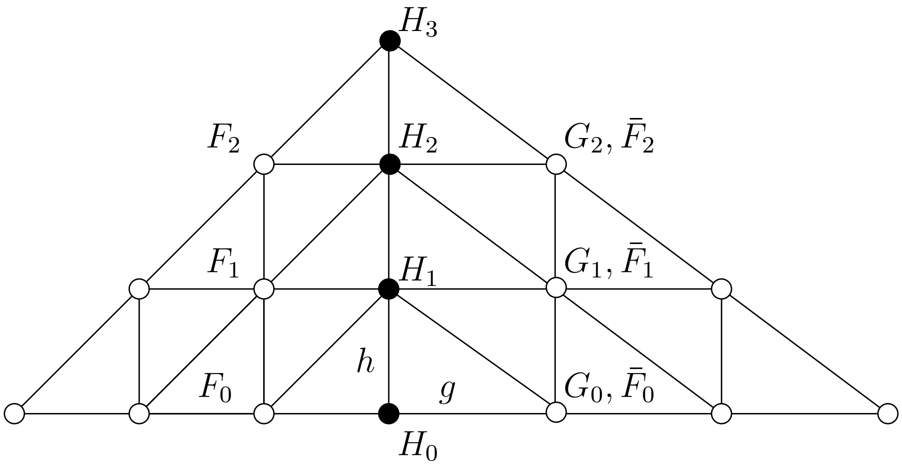


Figure 2.2: Domain of cubic patches F, $\bar{F}$, and G.

panels, say F_i, H_i, $\bar{F}_i$, and H_{i+1}, are coplanar ($i = 0, 1, 2$) (Section 1.2.1). Since this is the functional setting, the spacing between consecutive H_i^{xy} is a constant h, and the spacing between each $\bar{F}_i^{xy}$, H_i^{xy} pair is a constant g. The normal vector $\vec{N}$ for patch $\bar{F}$ (also the normal of patch F) is calculated as the cross product of the two directional derivatives in the x and y domain directions:

$$\vec{N} = \left(n \sum_{i=0}^{n-1} (H_{i+1} - H_i) B_i^{n-1}(t) \right) \times \left(n \sum_{i=0}^{n-1} (\bar{F}_i - H_i) B_i^{n-1}(t) \right). \tag{2.1}$$

Here $\bar{F}$ and F are defined to be in degree n. To simplify the computation, we assume that the boundary H in the domain is parallel to the y-axis, and that the bottom edges of F and $\bar{F}$ in the domain are parallel to the x-axis (in Section 2.2, if the patches have arbitrary domain triangles, a re-parameterization on those patches will yield the arrangement I have just described).

Letting

$$\vec{H}_i = H_{i+1} - H_i, \qquad \vec{\bar{F}}_i = \bar{F}_i - H_i,$$

then

$$\vec{H}_i = 0\vec{x} + h\vec{y} + \vec{H}_i^z \vec{z}, \qquad \vec{\bar{F}}_i = g\vec{x} + 0\vec{y} + \vec{\bar{F}}_i^z \vec{z}.$$

The normal $\vec{N}$ is calculated as

$$\begin{aligned}
\vec{N} &= \left(n \sum_{i=0}^{n-1} [0, h, \vec{H}_i^z] B_i^{n-1}(t) \right) \times \left(n \sum_{j=0}^{n-1} [g, 0, \vec{\bar{F}}_j^z] B_j^{n-1}(t) \right) \\
&= n^2 \left[h \sum_{j=0}^{n-1} \vec{\bar{F}}_j^z B_j^{n-1}(t), g \sum_{i=0}^{n-1} \vec{H}_i^z B_i^{n-1}(t), -hg \right] \qquad (2.2)
\end{aligned}$$

Now consider a patch G over the same domain as $\bar{F}$ that shares the H_i boundary control points with F, but whose control points differ from $\bar{F}$ elsewhere. We can think of the G_i as being displaced from the $\bar{F}_i$ along the $\vec{z}$-axis: $G_j^z = \bar{F}_j^z + z_j$. Then the equation for the normal $\vec{N'}$ of patch G along the boundary H is (based on Equation 2.2)

$$\begin{aligned}
\vec{N'} &= n^2 \left[h \sum_{j=0}^{n-1} (\vec{\bar{F}}_j^z + z_j), g \sum_{i=0}^{n-1} \vec{H}_i^z, -hg \right] B_i^{n-1}(t) B_j^{n-1}(t) \\
&= \vec{N} + n^2 \sum_{j=0}^{n-1} [hz_j, 0, 0] \, B_j^{n-1}(t). \qquad (2.3)
\end{aligned}$$

Note that both $\vec{N}$ and $\vec{N'}$ are not normalized. Letting $I = n^2 \sum_{j=0}^{n-1} [hz_j, 0, 0] B_j^{n-1}(t)$, then $\vec{N'} = \vec{N} + \vec{I}$. The relationship among $\vec{I}$, $\vec{N}$ and $\vec{N'}$ is shown in Figure 2.3(a). In the following sections, I will describe two methods to bound the angle between N and N'.

2.1.1 Upper bound one

The angle θ between N and N' is defined as

$$\begin{aligned}
\theta &= \arcsin\left(\frac{|\vec{N} \times \vec{N'}|}{|\vec{N}||\vec{N'}|} \right) = \arcsin\left(\frac{|\vec{N} \times (\vec{N} + \vec{I})|}{|\vec{N}||\vec{N'}|} \right) = \arcsin\left(\frac{|\vec{N} \times \vec{I}|}{|\vec{N}||\vec{N'}|} \right) \\
&= \arcsin\left(\frac{n^4 h^2 \left| \sum_{j=0}^{n-1} z_j B_j^{n-1}(t) \left[0, g, -\sum_{i=0}^{n-1} \vec{H}_i^z B_i^{n-1}(t) \right] \right|}{|\vec{N}||\vec{N'}|} \right). \qquad (2.4)
\end{aligned}$$

It is difficult to calculate a true maximum value for θ using Equation 2.4. Instead of bounding θ as a whole term, it is easier to calculate an upper bound by finding the maximum/minimum value for each part in the numerator and denominator separately.

Theorem 2.1.1. *For functional triangular Bézier patches F and G arranged as in Figure 2.2, an upper bound on the angle θ between the normals of the two patches along their common boundaries is given by*

$$\theta \leq \arcsin\left(\frac{n^4 h^2 L_z \sqrt{g^2 + (L_h)^2}}{L_{\vec{N}} L_{\vec{N}'}}\right), \tag{2.5}$$

where

$$L_z = \max\left(\Big|\sum_{j=0}^{n-1} z_j B_j^{n-1}(t)\Big|\right), \quad L_h = \max\left(\Big|\sum_{i=0}^{n-1} \vec{H}_i^z B_i^{n-1}(t)\Big|\right).$$

Here $L_{\vec{N}}$ and $L_{\vec{N}'}$ denote the minimum length of $\vec{N}$ and $\vec{N}'$ as we vary t over $[0, 1]$.

Proof. The proof of this theorem is given by bounding the components in Equation 2.4 separately. Since L_z and L_h are degree $n-1$ Bézier curves, taking the maximum absolute value of z_j and $\vec{H}_i^z$ will generate the maximum values for L_z and L_h(see Equation 2.2). We bound $L_{\vec{N}}$ as

$$\begin{aligned} L_{\vec{N}} &\geq \Big| n^2 \left[h \min \Big| \sum_{j=0}^{n-1} \vec{\bar{F}}_j^z B_j^{n-1}(t) \Big|, g \min \Big| \sum_{i=0}^{n-1} \vec{H}_i^z B_i^{n-1}(t) \Big|, -hg \right] \Big| \\ &\geq n^2 \Big| \left[h \,\mathrm{Zmin}(\bar{F}_j^z), g \,\mathrm{Zmin}(\vec{H}_i^z), -hg \right] \Big|, \end{aligned} \tag{2.6}$$

where $\mathrm{Zmin}(a_i)$ is zero if the a_i are of mixed sign, and is the minimum of the absolute values of the a_i otherwise. Similarly, the definition of $L_{\vec{N}'}$ is

$$L_{\vec{N}'} \geq n^2 \big| [h \,\mathrm{Zmin}(\bar{G}_j^z), g \,\mathrm{Zmin}(\vec{H}_i^z), -hg] \big|.$$

□

If patches F and G are both cubic triangular Bézier patches, and the panels at each end of the boundary are coplanar, then the control points of these end panels have zero z displacement, *i.e.*, $G_0 = \bar{F}_0$ and $G_2 = \bar{F}_2$, and $z_0 = z_2 = 0$. Thus, the only freedom to manipulate is G_1. The upper bound defined by Equation 2.5 simplifies to

$$\theta \leq \arcsin\left(\frac{n^4 h^2 |z_1| \sqrt{g^2 + (L_h)^2}}{2 L_{\vec{N}} L_{\vec{N}'}}\right). \tag{2.7}$$

Here the extra factor of 2 in the denominator is because the Bernstein polynomial $B_j^2(t)$ obtains its maximum value over [0, 1] at $t = \frac{1}{2}$, and $B_j^2(\frac{1}{2}) = \frac{1}{2}$. In the remaining part of this thesis, I use the acronym of FB1 (functional surface upper bound 1) to denote the upper bound defined by Equation 2.5.

2.1.2 Upper bound two

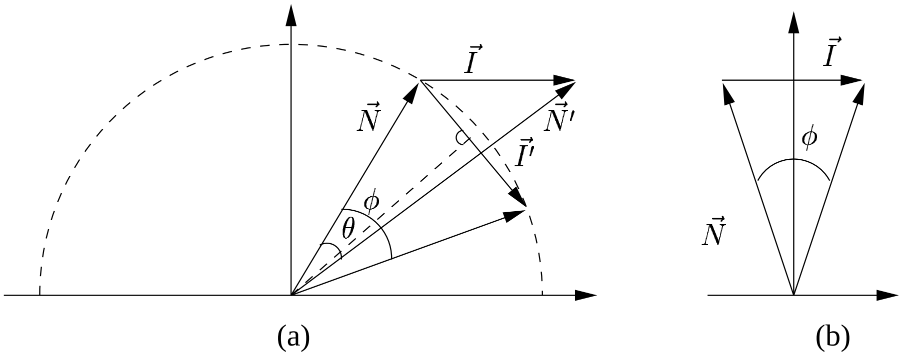


Figure 2.3: Calculating a bound on θ.

Another upper bound of the angle θ is calculated by using a different method to bound the surface normals.

Theorem 2.1.2. *For functional triangular Bézier patches F and G arranged in Figure 2.2 and discussed in the text, the bound on the angle θ between the normals of the two patches along their common boundaries is given by*

$$\theta \leq \arcsin\left(\frac{hn^2L_z}{2L}\right), \tag{2.8}$$

where L_z is defined as in method 1 and

$$L = min(L_{\vec{N}}, L_{\vec{N'}}).$$

Proof. The goal is to bound the angle θ between $\vec{N}$ and $\vec{N'}$ for $t \in [0,1]$. For now, assume that $|\vec{N}| \leq |\vec{N'}|$. As illustrated in Figure 2.3(a), I' is the vector that forms a chord on the circle of radius $|\vec{N}|$, where its tail is placed at the head of $\vec{N}$ with $|I| = |I'|$, and $I \cdot I' \geq 0$. Looking at Figure 2.3(a), we see that $\theta \leq \phi$. For any $\vec{N}$ and $\vec{I}$ of fixed length, angle ϕ reaches its maximum value when $|\vec{N}| = |\vec{N'}|$ (Figure 2.3(b)). The angle θ can be bounded by computing a bound on ϕ:

$$\begin{aligned} \theta \leq \phi &\leq 2 \arcsin\left(\frac{|\vec{I}|}{2L}\right) \\ &= 2 \arcsin\left(\frac{n^2 \left|\sum_{j=0}^{n-1} [hz_j, 0, 0] B_j^{n-1}(t)\right|}{2L}\right) \\ &= 2 \arcsin\left(\frac{hn^2 L_z}{2L}\right). \end{aligned} \tag{2.9}$$

□

Now we consider the special case of two cubic Bézier patches F and G where the panel pairs at each end of the boundary are coplanar. As mentioned in method 1, we have $G_0 = \bar{F}_0$ and $G_2 = \bar{F}_2$, therefore $z_0 = z_2 = 0$. Equation 2.8 simplifies to

$$\theta \leq 2 \arcsin\left(\frac{|h9 \sum_{j=0}^{2} z_j B_j^2(t)|}{2L}\right) \leq 2 \arcsin\left(\frac{9h|z_1|}{4L}\right). \tag{2.10}$$

Again, the Bernstein polynomial $B_j^2(t)$ obtains its maximum over [0, 1] at $t = \frac{1}{2}$, and $B_j^2(\frac{1}{2}) = \frac{1}{2}$. For any approximately C^1 continuous surface, we can use the minimum result of the two methods as a bound of the discontinuity angle. Let FB2 denote the upper bound defined by Equation 2.8.

2.2 Data fitting scheme

To test the two upper bounds, I devised a surface fitting scheme and sampled the Franke functions as test data [22]. By varying the sampling density, a set of ε-C^1 surfaces are constructed with a small amount of C^1 discontinuities. The following ε-C^1 construction

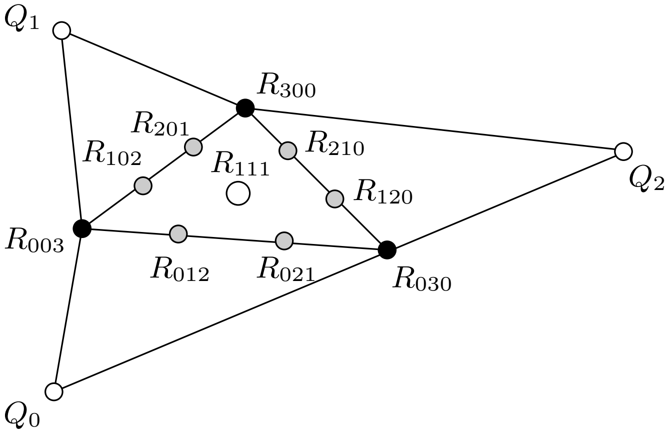


Figure 2.4: Construction of the functional C^1 scheme.

is the first step of a scheme that appeared in [29]. Three types of control points are constructed: data points, boundary points and one interior point. For a data triangle $\triangle P_0P_1P_2$, we construct a cubic Bézier patch R (shown in Figure 2.4) to interpolate the position and the normal at each of three vertices:

1. Set the three corner data points (black points in Figure 2.4) to the data vertices:

$$R_{300} = P_0,\ R_{030} = P_1,\ R_{003} = P_2.$$

2. For the other six boundary control points (gray points in Figure 2.4), we calculate the intersection of a line parallel to the z-axis with the tangent plane of the nearest corner point. For example, to compute R_{210}, we intersect the line through R_{210}^{xy} in the direction of the z-axis with the tangent plane at R_{300}. The intersection point is the position for R_{210}. This forces the patch to interpolate the normal data of data vertices, and ensures that all the control points surrounding a data point are coplanar.

3. Set the interior control point R_{111}(the big white point in Figure 2.4) by evaluating R at the three adjacent data vertices, say Q_0, Q_1 and Q_2, and averaging the results.

> This step is the same as the one used by Foley-Opitz to set the center point in their hybrid patch construction [13].

For the data triangle $\triangle P_0 P_1 P_2$ with adjacent data points Q_0, Q_1 and Q_2 (Figure 2.4), the last step of my construction needs to calculate the z-coordinate of the interior control point R_{111}.

We compute three values for R^z_{111}: R^z_0, R^z_1, and R^z_2 by evaluating the patch R at the three Q^{xy}_s values, and require $R(Q^{xy}_s) = Q^z_s$, $s = 0, 1, 2$. Each evaluation provides a linear equation in the unknown control point R_{111}. Solving each equation gives one of the R^z_s, which are averaged to set R^z_{111}. Letting (t^0_i, t^1_i, t^2_i) be the barycentric coordinates of Q^{xy}_s relative to $\triangle P^{xy}_0\, P^{xy}_1\, P^{xy}_2$, evaluating R at Q^{xy}_s gives us

$$\begin{aligned}
Q^z_s &= \sum_{i+j+k=3} R^z_{ijk} B^3_{ijk}((t^0_i, t^1_i, t^2_i)) \\
&= R^z_i B^3_{111}(t^0_i, t^1_i, t^2_i) + \sum_{ijk, ijk \neq 111} R^z_{ijk} B^3_{ijk}((t^0_i, t^1_i, t^2_i)) \\
\Longrightarrow R^z_s &= \frac{Q^z_s - \sum_{ijk, ijk\neq 111} R^z_{ijk} B^3_{ijk}((t^0_i, t^1_i, t^2_i))}{B^3_{111}(t^0_i, t^1_i, t^2_i)}, \\
R^z_{111} &= \frac{R^z_0 + R^z_1 + R^z_2}{3}.
\end{aligned}$$

Note that if all six data points and normals (P_0, P_1, P_2, Q_0, Q_1, Q_2) come from a single cubic polynomial $\mathcal{C}$, then $R_0 = R_1 = R_2 = R_{111}$ and the resulting surface patch will reproduce this polynomial, *i.e.*, $R = \mathcal{C}$.

The cubic data fitting scheme introduced above meets the requirement for generating the test data; in particular, the cubic convergence results in decreasing discontinuities as we increase the sampling density of the base functions. With this scheme, I would need a special construction for the boundary data triangles, because they have only one or two neighboring triangles rather than the three neighbors that internal data triangles have. While this is easily done by constructing one or two R^z_i respectively, to simplify the analysis, I do not construct patches for the boundary layer of triangles.

To use the approximate continuity result of Section 2.1, the control points of two adjacent patches F and G must be arranged as in Figure 2.2 (*i.e.*, the common boundary H must be aligned with the y-axis, and another boundary must align with the x-axis). If

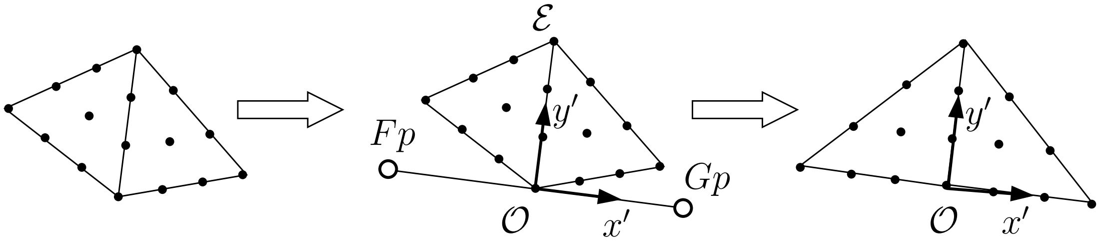


Figure 2.5: Frame change for the approximate continuity calculation.

the patch domain does not meet these constraints, we can change the patch representation as follows (see Figure 2.5):

1. Create a coordinate frame $\mathcal{F} = \{\mathcal{O}, x', y', z'\}$, with the origin $\mathcal{O}$ at one end of the common boundary, and $\mathcal{E}$ for the other end of the boundary; align the y' basis vector with the common boundary; set the x' basis vector to be perpendicular to y' in the xy-plane; and set the z' basis vector to be parallel to the z-axis.
2. Position two points Fp and Gp in the domain along the x'-axis, with $|\mathcal{O} - Fp| = |\mathcal{O} - Gp| = |\mathcal{E} - \mathcal{O}|$.
3. Re-parameterize F by evaluating F at Fp using the de Casteljau algorithm (Section 1.1.2), and extracting the sub-patch over the domain $\triangle Fp\mathcal{O}\mathcal{E}$. A similar process can be used to extract the representation of G over the domain triangle $\triangle Gp\mathcal{E}\mathcal{O}$.

Now we can apply the continuity bounds described in Section 2.1, using the coordinates of the control points relative to $\mathcal{F}$. The $\bar{F}_i$ of Figure 2.2 will have the same xy-positions as the G_i but different z-values, so that patch $\bar{F}$ meets patch F with C^1 continuity. The points $\bar{F}_0$ and $\bar{F}_2$ already meet the required C^1 planarity conditions; point $\bar{F}_1$ is given by

$$\bar{F}_1 = t_0 H_1 + t_1 H_2 + t_3 F_1,$$

where (t_0, t_1, t_2) are the barycentric coordinates of G_1^{xy} relative to $\triangle H_1^{xy}\, H_2^{xy}\, F_1^{xy}$.

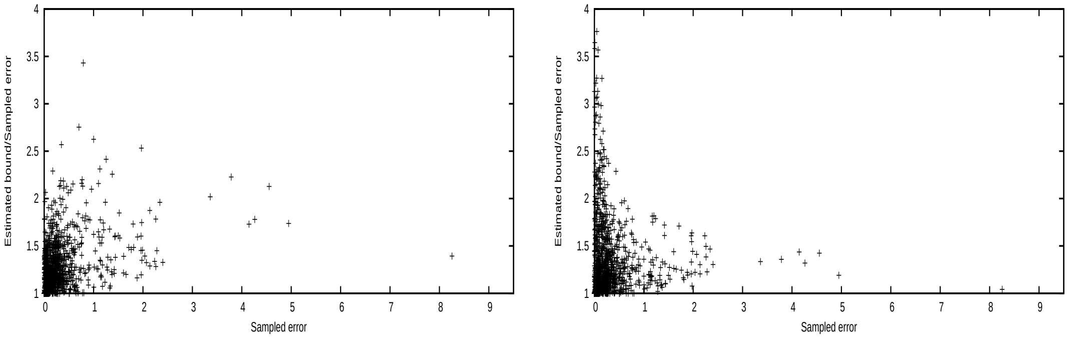
Table 2.1: Discontinuity of Franke function surfaces.

Functions	Grid	Discontinuity	FB1	FB2
F1	10×10	26.487492	67.894637	35.814840
	20×20	8.254011	11.494557	8.597684
	40×40	1.226258	1.469088	1.266205
F2	10×10	8.059139	13.315756	11.347997
	20×20	2.676044	3.660120	3.431178
	40×40	0.438722	0.515085	0.541983
F3	10×10	1.430003	1.630511	1.498616
	20×20	0.231934	0.240315	0.235121
	40×40	0.030659	0.030940	0.030764
F4	10×10	0.512760	0.545877	0.519773
	20×20	0.069462	0.070669	0.069708
	40×40	0.008862	0.008901	0.008870
F5	10×10	5.426008	9.497992	6.355084
	20×20	0.988369	1.235251	1.039557
	40×40	0.137503	0.146953	0.139415
F6	10×10	0.432433	0.503140	0.506612
	20×20	0.081190	0.089400	0.097062
	40×40	0.013420	0.014184	0.016166

2.3 Tests and results

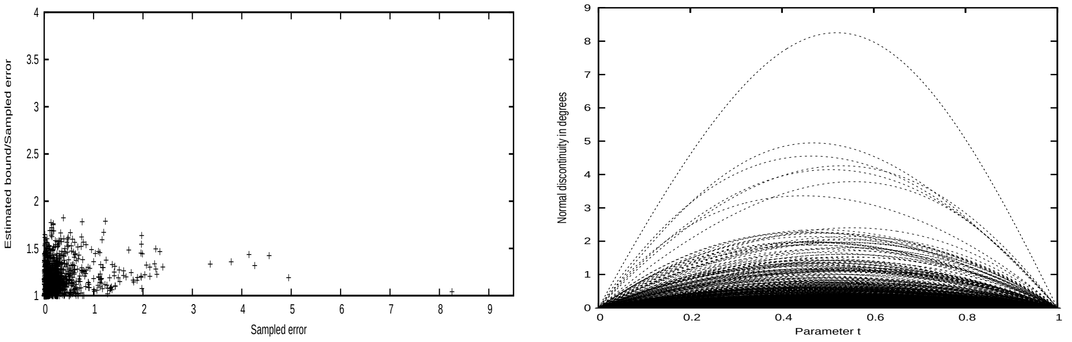
For test data, I sampled the six “Franke functions” as described by Grandine [14], and fit approximate C^1 surfaces to these data sets using the scheme described in Section 2.2. Since we are testing the boundary discontinuity and the convergence properties as the sampling rate is increased, these six functions are sampled uniformly in the xy-plane.

The discontinuity between surface normals are bounded using three methods. The first method samples the common boundary of every pair of adjacent patches at 100 points, computing the surface normals, and calculating the angle between the two normals. The



(a) Bound 1 over sampled error ratio.

(b) Bound 2 over sampled error ratio.



(c) Minimum Bound over sampled error ratio.

(d) Discontinuity curves.

Figure 2.6: Upper bounds analysis of Franke function 1 surface.

other two methods, FB1 and FB2, calculate the upper bounds as described in Section 2.1. Table 2.1 shows the maximum discontinuity angles and the two upper bounds for all boundary curves of the surface; all the numbers are in degrees.

The column labeled “Grid” in Table 2.1 refers to the number of cells in the square grid to fit patches on; two triangular patches were fit to each square of the grid. Since no patch was fit to the boundary layer of squares, the actual number of samples taken from the corresponding Franke function was three higher in each dimension (e.g., a 10×10 grid means 13×13 samples). The samples were taken so that (ignoring the boundary layer) the outer layer aligns with the $[0, 1]$-square in the domain. As shown in Table 2.1, although the discontinuity of my scheme is noticeable at low samplings of the Franke function, the discontinuity drops quickly when sampling density is increased; the discontinuity appears

Table 2.2: Ratio distribution for Franke Function 1.

Ratio	1.0-1.5	1.5-2.0	2.0-2.5	2.5-3.0	3.0-3.5	3.5-
Method 1	84.43	12.76	1.98	0.34	0.09	0.0
Method 2	77.33	15.86	4.57	1.29	0.60	0.34
Combined	94.83	5.17	0.0	0.0	0.0	0.0

to drop as $O(h^3)$. Generally, FB2 shows better results than FB1. The plot of the normal discontinuities for all the patch boundaries (Figure 2.6(d)) shows that the maximum discontinuity occurs near the middle of the curve but not exactly at $t = 0.5$.

When I plot the ratio of the estimated discontinuity to the maximum sampled discontinuity for all boundaries of one surface, we see that the ratio can be higher than the ratio of maximum values in Table 2.1. Plots were generated for Function 1, on a 20×20 grid (Figure 2.6(a,b,c)). Each point on the graph is the ratio of the error bound to the numerically computed maximum error for the boundary between two pairs of patches. Using FB1 (Figure 2.6(a)), we generate more boundary curves with a lower ratio than using FB2 (Figure 2.6(b)), especially for those with lower discontinuity angles. But FB2 is better for boundary curves where there is a high normal discontinuity. In an application, we can take the smaller result of the two bounds (Figure 2.6(c)). Table 2.2 shows the percentage breakdown of upper bound ratios for function 1. Table 2.3 shows the percentage distribution for other functions. All the surfaces in Table 2.2 and Table 2.3 were sampled on a 20×20 grid. For the combination of the two methods, the majority of the points are in the lower end of the ratio (Figure 2.6(c)).

Looking at the shaded images, the shape quality of ϵ-C^1 surfaces seemed good, and I wanted to compare it to an existing C^1 surface fitting scheme. Since a local polynomial scheme with cubic convergence would provide the most meaningful comparison, the new ϵ-C^1 scheme was compared to a cubic precision Clough-Tocher interpolant [28]. The visual results of these comparisons are shown in Figures 2.7(a) to 2.7(h).

Figure 2.7(a), left, shows the surface produced when my scheme is applied to a 5×5 sampling of the Franke 1 function. The C^1 discontinuities are clearly visible in the surface (the maximum sampled normal discontinuity is 33.96 degrees). Note, however,

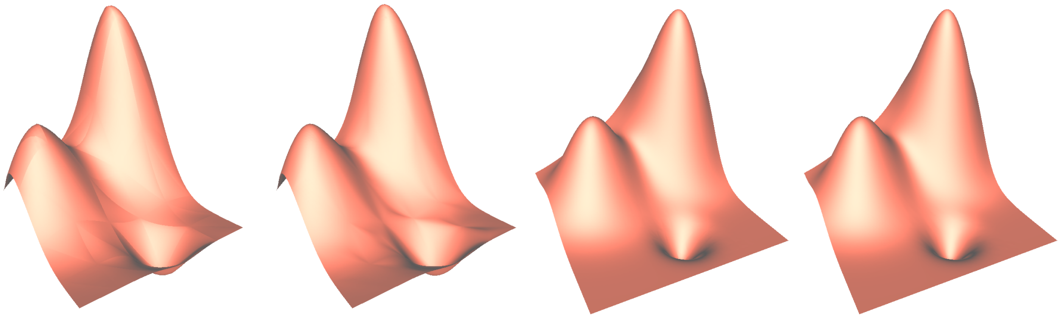
Table 2.3: Ratio distribution for other Franke functions.

	Ratio	1.0-1.1	1.1-1.2	1.2-1.3	1.3-1.4	1.4-1.5	1.5-
F2	Method 1	71.72	5.86	9.48	6.21	0.00	6.72
	Method 2	74.66	8.97	9.48	6.55	0.00	0.34
	Combined	74.66	8.97	16.03	0.0	0.00	0.34
F3	Method 1	90.17	9.57	0.26	0.00	0.00	0.00
	Method 2	71.43	17.84	8.71	3.02	0.00	0.00
	Combined	96.12	3.88	0.00	0.0	0.00	0.00
F4	Method 1	100.0	0.00	0.00	0.00	0.00	0.00
	Method 2	68.10	31.90	0.00	0.00	0.00	0.00
	Combined	100.0	0.0	0.00	0.00	0.00	0.00
F5	Method 1	68.97	18.97	7.41	3.28	1.03	0.34
	Method 2	72.24	10.34	6.90	4.48	1.90	4.14
	Combined	78.45	15.34	4.48	1.38	0.34	0.0
F6	Method 1	96.03	3.79	0.17	0.00	0.00	0.00
	Method 2	75.00	21.38	2.24	0.86	0.34	0.17
	Combined	99.31	0.69	0.0	0.0	0.0	0.0

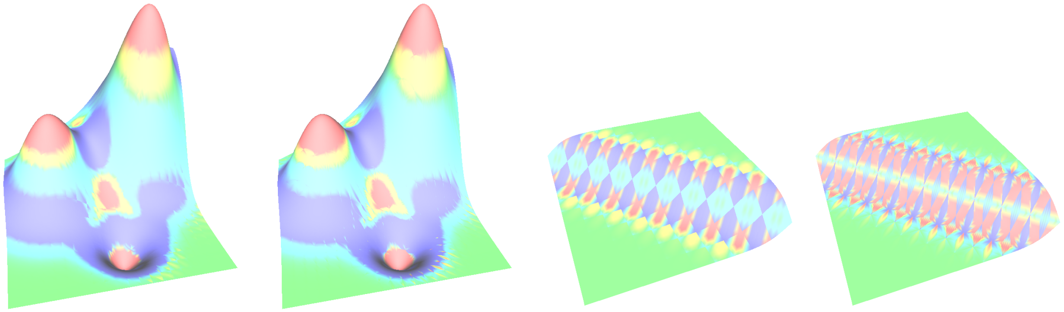
that the cubic precision Clough-Tocher interpolant has similar artifacts in the same location (Figure 2.7(a), right). Increasing the sampling density of the Franke 1 function to 20 × 20, both my methods and the Clough-Tocher method produce good results. Looking at Gaussian curvature plots of the surfaces fit to the 20×20 sampling of the Franke 1 function (Figure 2.7(c)), the approximate C^1 scheme gives a slightly smoother curvature plot than the C^1 Clough-Tocher scheme. (In Figures 2.7(c) to 2.7(h), the left image is the Gaussian curvature of the surface created by my ϵ-C^1 scheme; the right from the cubic precision Clough-Toucher scheme).

The remaining figures show a Gaussian curvature comparison of my ϵ-C^1 scheme (left of each figure) to the cubic precision Clough-Tocher scheme (right of the figure) on Franke functions 2, 3, 4, 5, and 6. For the Franke functions 3–6, both schemes construct a reasonable surface for 10x10 samplings. For the Franke 2 function, both schemes did a relatively poor job of fitting a surface to the data. The maximum/minimum Gaussian curvature of the surface fit by my scheme to this data is $(23.3392, -48.035)$; the maximum/minimum curvature of the Clough-Tocher surface is $(71.5072, -97.6572)$. This Franke surface is a ruled surface, so its Gaussian curvature is 0 everywhere. The Franke 2 function is known to be difficult to fit with polynomials because of ringing effects.

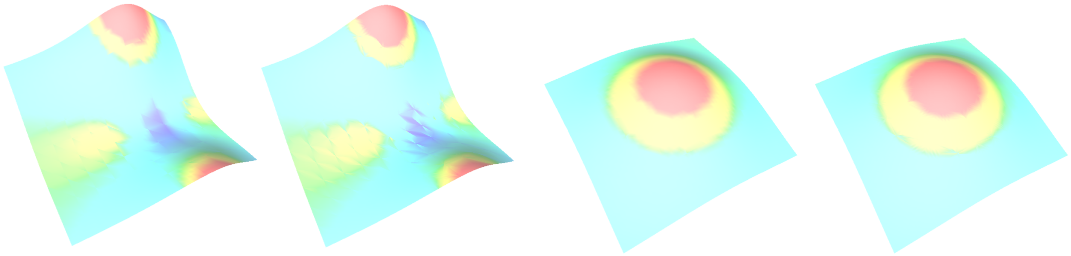
In summary, the results indicate that the discontinuity bound is reasonable (although not excellent), and that ε-C^1 surfaces are of comparible quality to C^1 surfaces. Note also that the surface constructed by the Clough-Tocher scheme has three times as many Bézier patches as the ϵ-C^1 surface.



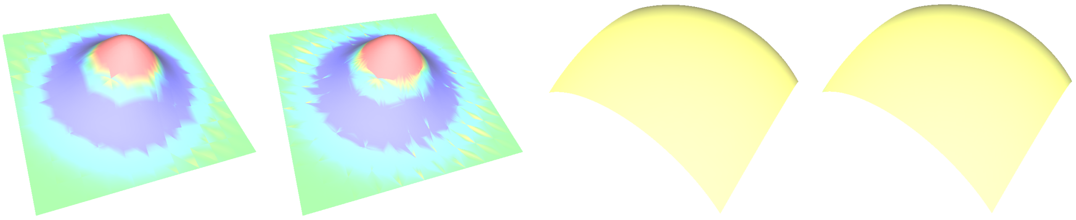
(a) Franke function 1, sampled at 5 × 5. (b) Franke function 1, sampled at 20 × 20.



(c) Franke function 1, sampled at 20 × 20. (d) Franke function 2, sampled at 10 × 10.



(e) Franke function 3, sampled at 10 × 10. (f) Franke function 4, sampled at 10 × 10.



(g) Franke function 5, sampled at 10 × 10. (h) Franke function 6, sampled at 10 × 10.

Figure 2.7: Results of surfaces fit on Franke functions.

Chapter 3

Parametric surface fitting with domain splitting

To interpolate a data set with arbitrary topology, we need to use parametric triangular Bézier patches instead of functional patches. Generally, parametric interpolating surfaces are required to have at least G^1 continuity [11, 34, 33]. However, fulfilment of these conditions is sometimes impossible. In this chapter, I introduce an ϵ-G^1 parametric scheme that performs a Clough-Tocher like 3-to-1 split using cubic parametric patches. By relaxing the continuity constraints to ϵ-G^1, I am able to use low degree (cubic) patches to construct surfaces with smoothness similar to existing G^1 schemes. In my ϵ-G^1 scheme, two neighbouring patches are also required to have equal normals at interior points on the common boundary curve to reduce the normal discontinuity. I begin this chapter with a review of the problem that cubic patches have for constructing a G_1 join in the parametric setting.

3.1 G^1 singularity for cubic patches

As shown in Figure 3.1, two cubic patches F and G meet with G^1 continuity along a common boundary curve H. Here only the boundary curve control points and the second row of control points of F and G are shown. For a point P on the boundary curve, we calculate the tangent planes of F and G, on which we choose four points represented by

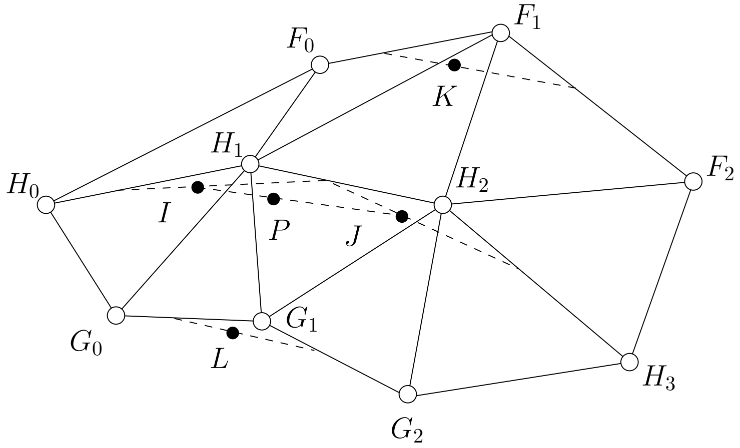


Figure 3.1: G^1 constraints on control points.

quadratic Bézier functions:

$$I = \sum_{i=0}^{2} H_i B_i^2(t), \quad J = \sum_{i=0}^{2} H_{i+1} B_i^2(t), \quad K = \sum_{i=0}^{2} F_i B_i^2(t), \quad L = \sum_{i=0}^{2} G_i B_i^2(t).$$

The tangent plane for patch F at point P is defined by $\triangle KIJ$, and the tangent plane for G is defined by $\triangle IJL$, as shown in Figure 3.1. As the parameter t varies from 0 to 1, the two tangent planes are evaluated along the common boundary H. Piper developed the G^1 continuity constraints [34] for F and G: the four points should be co-planar at any t value, which can be expressed in the matrix form as

$$Det \begin{pmatrix} 1 & I \\ 1 & J \\ 1 & K \\ 1 & L \end{pmatrix} = 0. \tag{3.1}$$

To interpolate the normal vectors at two data points H_0 and H_3, two pairs of end control panels ($\triangle H_0 H_1 F_0$ and $\triangle H_0 H_1 G_0$, $\triangle H_2 H_3 F_2$ and $\triangle H_2 H_3 G_2$) are set to be coplanar. The question is where to place F_1 and G_1 so that Equation 3.1 is satisfied at an arbitrary point on the boundary curve. Piper [34] and Peters [33] both showed that for some data

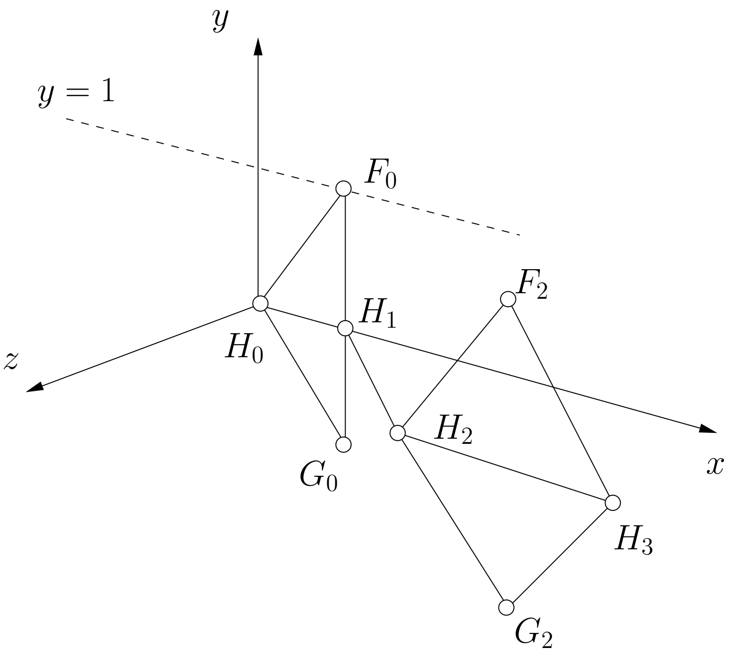


Figure 3.2: A data set without cubic solution.

sets, there might be no G^1 solution when using cubic patches. Piper gave such a singular example [34] as shown in Figure 3.2, with the control points set as follows:

$$\begin{aligned}
F_0 &= (1,1,0), & F_1 &= ?, & F_2 &= (3,1,1), & & \\
H_0 &= (0,0,0), & H_1 &= (1,0,0), & H_2 &= (2,0,1), & H_3 &= (4,0,1), \\
G_0 &= (1,\text{-}1,0), & G_1 &= ?, & G_2 &= (3,\text{-}1,1).
\end{aligned}$$

For this setting of H_0, H_1, H_2, H_3, F_0, F_2, G_0, and G_2, there is no setting for F_1 and G_1 to get a G^1 continuous join between F and G. Further, Peters' work shows that there will be no cubic G^1 solution as long as F_0 is on the line of $y = 1$ in xoz plane, except for $x = 0$, shown as a dashed line in Figure 3.2. Unmentioned by Peters is that as F_0 approaches the line of $y = 1$, a G^1 solution exists, but the resulting surface may have poor shape quality. In Figure 3.3, three example surfaces with two patches F and G were rendered with different positions of F_0: from left to right, the y-coordinate of F_0 is changed to 0.5, 0.95, and 0.99. For each of these three surfaces, F_1 and G_1 are set to the unique positions that yield G^1

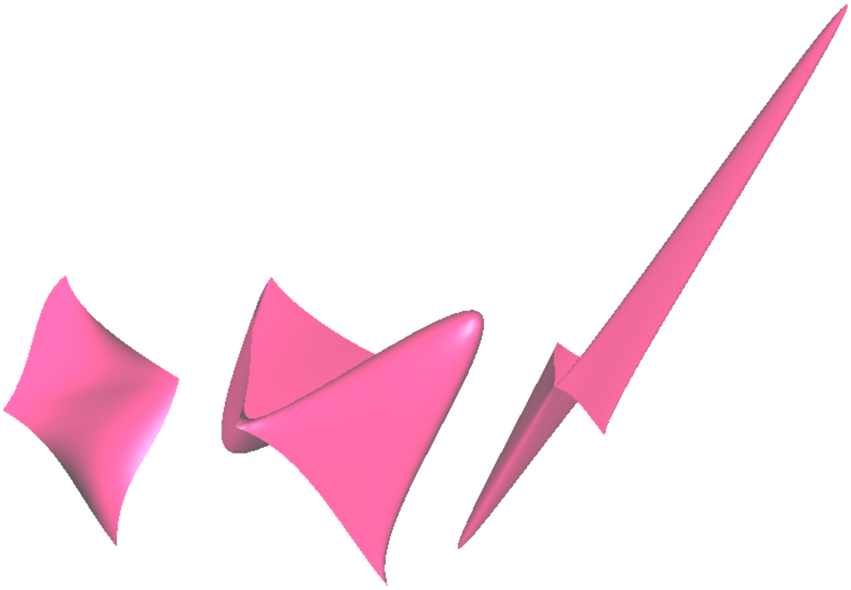


Figure 3.3: Patches of near singular configurations.

continuity along the boundary curve. As shown in Figure 3.3, when F_0 approaches the "singular line" of $y = 1$ along the positive y-axis, F_1 and G_1 move toward infinity and the surface shape gets worse. Finally the centre of the surface goes to infinity when $F_0^y = 1$. If F_0^y approaches 1 from the direction of negative y, a similar trend is also observed.

For the surfaces shown in Figure 3.3, the value of F_0^x was 1. To check the surface shape for other values of F_0^x, I set F_0^x by sampling uniformly in a rectangular area perpendicular to z-axis with the center of $(0, 1, 0)$, excluding the singularity positions. For each sampled position of F_0, I calculated G^1 solutions for F_1 and G_1 and examined the shape quality of the resulting patches. Instead of rendering the resulting patches, I estimated the shape quality by plotting the positions of F_1 and G_1.

I first solved F_1 and G_1 symbolically with F_0 as an invariant. As indicated by Equation 3.1, the determinant of the matrix composed of four tangent points should be zero for any t value to guarantee the co-planarity of the two tangent planes. Expanding Equation 3.1 using Maple, we have

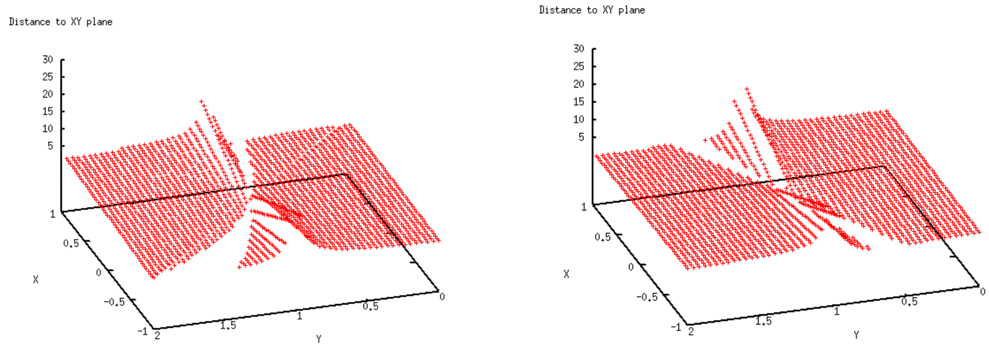
$$Det\begin{pmatrix} 1 & I \\ 1 & J \\ 1 & K \\ 1 & L \end{pmatrix} = C_0t^6 + C_1t^5 + C_2t^4 + C_3t^3 + C_4t^2 + C_5t,$$

$$\begin{aligned}
C_0 &= 8F_1^yG_1^x - 4G_1^x - 4F_0^yG_1^x - 2F_0^yG_1^z - 2G_1^z + 4F_1^yG_1^z - 16F_1^y + 8F_0^y + 8, \\
C_1 &= 48F_1^y + 6F_0^yG_1^z + 8G_1^x + 2G_1^z - 24F_1^yG_1^x - 32F_0^y + 16F_0^yG_1^x - 8F_1^yG_1^z - 16, \\
C_2 &= 24F_1^yG_1^x + 8F_1^yG_1^z - 4G_1^x - 4F_1^x - 2F_1^z - 2G_1^z + 2F_0^x - 8F_0^yG_1^z + \\
&\quad 50F_0^y - 52F_1^y - 24F_0^yG_1^x + 16, \\
C_3 &= 8F_1^x - 6F_0^x + 8F_0^yG_1^z + 2G_1^z + 16F_0^yG_1^x - 38F_0^y - 8F_1^yG_1^x - 8F_1^yG_1^z + \\
&\quad 2F_1^z + 24F_1^y - 12, \\
C_4 &= -4F_1^y + 14F_0^y - 4F_1^x - 2F_1^z + 6F_0^x - 6F_0^yG_1^z - 4F_0^yG_1^x + 4F_1^yG_1^z + 4, \\
C_5 &= 8F_1^yG_1^x - 8G_1^x - 4G_1^z + 4F_1^yG_1^z - 16F_1^y + 16.
\end{aligned}$$

Here (F_1^x, F_1^y, F_1^z) and (G_1^x, G_1^y, G_1^z) are Euclidean coordinates of F_1 and G_1. A necessary G^1 condition is $C_i = 0, i = 0, 1 \ldots 5$. From these six equations, F_1 and G_1 can be solved. For each solution of F_1 and G_1, the shape quality of resulting surfaces mainly depends on the x- and z-coordinates of F_1 and G_1. Therefore, I simplified the solutions by setting $G_1^y = -1$ (the same y-coordinate as G_0 and G_2). Finally, we have G^1 solutions for F_1 and G_1 given as

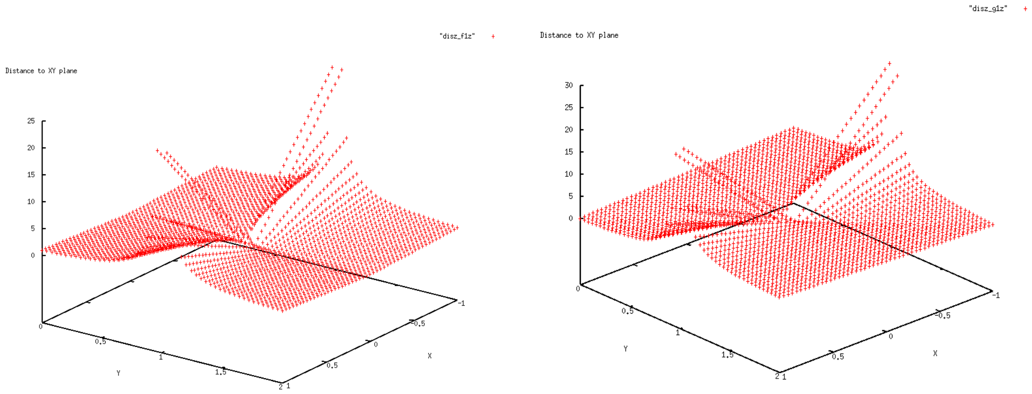
$$\begin{aligned}
F_1 &= \left(F_0^x + \frac{F_0^y + 2}{2} + \frac{F_0^x}{2(F_0^y - 1)}, \quad \frac{F_0^y + 1}{2}, \quad -\frac{F_0^x}{F_0^y - 1}\right), \\
G_1 &= \left(\frac{3}{2} - \frac{F_0^x}{2(F_0^y - 1)}, \quad -1, \quad \frac{F_0^x}{F_0^y - 1} + 1\right).
\end{aligned} \tag{3.2}$$

In Equation 3.2, all unknown coordinates are linear or linear rational functions of F_0^x and F_0^y. For the plots of Figure 3.4, the x-axis is F_0^x, the y-axis is F_0^y and the z-axis is the absolute value of F_1^x, G_1^x, F_1^z and G_1^z individually for each plot. Since the value of F_1^y is a linear function and G_1^y is a constant value, I did not generate plots for them. In



(a) Plot of F_1^x.

(b) Plot of G_1^x.



(c) Plot of F_1^z.

(d) Plot of G_1^z.

Figure 3.4: Plots of coordinates for different configurations.

each plot of Figure 3.4, the large height value indicates that either F_1 or G_1 is far away from the end panels (for F_1^z and G_1^z), or the center control panels are nearly degenerate (for F_1^x and G_1^x), both resulting in poor shape quality. As shown in Figure 3.4, when F_0 approaches the singular line of $y = 1$ (excluding $(0, 1, 0)$), the z-coordinate gets bigger, therefore the surface shape gets worse. If F_0 is set far away from the singular line, the end panel $\triangle H_0H_1F_0$ is nearly degenerate or approaches an extraordinary value, which also results in poor surface shape.

In summary, if we construct cubic patches by generating the boundary control points first, then we cannot always find a G^1 solution for the interior control points. And second, even if there exists a G^1 solution, the resulting surfaces may have poor shape quality. These are the two motivations for using approximate continuity to construct interpolating surfaces.

3.2 Cubic approximate G^1 cross boundary scheme

The possible benefits of using approximate continuity include two aspects:

- Finding good, cubic solutions for the singular or near-singular cases.
- Using approximate continuity to trade off between continuity and shape properties.

To avoid the cubic singular cases, current surface fitting schemes use quartic [38], quintic [18], or sextic [25] patches to guarantee G^1 continuity along the boundary curves.

In this section, I introduce an ϵ-G^1 method using cubic patches, interpolating the position and normal of each data point. As shown in Figure 3.5, two adjacent cubic patches F and G meet at a common boundary curve H. Since the positions and normals at the data points are interpolated, two pairs of end panels are coplanar, and the patches meet with G^1 continuity at H_0 and H_3. For all others points along the boundary, the central panels, *i.e.*, $\triangle F_1H_1H_2$ and $\triangle G_1H_1H_2$, will determine the continuity.

The continuity conditions can be checked by examining the angles between the two normals at a given boundary point. The definitions of Peters [33] are used to illustrated the constraints, as shown in Figure 3.5. The two pairs of end panels, *i.e.*, $\triangle F_0H_0H_1$ and $\triangle G_0H_0H_1$, $\triangle F_2H_2H_3$ and $\triangle G_2H_2H_3$, are set to be coplanar to interpolate the normals at end data points H_0 and H_3. Letting $\overline{F_0G_0}$ intersect $\overline{H_0H_1}$ at P_0, $\overline{F_2G_2}$ intersect $\overline{H_2H_3}$ at P_2, as shown in Figure 3.5, Peters defined the ratios of the intersection as

$$\eta_0 = \frac{|P_0 - H_0|}{|H_1 - H_0|}, \quad \gamma_0 = \frac{|G_0 - P_0|}{|G_0 - F_0|}, \quad \eta_2 = \frac{|P_2 - H_2|}{|H_3 - H_2|}, \quad \gamma_2 = \frac{|G_2 - P_2|}{|G_2 - F_2|}.$$

In each pair of panels, we have

$$\begin{aligned}
\eta_0(H_1 - H_0) &= \gamma_0(F_0 - H_0) + (1-\gamma_0)(G_0 - H_0), \\
\eta_2(H_3 - H_2) &= \gamma_2(F_2 - H_2) + (1-\gamma_2)(G_2 - H_2). \qquad (3.3)
\end{aligned}$$

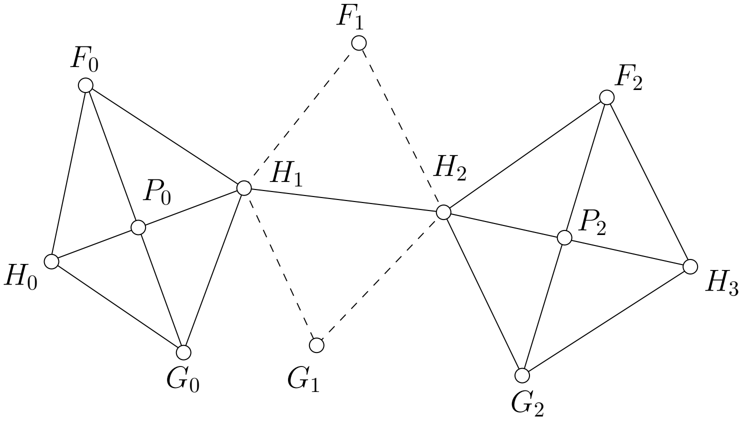


Figure 3.5: Ratios of intersection in control panels.

Peters has shown that there are valid positions for F_1 and G_1 so that patches F and G meet with tangential continuity, if and only if following constraint is satisfied:

For η_0,γ_0, $\eta 2$, and γ_2 calculated using Equation 3.3, either $\gamma_0 \neq \gamma_2$, or $\gamma_0 = \gamma_2$ and $\eta_0 = \eta_2$.

If this constraint is satisfied, Peters gave the solution to set F_1 and G_1 [33] so that we have G^1 continuity along the boundary. But in some cases, the above constraint can not be fulfilled. Further, even when there is a G^1 solution for F_1 and G_1, the shape of the resulting surface might be unacceptable (Figure 3.3).

An approximate solution for F_1 and G_1 is to average the end panel data, including the intersection ratios and cross boundary vectors:

$$\begin{aligned} & \eta_1 = \frac{\eta_0 + \eta_2}{2}, \quad \gamma_1 = \frac{\gamma_0 + \gamma_2}{2}, \quad \vec{u}_1 = H_2 - H_1, \quad \vec{s}_1 = \frac{F_0 - G_0}{2} + \frac{F_2 - G_2}{2}, \\ \Longrightarrow \quad & F_1 = (1 - \gamma_1)\vec{s}_1 + \eta_1 \vec{u}_1, \quad G_1 = -\gamma_1 \vec{s}_1 + \eta_1 \vec{u}_1. \end{aligned}$$

Here the central control panels $\triangle F_1 H_1 H_2$ and $\triangle G_1 H_1 H_2$ are set to be coplanar. Neighbouring patches constructed with this scheme do not in general meet with G^1 continuity. Further, while the coplanar and averaged panels result in normal discontinuity less than 8 degrees, a smaller discontinuity is likely required for most applications. However, this construction provides us an initial arrangement for F_1 and G_1; I will show how to reduce this normal discontinuity in the next section.

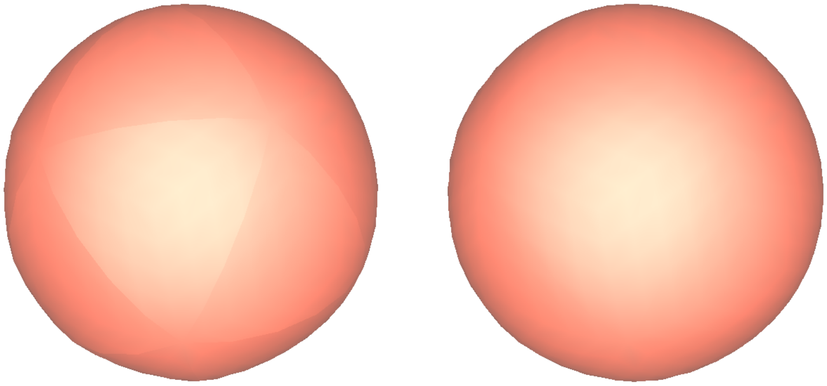


Figure 3.6: Icosahedron surface before and after the adjustments.

3.2.1 Adjustment of control points

To test the results of setting the boundary control points as in Section 3.2, I used the ε-G^1 data fitting scheme described in Section 3.3 to interpolate data sets of an icosahedron and a torus sampled with 10 data triangles per ring.

The left picture in Figure 3.6 shows the resulting surface of an icosahedron. The maximum discontinuity angle of this surface is 7.20 degree, and the creases along macro-patch boundaries are visible. To reduce the discontinuity, I adjusted F_1 and G_1 so as to have equal normals at additional interior points along the boundary (we already have equal normals at two end points). The reason why such an adjustment reduces the discontinuity will be explained in Chapter 5. To make this adjustment for a given point P on the boundary curve, I performed de Casteljau's algorithm on patches F and G individually until the last step, leaving the two triangles $\triangle F$ and $\triangle G$ spanned by vectors $\vec{u}$, $\vec{v}$, $\vec{w}$, as shown in Figure 3.7(b). For the vectors shown in Figure 3.7(a), define

$$\vec{u}_i = H_{i+1} - H_i, \quad \vec{v}_i = F_i - H_i, \quad \vec{w}_i = G_i - H_i,$$

then $\vec{u}$, $\vec{v}$, $\vec{w}$ are all defined by quadratic Bézier functions:

$$\vec{u} = \sum_{i=0}^{2} \vec{u}_i B_i^n(t), \quad \vec{v} = \sum_{i=0}^{2} \vec{v}_i B_i^n(t), \quad \vec{w} = \sum_{i=0}^{2} \vec{w}_i B_i^n(t).$$

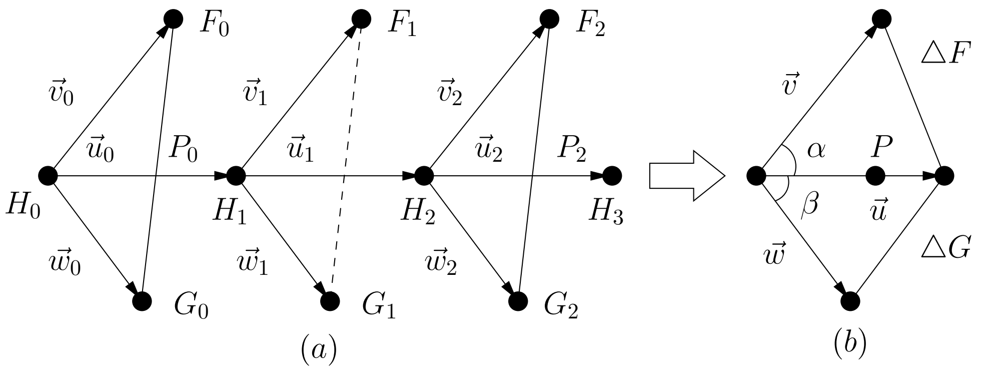


Figure 3.7: Control vectors along a boundary curve.

The normal vectors of $\triangle F$ and $\triangle G$ are defined by the cross products of the difference vectors:

$$\vec{N_F} = \frac{\vec{u} \times \vec{v}}{|\vec{u} \times \vec{v}|}, \quad \vec{N_G} = \frac{\vec{w} \times \vec{u}}{|\vec{w} \times \vec{u}|}.$$

Define the angle spanned by $\vec{u}$ and $\vec{v}$ as α, and the angle between $\vec{w}$ and $\vec{u}$ as β (Figure 3.7 (b)). Suppose we already have an ε-G^1 solution using the method from Section 3.1; now adjust F_1 and G_1, *i.e.*, vectors $\vec{v}_1$ and $\vec{w}_1$, so that patches F and G meet with equal normals at additional interior points on the boundary. If we add offset vectors, say $\vec{f}$ and $\vec{g}$, to F_1 and G_1 individually, the difference vectors in the center panels can be rewritten as

$$\vec{u_1'} = \vec{u}_1, \quad \vec{v_1'} = \vec{v}_1 + \vec{f}, \quad \vec{w_1'} = \vec{w}_1 + \vec{g}.$$

The vectors in $\triangle F$ and $\triangle G$ then change to

$$\vec{u'} = \vec{u}, \quad \vec{v'} = \vec{v} + 2t(1-t)\vec{f}, \quad \vec{w'} = \vec{w} + 2t(1-t)\vec{g}.$$

After adjustments, we want $\vec{u'}$, $\vec{v'}$, $\vec{w'}$ to be coplanar at P, therefore we should have

$$\vec{v'} \cdot (\vec{w'} \times \vec{u'}) = (\vec{v} + 2t(1-t)\vec{f}) \cdot ((\vec{w} + 2t(1-t)\vec{g}) \times \vec{u}) = 0. \tag{3.4}$$

The goal is to find $\vec{f}$ and $\vec{g}$ that satisfy Equation 3.4.

3.2.2 One equal-normal point

In Equation 3.4, we first suppose G_1 is fixed, then we can solve for F_1. Letting $\vec{g} = 0$, and $t \in (0,1)$, Equation 3.4 becomes

$$\begin{aligned} \vec{v}' \cdot (\vec{w}' \times \vec{u}') &= (\vec{v} + 2t(1-t)\vec{f}) \cdot (\vec{w} \times \vec{u}) = 0 \\ \Longrightarrow \vec{f} \cdot \vec{N_G} &= -\frac{\vec{v} \cdot \vec{N_G}}{2t(1-t)}. \end{aligned} \tag{3.5}$$

Vectors $\vec{v}$ and $\vec{N_G}$ are fixed once the parameter t is set. The geometric meaning of Equation 3.5 is that all the solutions of $\vec{f}$ define a plane perpendicular to $\vec{N_G}$. To have equal normals at a third point, we solve Equation 3.5 with a given t value. Since we cannot solve the three unknown components of $\vec{f}$ with only one equation, I set $\vec{f}$ parallel to $\vec{N_G}$ to minimize the length of $\vec{f}$ (in Section 3.2.3, an alternative approach is to have more than one equal-normal point on the boundary). Then Equation 3.5 becomes

$$\vec{f} = -\frac{\vec{v} \cdot \vec{N_G}}{2t(1-t)} \vec{N_G}. \tag{3.6}$$

Equation 3.6 shows how to adjust F_1 when G_1 is fixed.

Now we consider the general case of how to solve $\vec{f}$ and $\vec{g}$ simultaneously to have one additional equal-normal point along the boundary. In a practical surface fitting scheme, the adjustments to F_1 and G_1 should obey the following rules:

- F_1 and G_1 should be adjusted only once.
- The changes to F_1 and G_1 should be symmetric.
- Vectors $\vec{f}$ and $\vec{g}$ should be independent of which of F and G we construct first.

To solve for $\vec{f}$ and $\vec{g}$ simultaneously from Equation 3.4, we first find the directions of $\vec{f}$ and $\vec{g}$, then calculate their lengths. As shown in Figure 3.8(a), $\triangle F$ and $\triangle G$ are the results of evaluating patches F and G using de Casteljau's algorithm at the boundary point P until the last step. Before adjustments, the two normal vectors $\vec{N_F}$ and $\vec{N_G}$ are not equal, and $\triangle F$ and $\triangle G$ are not coplanar, as shown in Figure 3.8(a). After adjustments, $\triangle F$ and $\triangle G$ should be on the same plane defined by a normal vector $\vec{N}$, as shown in Figure 3.8(a).

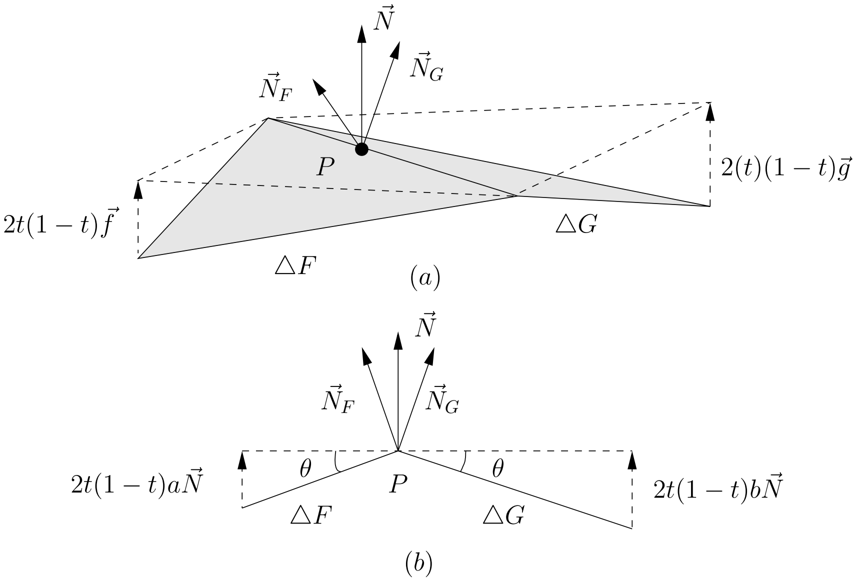


Figure 3.8: Adjustment of a pair of control points.

A 2D side-view is presented in Figure 3.8(b), where $\triangle F$ and $\triangle G$ are projected into the plane perpendicular with $\vec{u}$. To distribute the adjustments evenly, set the target normal $\vec{N}$ to bisect $\vec{N_F}$ and $\vec{N_G}$ (other choices of $\vec{N}$ might generate a surface with better visual quality, and should be investigated in the future):

$$\vec{N} = \frac{\vec{N_F} + \vec{N_G}}{|\vec{N_F} + \vec{N_G}|}. \tag{3.7}$$

After the adjustments, both $\triangle F$ and $\triangle G$ are on the target plane. If we set $\vec{f}$ and $\vec{g}$ to be parallel to $\vec{N}$ to minimize the vector length, then $\vec{f}$ and $\vec{g}$ become

$$\vec{f} = a\vec{N}\ , \quad \vec{g} = b\vec{N}.$$

As shown in Figure 3.8, the ratio between vector length a and b is

$$\frac{a}{b} = \frac{Area(\triangle F)}{Area(\triangle G)} = \frac{|\vec{u} \times \vec{v}|}{|\vec{w} \times \vec{u}|}. \tag{3.8}$$

Substituting Equation 3.8 in Equation 3.4 leads to

$$\begin{aligned} \vec{v'} \cdot (\vec{w'} \times \vec{u'}) &= (\vec{v} + 2t(1-t)a\vec{N}) \cdot ((\vec{w} + 2t(1-t)b\vec{N}) \times \vec{u}) \\ &= \vec{v} \cdot \vec{N_G} + 4t(1-t)a\vec{N} \cdot \vec{N_G} \\ &= 0 \\ \Longrightarrow a &= -\frac{|\vec{v}| \sin(\alpha) \sin(\theta)}{2t(1-t)}, \quad b = -\frac{|\vec{w}| \sin(\beta) \sin(\theta)}{2t(1-t)}. \end{aligned} \tag{3.9}$$

Here, α is the angle between vector $\vec{v}$ and $\vec{u}$; β is the angle between $\vec{u}$ and $\vec{w}$ (Figure 3.7(b)); θ is the angle between $\vec{N}$ and $\vec{N_F}$ ($\vec{N}$ and $\vec{N_G}$), as shown Figure 3.8(b). Equation 3.9 shows that for two patches meeting with ε-G^1 continuity, equal-normal points can be achieved at the interior of the boundary curve by adjusting the center control points. Note that if the two patches already meet with equal normals at P with initial positions of F_1 and G_1, then $\sin(\theta) = 0$ and $a = b = 0$, and the adjustment leaves F_1 and G_1 unchanged.

3.2.3 Multiple equal-normal points

To have more interior equal-normal points on the boundary, I first construct a target plane for each chosen t value as discussed in Section 3.2.2. If it is possible to set $\vec{f}$ and $\vec{g}$ in a way that F_1 and G_1 will be on the intersection of the target planes, then there can be multiple equal-normal points along the boundary. Using this technique, there can be up to three interior equal-normal points.

For the case of two equal-normal points, the two target normals are set by Equation 3.7 individually. If the two target planes are not parallel to each other, as long as $\vec{f}$ and $\vec{g}$ move F_1 and G_1 to the intersection line of the two planes, there will be two equal-normal points along the boundary. By defining $\vec{f}$ and $\vec{g}$ perpendicular to this intersection line, we have enough constraints to solve for $\vec{f}$ and $\vec{g}$. To have equal normals at three different t values, I compute the three target normals by Equation 3.7 individually. If these three target planes (defined by the three target normals) intersect at one point, then both $\vec{f}$ and $\vec{g}$ are solved uniquely. Since the two or three target planes are not guaranteed to intersect, the solution for $\vec{f}$ and $\vec{g}$ is not guaranteed for the two or three equal-normal points methods.

Theoretically, having more equal-normal points along boundaries can reduce the discontinuity more than the one-point method does (detailed explanation in Chapter 5). However, I found in practice that methods using two or three equal-normal points (if solutions exist) do not provide substantial improvements over the one-point method and might result in poor surface shape. The poor shape occurs when the two or three equal-normal points solutions move the control points far away. In my thesis, I only use the method with one equal-normal point along each boundary curve.

3.3 Surface fitting scheme

To test the approximate G^1 boundary construction, a simple cubic parametric surface fitting scheme was devised. This scheme constructs boundary curves first, then sets the interior control points of the patch. For each boundary, we need the end panel data to construct the center points F_1 and G_1; further, to perform a visual inspection of the normal discontinuity, we need to construct complete surface patches for rendering. My cubic ε-G^1 scheme is similar to Clough-Tocher's scheme for data above a plane [3]. The fitting scheme proceeds in the following steps:

1. Construct a cubic patch for each data triangle, as shown in Figure 3.9 (left). This initial patch interpolates the positions and normal vectors of the three data vertices. The center control point of the patch is set to be a weighted average of the other nine control points, using Farin's quadratic precision method [9]. I then subdivide the patch into three cubic patches (called *micro-patches*), as shown in Figure 3.9.

2. Figure 3.10 shows the three micro-patches of current data triangle and the patches of three neighbouring triangles (in dashed lines) subdivided the same way. The center control points of the micro-patches are shown as large black dots in Figure 3.10, including those of micro-patches from adjacent triangles. Set the center control points so that two micro-patches have ε-G^1 continuity along the *macro-patch* (the patch corresponding to the whole data triangle) boundary. I used the method introduced in Section 3.2.2 to have equal normals at one interior point along each macro-patch boundary.

3. Each of the small and large white points are constructed as the gravity centre of the adjacent three control points, as in Clough-Tocher's scheme [10][28].

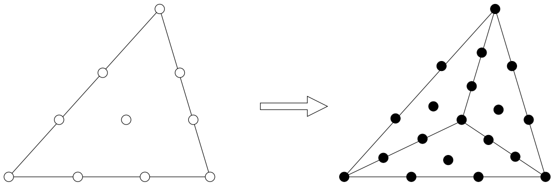


Figure 3.9: Macro patch split.

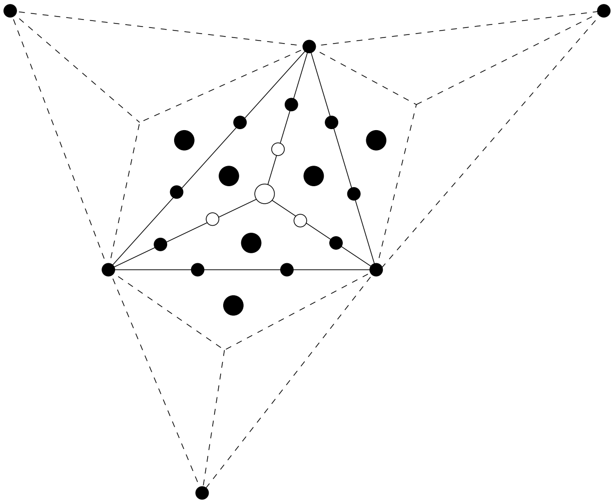
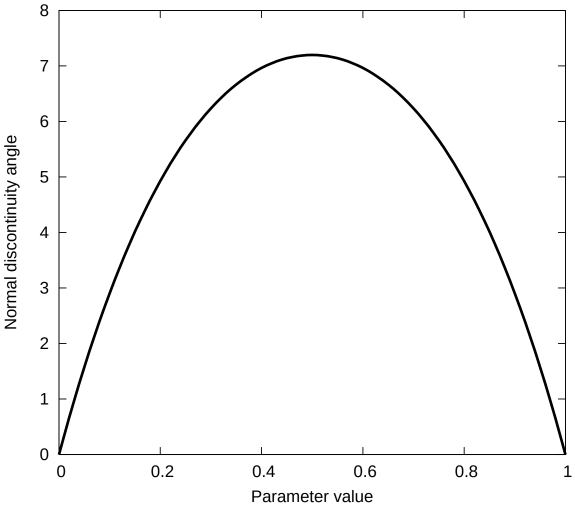


Figure 3.10: Construction of micro-patches.

In this scheme, the second step uses the method introduced in Section 3.1 to set initial positions for the big black points. Points F_1 and G_1 are then adjusted by adding an offset vector calculated with method of Section 3.2.1. This scheme has ε-G^1 continuity along the macro-patch boundary. C^1 continuity is guaranteed along the internal boundary between two micro-patches (with the three micro-patches meeting with C^2 continuity at the split point, a result of the averaging in Step 3 [10]).

3.4 Results

The data fitting scheme described in the previous section was applied to various examples to test how large the discontinuity was along the boundaries using the various variations of the ε-G^1 constructions. Further, since the ϵ-G^1 scheme is a complete data fitting scheme, I compared it with some G^1 schemes of similar complexity. Subsection 3.4.1 gives tests on the magnitude of the discontinuity along the boundaries, and Subsection 3.4.2 compares the surfaces of my ϵ-G^1 scheme to those of Shirman-Séquin and Peters.



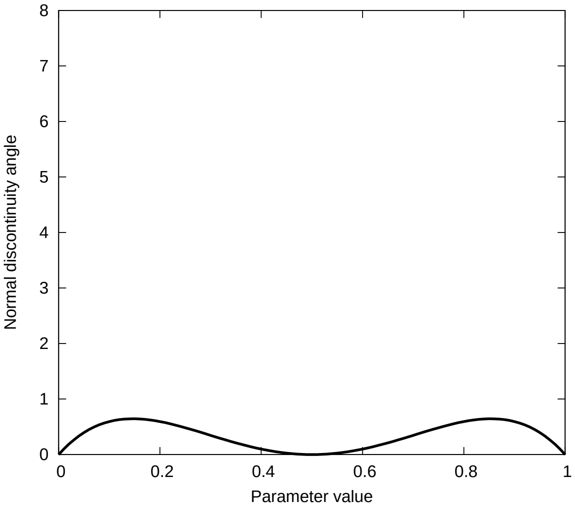


Figure 3.11: Discontinuity before reduction. Figure 3.12: Discontinuity after reduction.

3.4.1 Discontinuity reduction

I reduce the discontinuity on a surface with ϵ-G^1 continuity by increasing the number of equal-normal points along boundary curves.

To check the results of the discontinuity reduction for an individual boundary curve, I sampled along a boundary curve for the maximum discontinuity on an icosahedron surface. At each of the 100 sampled points, the discontinuity angle between two surface normals was calculated and the results in degree are shown in Figure 3.11 and Figure 3.12. Before the reduction operations, the discontinuity angle reaches the highest value at $t = 0.5$, as shown in Figure 3.11. After the adjustments to have equal normals at the middle point, the discontinuity at $t = 0.5$ is zero, and the maximum discontinuity drops dramatically. The right picture in Figure 3.6 shows the resulting surface of icosahedron after having equal normals at a middle boundary point; the creases observed in the left picture are gone.

Figure 3.13 shows the results of the discontinuity reduction for all the boundary curves of a torus surface sampled at 10 data triangles per ring. As shown in Figure 3.13, the maximum discontinuity angle with equal normals only at the boundary end points is compared to the maximum discontinuity angle with equal normals at boundary end points and at $t = 0.5$. Due to the symmetrical pattern used to sample the torus model, multiple bound-

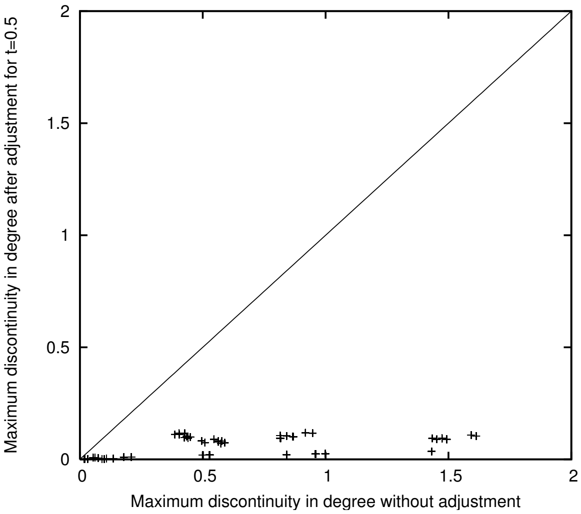


Figure 3.13: Discontinuity reduction.

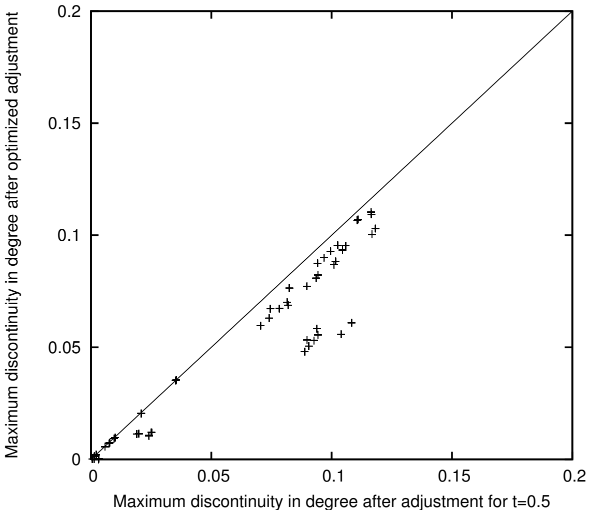


Figure 3.14: Discontinuity comparison.

aries share the same patch configuration. Therefore each point in Figure 3.13 represents more than one boundary curve. The plot shows that for every boundary, the discontinuity angle has been dramatically reduced.

One question that remains to be addressed is which t values should we use for these points? For the method that has one interior equal-normal point, we could perform a numerical search to determine the t value that gives the smallest maximum discontinuity. Figure 3.14 shows a comparison of results of having equal normals at $t = 0.5$ with having equal normals at the optimal t value. For this comparison, I chose a sequence of t values from 0 to 1, with a step of 0.01. For each value of t, I adjusted F_1 and G_1 to have equal normals at the corresponding point and then calculate the maximum discontinuity angle along the boundary. Finally I compared the minimum of these biggest discontinuity angles with the maximum angle for having equal normals at $t = 0.5$, as shown in Figure 3.14. Although the optimal t value varies from 0.12 to 0.88, the result using $t = 0.5$ is nearly as good as the optimal t value, with a difference of less than 0.05 degrees. When we set $t = 0.5$, Equation 3.9 simplifies to

$$a = -2|\vec{v}|\sin(\alpha)\sin(\theta), \qquad b = -2|\vec{w}|\sin(\beta)\sin(\theta). \tag{3.10}$$

To test the results of multiple equal-normal points, I set $t = \frac{1}{3}, \frac{2}{3}$ for the two-point

Table 3.1: Discontinuity angles before and after adjustments.

Model	Before	Interior points with equal normals		
		One	Two	Three
Icosahedron	7.20	0.65	0.52	0.55
Torus	1.61	0.12	0.04	0.11

Table 3.2: Discontinuity angles of torus surfaces.

Sampling rate	Max discontinuity
10	0.117087
20	0.008262
40	0.000485

method, and $0.25, 0.5, 0.75$ for the three-point method. Table 3.1 shows the maximum discontinuity angles of an icosahedron and a torus (sampled by 10 triangles per ring), before and after the adjustments, where all the angles are in degree. As shown in Table 3.1, the one-point method reduces the discontinuity significantly; the two-points method reduces the discontinuity further, but not much; while the three-point method is almost the same as that of the one-point method. This experiment suggests that the three methods basically generate similar results for the discontinuity reduction. Since the two- and three-point methods are not guaranteed to work, I used the one-point method in the remaining experiments.

The other factor influencing the discontinuity on the resulting surfaces is the sampling rate of the data mesh. Table 3.2 shows the maximum discontinuity angle of torus surfaces with different sampling resolutions. As we increase the sampling rate, the maximum normal discontinuity angle drops quickly. All the data in Table 3.2 are generated using the one equal-normal point method discussed above. While we can achieve a reduction in the discontinuity when we increase the sampling rate, we cannot do this for some data meshes, such as those scanned from real objects. Increasing the sampling rates is more useful for testing than for real application.

Table 3.3: Gaussian curvature of torus surfaces.

Sampling rate	Our scheme	Shirman-Séquin's
10	(2.09477, -4.70887)	(2.27557, -5.24718)
20	(1.52910, -4.26633)	(1.56662, -4.40811)
40	(1.38138, -4.07419)	(1.39029, -4.10928)

3.4.2 Comparison to G^1 schemes

To check the surface quality of my ϵ-G^1 surfaces, I applied my surface fitting scheme and two G^1 schemes to the data mesh of a torus and compared the results. There are other G^1 schemes that use optimizations that generate better surfaces than the two G^1 schemes to which I compare my scheme. However, I feel it is more appropriate to compare my ϵ-G^1 scheme to those schemes with similar computational costs. In the Chapter 4, I present a more sophisticated ϵ-G^1 scheme and compare it to some better G^1 schemes.

In this section, I compare the results of my scheme to that of Shirman-Séquin's scheme using quartic boundary curves and to Peter's scheme with cubic patches. For these comparisons, I only used the variation of one equal-normal point along the macro-boundaries. The main concern was that the ϵ-G^1 scheme should not perform worse than these two schemes, and we had some hopes that it might perform better.

In Figure 3.15, the first row shows the surface with 10 data triangles sampled for each ring; the second row shows the sampling rate at 20; the surface on the left of each row was generated by my scheme, and the surfaces on the right were generated with Shirman-Séquin's scheme with cubic boundary curves (Peters' scheme can not be used on this data set, because the data set does not admit a G^1 solution for cubics). My scheme has the same visual continuity as Shirman-Séquin's.

From the data set used in Figure 3.15, I perturbed the locations of vertices on the torus to avoid the singular case that Peters' cubic scheme fails on. This allowed us to use Peters' scheme [33] to fit cubic surfaces to the data, although since it is near a singular case we expected Peters scheme to perform poorly. As shown in Figure 3.16, the surface on the left is the result of the ϵ-G^1 scheme, the one in the middle is the cubic solution from Peters', the one on the right is from Shirman-Séquin's. Although Peters' cubic solution

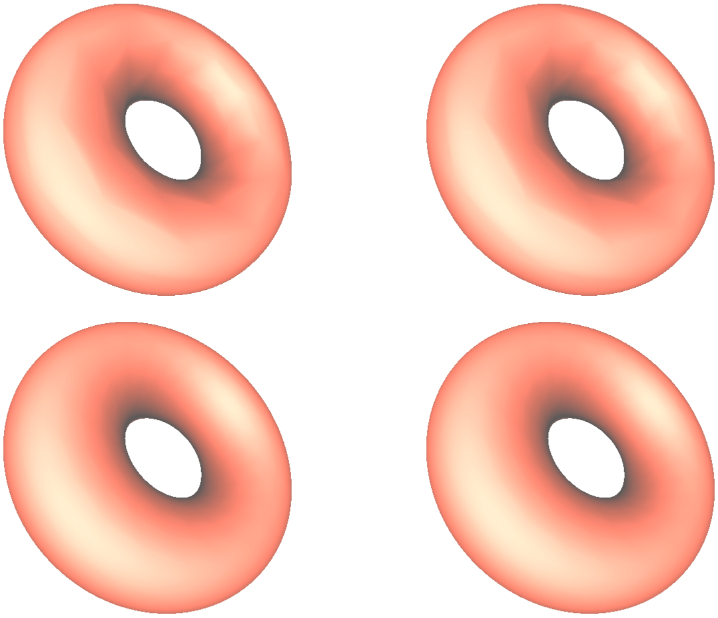


Figure 3.15: Torus surfaces compared to Shirman-Séquin's.

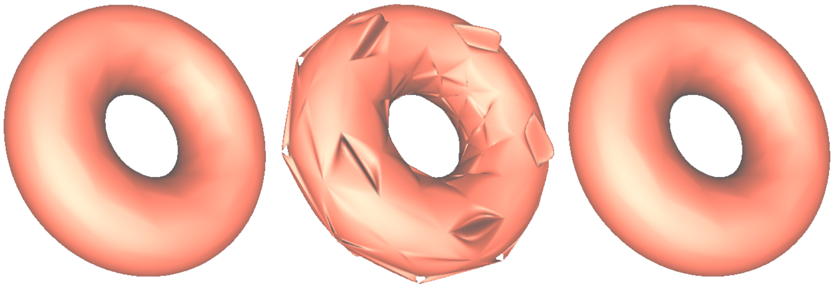


Figure 3.16: Surfaces compared to Peter's and Shirman-Séquin's.

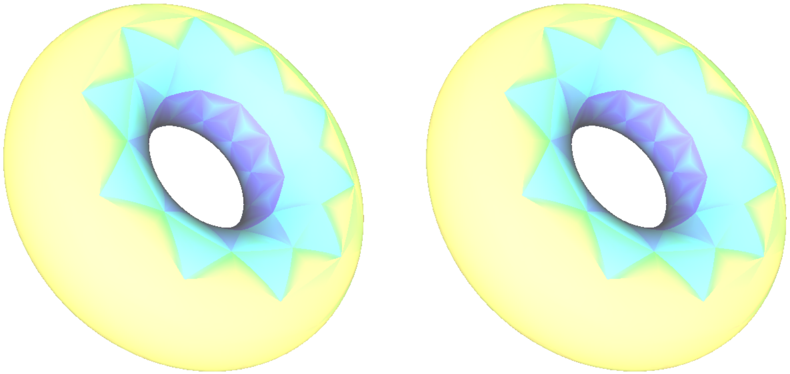


Figure 3.17: Torus curvature plot compared to Shirman-Séquin's.

can guarantee G^1 continuity along the boundary, the surface has poor shape as expected.

If we look at Gaussian curvature plots of the surfaces fit to the torus with a sampling rate of 10, as shown in Figure 3.17, my approximate G^1 solution gives almost the same curvature plot as Shirman-Séquin's scheme. If we look at the maximum/minimum Gaussian curvature values at different sampling rates (Table 3.3), we find that surfaces from our scheme have a slightly better curvature range than Shirman-Séquin's. Here the actual range of Gaussian curvature values for the torus sampled to get the data sets is $[\frac{4}{3}, -4]$.

Chapter 4

Parametric surface fitting without domain splitting

Other than the domain splitting methods, some polynomial G^1 surface schemes interpolate the given data set using one Bézier patch per data triangle. For these one-patch schemes, the *twist compatibility* problem or *vertex consistency* problem, has to be resolved [25, 37, 41]. Existing G^1 schemes solve this problem by using high degree patches for each data triangle. For example, Loop designed a scheme using sextic triangular Bézier patches in the one-to-one correspondence with data triangles [25]. In Loop's scheme, the boundary control points are constructed to guarantee solutions to the twist compatibility problem.

In this chapter, I will introduce an approximate G^1 scheme that fits one Bézier patch for each data triangle. In this ε-G^1 scheme, positions of the data vertices are interpolated by using one quintic triangular patch for each data triangle, generating surfaces with approximate G^1 continuity along the patch boundaries. To reduce the normal discontinuity, the method described in Chapter 3 is extended to adjust two neighbouring quintic patches to have equal normals at the middle point of their common boundary curve. I will begin by reviewing the twist compatibility problem and Loop's solution to it, since while I do not have to solve the twist compatibility problem, I use Loop's construction as a basis for my approximate G^1 construction.

4.1 Twist compatibility

When constructing a G^1 continuous surface with one patch for each data triangle of the data mesh, we have to solve the twist compatibility problem. Although solving this problem is not required for creating surfaces with approximate G^1 continuity, the solution to this problem has great impact on the shape quality of the resulting surface. In Loop's scheme, the construction of boundary curves can guarantee solutions to the twist compatibility problem [25], and help to achieve good surface quality on the resulting surface. My ϵ-G^1 scheme constructs the tangents and twist terms at the data vertices the same as in Loop's scheme [25]. In this chapter, I review the twist compatibility problem and outline Loop's construction.

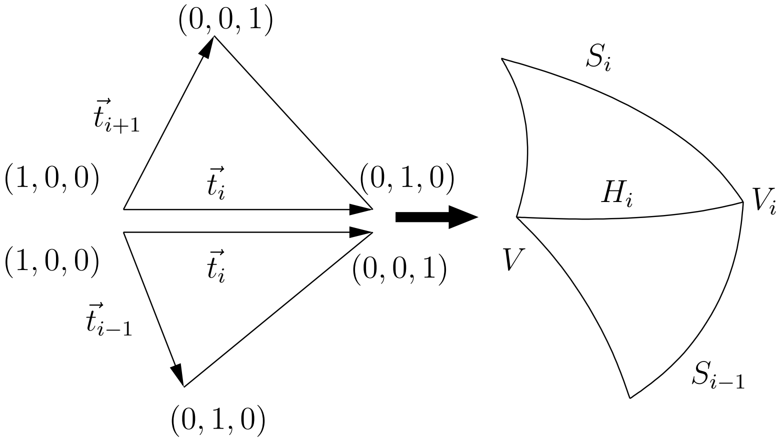


Figure 4.1: Two adjacent parametric patches.

For a given triangular closed mesh $\mathcal{M}$, a piecewise triangular surface is constructed to interpolate the positions of data vertices in $\mathcal{M}$. For each data vertex V in $\mathcal{M}$, the number of incident edges is referred to as the valence of V. Since we assume $\mathcal{M}$ is of arbitrary topology and triangulated without singularity, the valence of V is always greater than two.

For vertex V with valence n, two adjacent patches S_i and S_{i-1} and their domain triangles are shown in Figure 4.1. The direction for the partial derivative along the ith edge of V is defined as $\vec{t}_i$ in the domain triangles. The boundary curve between patch S_i and S_{i-1}

is $H_i(t)$, with $H_i(0) = V$ and $H_i(1) = V_i$. Assuming that the given data mesh $\mathcal{M}$ is not degenerate, we have

$$\frac{\partial S_i}{\partial \vec{t}_{i+1}} \times \frac{\partial S_i}{\partial \vec{t}_i} \neq 0.$$

Here the partial derivative $\partial S_i/\partial \vec{t}_i$ is a vector valued spline of one degree less than S_i. Then S_i and S_{i-1} meet with G^1 continuity along H_i if and only if

1. Patch S_i and S_{i-1} share the common boundary curve: $S_i(t, 1-t, 0) = S_{i-1}(t, 0, 1-t)$.
2. There are scalar valued functions $\mu(t)$ and $\nu(t)$ such that

$$\frac{\partial S_i}{\partial \vec{t}_i} = \mu(t)\frac{\partial S_i}{\partial \vec{t}_{i+1}} + \nu(t)\frac{\partial S_{i-1}}{\partial \vec{t}_{i-1}}. \tag{4.1}$$

To guarantee the proper orientation of the tangent planes, assume $\mu(t)\nu(t) \geq 0$. Differentiating Equation 4.1 along the direction of $\vec{t}_i$ and evaluating at $t = 0$ leads to the following equation:

$$\frac{\partial^2 S_i}{\partial \vec{t}_i^2} = \mu_i'(0)\frac{\partial S_i}{\partial \vec{t}_{i+1}} + \mu_i(0)\frac{\partial^2 S_i}{\partial \vec{t}_{i+1}\partial \vec{t}_i} + \nu_i'(0)\frac{\partial S_{i-1}}{\partial \vec{t}_{i-1}} + \nu_i(0)\frac{\partial^2 S_{i-1}}{\partial \vec{t}_{i-1}\partial \vec{t}_i}. \tag{4.2}$$

Here $\partial^2 S_i/\partial \vec{t}_{i+1}\partial \vec{t}_i$ and $\partial^2 S_{i-1}/\partial \vec{t}_{i-1}\partial \vec{t}_i$ are the mixed second order partial derivatives, *i.e.*, the twist terms. All other terms in Equation 4.2 are known once the boundary control points have been constructed. When constructing n patches to meet with G^1 continuity at V, there are n instances of Equations 4.2 to resolve. Solving this linear system is called the twist compatibility problem [4, 37, 41].

For surface fitting schemes without domain splitting, solving the twist compatibility problem is an essential issue. However, Watkins showed that a solution for an arbitrary boundary construction is not guaranteed to exist if the valence of V is a even number [41]. If a solution exists, while we can set the twist control points to generate a G^1 continuous surface, it is unclear whether the shape of such a surface is satisfactory.

To better illustrate the twist compatibility problem and also relate it to Loop's solution, I present Equation 4.2 in terms of control points. As we discussed in Chapter 3, it is not always possible to construct a G^1 continuous surface using cubic patches for an arbitrary data set [34]. Therefore we construct one quartic patch S_i for the data triangle $\triangle VV_iV_{i+1}$,

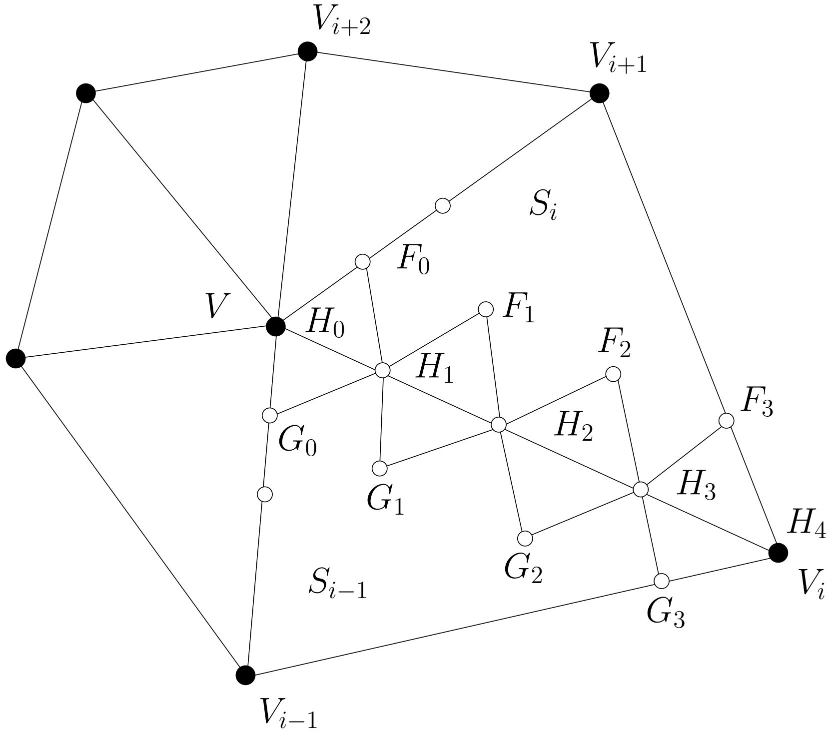


Figure 4.2: Patches surrounding a vertex.

as shown in Figure 4.2. Two adjacent patches S_i and S_{i-1} meet at the quartic boundary curve H, with control points H_i, $j = 0, 1, 2, 3, 4$. By default, we require that the vertex positions are interpolated, therefore $H_0 = V$ and $H_4 = V_i$. To interpolate the vertex normals as well, points H_1 and H_3 are placed on the tangent planes at V and V_i respectively. Finally, the middle control point H_2 is calculated using information from adjacent data vertices. Similarly, we set the other boundary curves originating from V.

The twist compatibility problem occurs when constructing the interior points after all boundary curves are set. As shown in Figure 4.2, if S_i and S_{i-1} meet with G^1 continuity, the twist control points F_1 and G_1 have to be set with certain constraints. With the definition of vectors as

$$\vec{u}_i = H_{i+1} - H_i, \quad \vec{v}_i = F_i - H_i, \quad \vec{w}_i = G_i - H_i,$$

Equation 4.1 turns into

$$\sum_{j=0}^{3} B_j^3(t)\vec{u}_i = \mu_i(t)\sum_{j=0}^{3} B_j^3(t)\vec{v}_i + \nu_i(t)\sum_{j=0}^{3} B_j^3(t)\vec{w}_i. \tag{4.3}$$

Here the $B_j^3(t)$ are the cubic Bernstein polynomials. Once all the tangent points are set, we evaluate Equation 4.3 at $t = 0$ and take the cross product with the vector $\vec{u}_0$ on both sides:

$$\begin{aligned} & \vec{u}_0 = \mu_i(0)\vec{v}_0 + \nu_i(0)\vec{w}_0 \\ \Longrightarrow\ & \vec{u}_0 \times \vec{u}_0 = \mu_i(0)\vec{v}_0 \times \vec{u}_0 + \nu_i(0)\vec{w}_0 \times \vec{u}_0 \\ \Longrightarrow\ & \frac{\mu_i(0)}{\nu_i(0)} = \frac{|\vec{w}_0 \times \vec{u}_0|}{|\vec{u}_0 \times \vec{v}_0|} = \frac{\text{area}\triangle H_0G_0H_1}{\text{area}\triangle H_0H_1F_0}. \end{aligned} \tag{4.4}$$

We also assume $\mu_i(0), \nu_i(0) > 0$, and $\mu_i(0) + \nu_i(0) = 1$, which means that the coefficients $\mu_i(0)$ and $\nu_i(0)$ can be determined from Equation 4.4, Similarly, the values of $\mu_i(1)$ and $\nu_i(1)$ are set using the information at vertex V_i:

$$\mu_i(0) = \mu_0, \quad \nu(0) = 1 - \mu_0, \quad \mu_i(1) = \mu_1, \quad \nu(1) = 1 - \mu_1.$$

To make the calculation simpler, we set $\mu_i(t)$ and $\nu_i(t)$ to be linear functions, therefore $\mu_i(t)$ and $\nu_i(t)$ are completely defined.

We can now expand both sides of Equation 4.3, and compare the coefficients of each Bernstein polynomial on both sides. Because the left side of Equation 4.3 is one degree lower than the right side, I first elevate the degree of the left side to quartic. Then Equation 4.3 is true for any t value if and only if

$$\begin{aligned} \vec{u}_0 &= \mu_0\vec{v}_0 + (1-\mu_0)\vec{w}_0, \\ 3\vec{u}_1 + \vec{u}_0 &= 3\mu_0\vec{v}_1 + \mu_1\vec{v}_0 + 3(1-\mu_0)\vec{w}_1 + (1-\mu_1)\vec{w}_0, \\ \vec{u}_2 + \vec{u}_1 &= \mu_0\vec{v}_2 + \mu_1\vec{v}_1 + (1-\mu_0)\vec{w}_2 + (1-\mu_1)\vec{w}_1, \\ \vec{u}_3 + 3\vec{u}_2 &= \mu_0\vec{v}_3 + 3\mu_1\vec{v}_2 + (1-\mu_0)\vec{w}_3 + 3(1-\mu_1)\vec{w}_2, \\ \vec{u}_3 &= \mu_1\vec{v}_3 + (1-\mu_1)\vec{w}_3. \end{aligned} \tag{4.5}$$

The first and fifth equations in Equation 4.5 are fulfilled after we set the boundary control points. Due to the symmetry of the patch layout, the fourth equation is equivalent to the second one. To solve the second and third equations, we replace the vectors with control points, giving

$$\begin{aligned} \mu_0 F_1 + (1-\mu_0)G_1 + \frac{1}{3}\mu_1 F_0 + \frac{1}{3}(1-\mu_1)G_0 &= H_2 + \frac{1}{3}H_1, && (4.6) \\ \mu_0 F_2 + (1-\mu_0)G_2 + \mu_1 F_1 + (1-\mu_1)G_1 &= H_3 + H_2. && (4.7) \end{aligned}$$

Equations 4.6 and 4.7 are equivalent to the G^1 constraints described by Farin [11](page 368). In Farin's book, the G^1 conditions are targeted for arbitrary degree patches and boundary curves. To solve the unknown points of F_1 and G_1, re-arrange Equation 4.6:

$$\begin{aligned} \mu_0 F_1 + (1-\mu_0)G_1 &= H_2 + \frac{1}{3}H_1 - \frac{1}{3}\mu_1 F_0 - \frac{1}{3}(1-\mu_1)G_0 \\ &= E. && (4.8) \end{aligned}$$

It is easy to confirm that the coefficients of points in Equation 4.8 sum to one, therefore E is an affine combination of the existing boundary control points. Constructing Equation 4.8 for each edge of the vertex V leads to a linear system. To better present this linear system, re-define P_i as the twist control point of the data triangle $\triangle V V_i V_{i+1}$ (F_1 in Equation 4.8). Counting the neighbouring vertices of V in anti-clockwise order, the point G_1 is therefore re-named as P_{i-1}. For a vertex V with valence of n, there are n instances of Equation 4.8, with n unknown points P_i:

$$\begin{aligned} \varphi_0 P_1 + (1-\varphi_0)P_0 &= E_0, \\ \varphi_1 P_2 + (1-\varphi_1)P_1 &= E_1, \\ \varphi_2 P_3 + (1-\varphi_2)P_2 &= E_2, \\ &\ldots, \\ \varphi_{n-1} P_0 + (1-\varphi_{n-1})P_{n-1} &= E_{n-1}. \end{aligned}$$

Here φ_i is the scalar value of $\mu_i(0)$ for the ith boundary of the vertex V. The matrix form of this linear system is

$$\begin{pmatrix} (1-\varphi_0) & \varphi_0 & 0 & \ldots & 0 \\ 0 & (1-\varphi_1) & \varphi_1 & 0 & \ldots \\ & & \vdots & & \\ \varphi_{n-1} & 0 & \ldots & 0 & (1-\varphi_{n-1}) \end{pmatrix} \begin{pmatrix} P_0 \\ P_1 \\ \vdots \\ P_{n-1} \end{pmatrix} = \begin{pmatrix} E_0 \\ E_1 \\ \vdots \\ E_{n-1} \end{pmatrix}. \qquad (4.9)$$

Solving the linear system of Equation 4.9 is equivalent to solving the twist compatibility problem presented in [37, 40, 41]. A simple calculation shows that the determinant of the coefficients matrix in Equation 4.9 is

$$D = \prod_{i=0}^{n-1}(1-\varphi_i) - (-1)^n \prod_{i=0}^{n-1}\varphi_i.$$

When n is even, D can be zero and in that case there will be no non-trivial solution; if n is odd, it is possible to set the twist points uniquely by solving Equation 4.9, but the solution is not guaranteed to satisfy the G^1 constraints at the other end of the boundary curve (Equation 4.7).

Instead of solving Equation 4.9 for an arbitrary configuration of boundary curves, Loop presented a solution by setting $\varphi_i = 0.5$, and constructing all the tangential control points surrounding a given vertex to form a regular "n-gon" [25].

In Loop's scheme, all the points in the definition of E (Equation 4.8) are set as the average of two points, each calculated uniquely from the corresponding data triangle. In this way, a solution to Equation 4.9 is guaranteed and a G^1 interpolant is always available for the given data mesh. After comparing results of different configurations, I found that Loop's quartic boundary curve construction can provide good surface quality. For an approximate G^1 scheme, we do not have to meet the constraints imposed by the G^1 conditions. In other words, an ε-G^1 surface is not required to have a solution for Equation 4.9. However, my new ε-G^1 scheme still uses the tangent and twist point construction from Loop's scheme because experiments showed that such a configuration can help to improve the surface smoothness and yield better overall shape quality. Since the new ε-G^1 scheme generates quintic surfaces, using the quartic boundaries of Loop's scheme poses no problem.

4.2 Loop's scheme

Loop's sextic scheme has the following steps:

1. Construct quartic boundary curves.
2. Solve the twist terms.

3. Construct tangent fields along the boundaries.
4. Elevate the boundary curves to degree six.
5. Set the second row of control points in the sextic patch to interpolate the tangent fields.
6. Set the remaining control point by averaging the control points from the steps above.

When the three vertices of a data triangle have the same valence, the patch constructed by Loop's scheme is quintic; if the three valences are all six, then the patch degree reduces to quartic. In the remaining part of this section, I will review the first two steps briefly; the details of the other steps of Loop's scheme are in [25]. For each quartic boundary curve with control points of H_i^0, ..., H_i^4, Loop constructs the first two control points as

$$H_i^0 = \alpha V + (1-\alpha)A = \alpha V + \frac{1-\alpha}{n}\sum_{j=1}^{n} V_j, \tag{4.10}$$

$$H_i^1 = H_i^0 + \frac{\beta}{n}\sum_{j=1}^{n}\left[\cos\left(\frac{2(j-i)\pi}{n}\right)\right] V_j. \tag{4.11}$$

Here n is the valence of vertex V; A is the averaged centroid if all V's neighboring vertices. The construction of H_i^0 and H_i^1 along all boundaries performs a first order Fourier transformation on neighbouring vertices of V [4]. If all the control points H_i^0 and H_i^1 are connected, they will form a regular n-gon with V being the centre, as shown in the Figure 4.3.

The α and β in Equation 4.11 are two shape parameters. The resulting surface interpolates the data vertices when $\alpha = 1$ and approximates the data mesh when α is in other values. In my ϵ-G^1 scheme, α is always set to be 1. Parameter β defines the tangent length at V. To have a pleasing surface shape, Loop recommends α and β to be

$$\alpha = \frac{1}{9}\left[4+\cos(\frac{2\pi}{n})\right], \quad \beta = \frac{1}{3}\left[1+\cos(\frac{2\pi}{n})\right]. \tag{4.12}$$

The last two control points H_i^3 and H_i^4 are set by Equation 4.10 and 4.11 calculated on the neighbouring vertex V_i. The middle point H_i^2 is set as the average of the two centroid

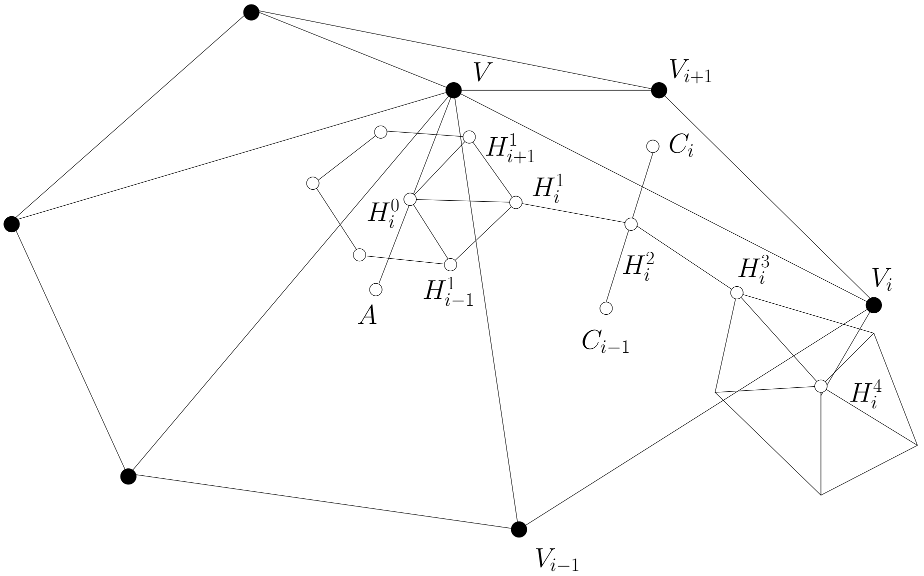


Figure 4.3: Boundary curve construction in Loop's scheme.

points C_i and C_{i-1} of the two adjacent data triangles:

$$H^2_i = \frac{C_{i-1}+C_i}{2} = \frac{V}{3} + \frac{V_i}{3} + \frac{V_{i+1}}{6} + \frac{V_{i-1}}{6}. \tag{4.13}$$

With the definition of $\phi_0 = \cos(2\pi/n)$, the twist terms at V for patch F_i are solved as

$$\begin{aligned}\frac{\partial^2 F_i}{\partial \vec{u}_i \partial \vec{u}_{i+1}} &= \frac{4}{n}\sum_{j=1}^{j=n}\left[\beta + \phi_0(-3+3\alpha-7\beta)\left(\cos\frac{2\pi(j-i)}{n} + \tan\frac{\pi}{n}\sin\frac{2\pi(j-i)}{n}\right)\right]V_j \\ &\quad +4\phi_0(1-3\alpha)V + 4\phi_0\left(V_{i+1}+V_{i-1}\right). \end{aligned} \tag{4.14}$$

The twist control points then can be calculated using Equation 4.14. Such a construction generates consistent mixed partial derivatives at V from all the surrounding patches [25].

4.3 Quintic approximate G^1 scheme

In Loop's scheme, the original quartic patch has to be degree elevated to sextic to fulfil the G^1 continuity constraints. In my approximate G^1 scheme, I first construct a quartic patch for each data triangle as an intermediate result. Such a quartic surface is actually approximate G^1 continuous and visually smooth on some models. To gain more freedom for better surface quality, a quintic patch is later constructed based on each quartic patch.

My quintic approximate G^1 surface fitting scheme has following steps:

1. Construct the quartic boundary curves for each data triangle, with a different middle control point setting than Loop's scheme.
2. Solve the twist terms and set the interior control points of the quartic patch.
3. Construct a quintic patch based on the quartic patch.
4. Adjust the quintic patch control points to reduce the normal discontinuity.

4.3.1 Quartic patch construction

In Loop's scheme, quartic control points H_i^0, H_i^1, H_i^3 and H_i^4 are created using Equations 4.10 and 4.11 with $\alpha = 1$. Figure 4.4 shows the construction of a boundary curve on an icosahedron. For the middle control point H_i^2, Loop's scheme sets it to be the average of C_{i-1} and C_i, the centroids of two adjacent data triangles (Equation 4.13). However, if the we need to interpolate the data vertices, *i.e.*, $V = H_i^0$, this construction results in a curve with an extra 'wiggle', as shown in Figure 4.4(a).

In my approximate G^1 scheme, I set H_i^2 differently to improve the surface shape, as shown in Figure 4.4(b). The new scheme obeys the following rules to set H_i^2, so that the twist terms can be solved locally.

1. Point H_i^2 must be the average of C_i and C_{i-1},
2. Point C_i must be set using only data at triangle VV_iV_{i+1}.

As an improvement over Loop's construction, I calculate a virtual point $\bar{V}$ for V and set H_i^2 as the average of $\bar{C}_{i-1}$ and $\bar{C}_i$, as shown in Figure 4.4(b), with $\bar{C}_i$ as the centroid of $\triangle\bar{V}\bar{V}_i\bar{V}_{i+1}$. All the virtual points of mesh $\mathcal{M}$ form a virtual mesh $\bar{\mathcal{M}}$. Ideally, a $\bar{\mathcal{M}}$ could be found so that the Loop's scheme can fit a surface to it, with α set by Equation 4.12.

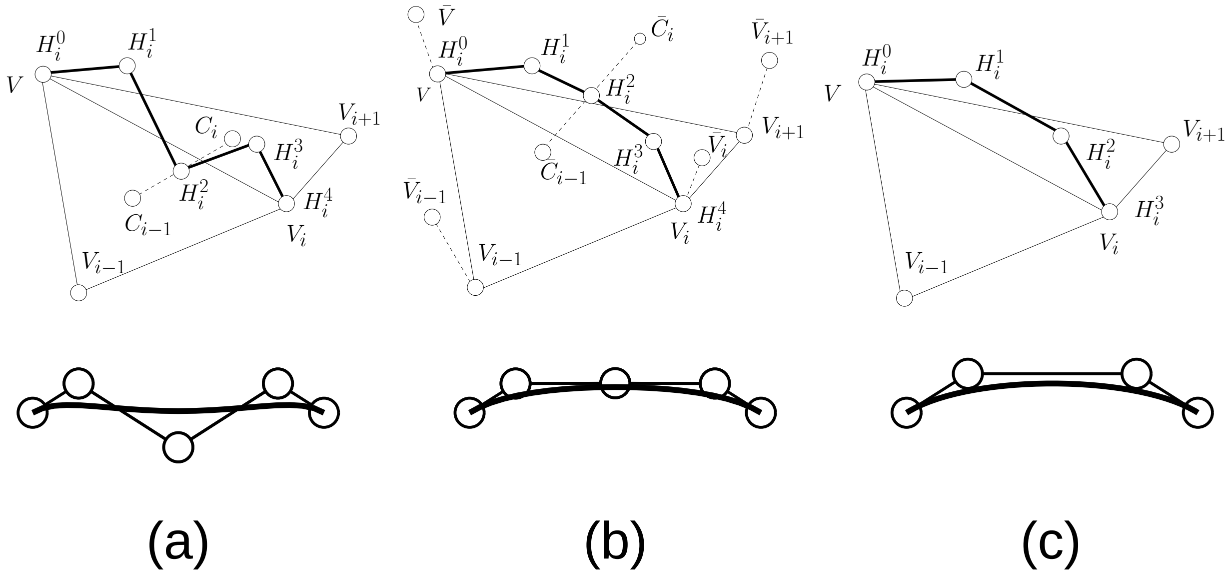


Figure 4.4: Comparison of different boundary curve constructions.

Therefore the resulting surface should interpolate all the data vertices of $\mathcal{M}$ and have a pleasing shape. With the introduction of $\bar{V}$, Equation 4.10 is rewritten to

$$\begin{aligned} V &= \alpha\bar{V} + \frac{1-\alpha}{n}\sum_{j=1}^{n}\bar{V}_j = \alpha\bar{V} + \frac{1-\alpha}{n}\bar{A} \\ \Longrightarrow \bar{V} &= V + \frac{1-\alpha}{\alpha}(V-A) + \frac{1-\alpha}{\alpha}(A-\bar{A}). \end{aligned} \tag{4.15}$$

Since $\mathcal{M}$ is of arbitrary topology, finding the solution for $\bar{V}$ from Equation 4.15 involves solving a global system, and the solution is not guaranteed. A lower cost approach is to introduce a new shape parameter γ, and construct $\bar{V}$ to be collinear with line VA (Figure 4.5):

$$\bar{V} = V + \gamma\frac{1-\alpha}{\alpha}(V-A). \tag{4.16}$$

The middle control point H_i^2 is then set as the average of the virtual centroids:

$$H_i^2 = \frac{\bar{C}_{i-1}}{2} + \frac{\bar{C}_i}{2} = \frac{\bar{V}}{3} + \frac{\bar{V}_i}{3} + \frac{\bar{V}_{i+1}}{6} + \frac{\bar{V}_{i-1}}{6}. \tag{4.17}$$

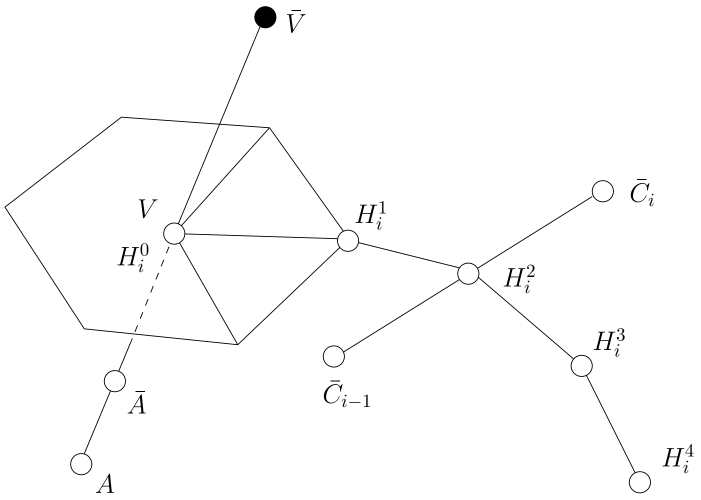


Figure 4.5: New construction of the middle control point.

As shown in the bottom of Figure 4.4(b), the quartic boundary curve with improved setting of H_i^2 does not have the wiggle of Loop's construction.

To further test the construction, I experimented with using a cubic boundary curve consisting of the four tangent control points constructed by Equations 4.10 and 4.11, as shown in Figure 4.4(c). The side view of this cubic curve shows similar shape as the improved quartic construction (Figure 4.4(b) and (c)). In Section 4.5, this cubic construction will be used to analyze the mesh optimization process of the tangent points construction in Loop' scheme, but the results presented there show the need to use a quartic bounary curve.

After creating the quartic boundary curves, now we build the intermediate quartic patch. From Equations 4.10 (with $\alpha = 1$), 4.11 and 4.17, all control points on the boundary curves in the quartic patch are set as

$$Q_{(4-k,k,0)} = H_i^k, \tag{4.18}$$

with $k = 0, 1, \ldots, 4$, shown as the small circles in Figure 4.6.

Points for the other two boundaries are obtained similarly by shifting the indices to the

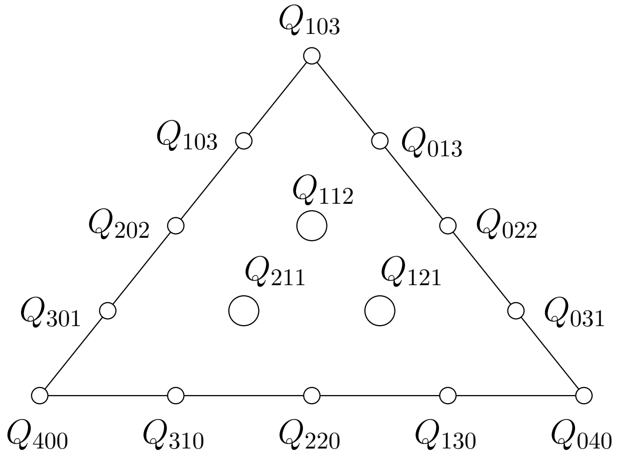


Figure 4.6: Quartic patch control points.

left once and twice. The twist control point Q_{211} is set as

$$Q_{211} = \frac{1}{12}\frac{\partial^2 F_i}{\partial \vec{u}_i \partial \vec{u}_{i+1}} - Q_{400} + Q_{301} + Q_{310}, \tag{4.19}$$

shown as a big circle in Figure 4.6. Here the twist term $\partial^2 F_i/(\partial \vec{u}_i \partial \vec{u}_{i+1})$ is calculated using Equation 4.2. Points Q_{121} and Q_{112} are calculated the same way with the twist term of the corresponding vertex and the control points with indices shifted to the left once and twice. The quartic patch is now G^1 continuous at the vertices, but not on other points along the boundary curves.

4.3.2 Quintic patch construction

The quartic patch construction in Section 4.3.1 generates a surface with large normal discontinuities along the common boundary between patches. Since there is not enough freedom left in the quartic patch to make further adjustments, the patch degree is raised to quintic and its control points are adjusted to reduce the normal discontinuity. The boundary control points of the quintic patch, shown as small circles in Figure 4.7, are calculated by a degree elevation from the quartic patch boundary points. For example, the

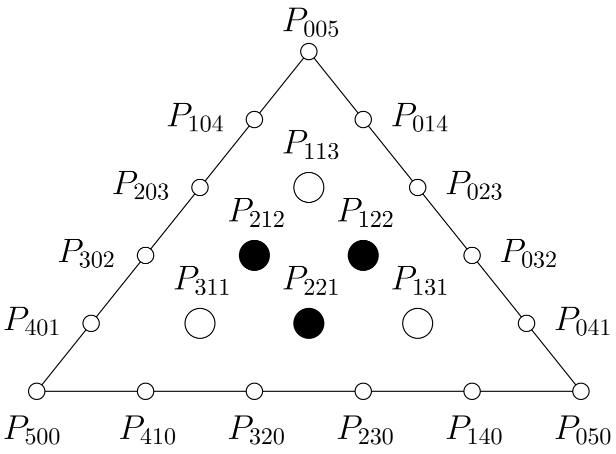


Figure 4.7: Quintic patch control points.

control points along the boundary of $P_{500}P_{050}$ are set as

$$P_{500} = Q_{400}, \qquad P_{5-k,k,0} = \frac{k}{5}Q_{5-k,k-1,0} + \frac{5-k}{5}Q_{4-k,k,0}, \tag{4.20}$$

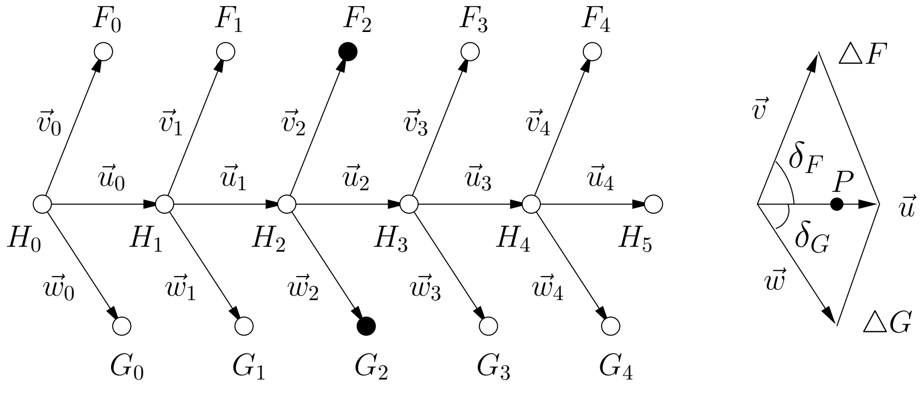
with $k = 1, 2, 3, 4$. The interior control point P_{311} defines the twist terms across neighbouring patches. I then set the twist terms in the quintic patch to be equivalent to those in the quartic patch:

$$P_{311} = \frac{3}{5}(Q_{211} - Q_{301} - Q_{310} + Q_{400}) + P_{401} + P_{410} - P_{500}. \tag{4.21}$$

There are still three control points unset in the quintic patch, indicated as filled circles in Figure 4.7. To limit the normal discontinuity, I set the positions of these three points by averaging the information of the control points available so far:

$$P_{221} = P_{320} + \frac{1}{2}(P_{311} - P_{410} + P_{131} - P_{230}). \tag{4.22}$$

This result is only an initial position for point P_{221}; Section 4.3.3 explains how to adjust it to reduce the normal discontinuity along the boundaries. The control points corresponding to other two boundaries are obtained by shifting the indices in Equations 4.20, 4.21, 4.22 to the left once and twice.



(a) (b)

Figure 4.8: Control vectors along a common boundary curve.

4.3.3 Adjustment to interior control points

Since the quintic surface constructed as in Section 4.3.2 is degree raised from the quartic surface of Section 4.3.1, it has large normal discontinuities along patch boundaries. To reduce the normal discontinuity along the boundaries, the control points are adjusted in a similar way as in Chapter 3; the difference is that here one quintic patch is used for each data triangle. The detailed analysis of discontinuity and the motivation of this adjustment will be presented in Chapter 5.

For two adjacent quintic patches F and G sharing common boundary H, the G^1 constraints only require two rows of control points of each patch being considered, as shown in Figure 4.8. To reduce the discontinuity, I will adjust F_2 and G_2 so that patches F and G have equal normals at additional interior points along the boundary (they already have equal normals at the two boundary end points). For the approximate scheme in this chapter, only the method for one additional equal-normal point is used. For a boundary curve point P corresponding to a parameter value t, we perform de Casteljau's algorithm on patches F and G individually until the last step, leaving the two tangent triangles $\triangle F$ and $\triangle G$ spanned by vectors $\vec{u}$, $\vec{v}$, $\vec{w}$, as shown in Figure 4.8(b). For the vectors shown in

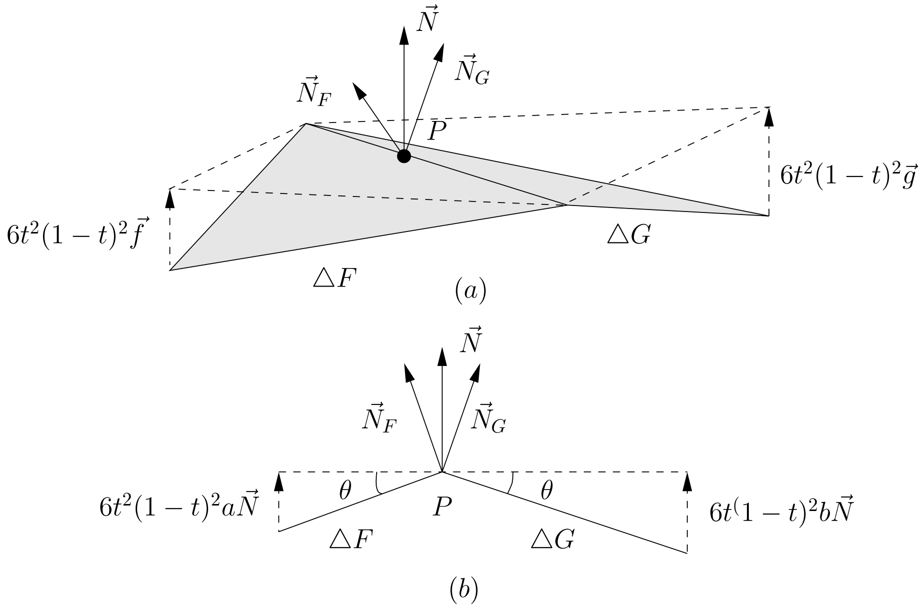


Figure 4.9: Adjustment of control points.

Figure 4.8(a), define

$$\vec{u}_i = H_{i+1} - H_i, \quad \vec{v}_i = F_i - H_i, \quad \vec{w}_i = G_i - H_i.$$

This defines $\vec{u}$, $\vec{v}$, $\vec{w}$ as quartic Bézier functions:

$$\vec{u} = \sum_{i=0}^{4} \vec{u}_i B_i^4(t), \quad \vec{v} = \sum_{i=0}^{4} \vec{v}_i B_i^4(t), \quad \vec{w} = \sum_{i=0}^{4} \vec{w}_i B_i^4(t).$$

The normal vectors of $\triangle F$ and $\triangle G$ are defined by the cross product of two tangent vectors:

$$\vec{N_F} = \frac{\vec{u} \times \vec{v}}{|\vec{u} \times \vec{v}|}, \qquad \vec{N_G} = \frac{\vec{w} \times \vec{u}}{|\vec{w} \times \vec{u}|}.$$

We define the angle spanned by $\vec{u}$ and $\vec{v}$ as δ_F, and the angle between $\vec{w}$ and $\vec{u}$ as δ_G (Figure 4.8 (b)). For the quintic patch constructed in Section 4.3.2, now we adjust F_2 and G_2 so that patches F and G meet with equal normals at one interior point on the boundary. Adding offset vectors, say $\vec{f}$ and $\vec{g}$, to F_2 and G_2 respectively, the vectors $\vec{v}$ and $\vec{w}$ in $\triangle F$ and $\triangle G$ then change to

$$\vec{v'} = \vec{v} + 6t^2(1-t)^2\vec{f}, \quad \vec{w'} = \vec{w} + 6t^2(1-t)^2\vec{g}.$$

After adjustments, we want $\vec{u}$, $\vec{v'}$, $\vec{w'}$ to be coplanar at P, that is

$$(\vec{v} + 6t^2(1-t)^2\vec{f}) \cdot ((\vec{w} + 6t^2(1-t)^2\vec{g}) \times \vec{u}) = 0. \tag{4.23}$$

The goal is to find $\vec{f}$ and $\vec{g}$ that satisfy Equation 4.23. Before adjustments, $\triangle F$ and $\triangle G$ (shown with gray color in Figure 4.9(a)) are not coplanar. After adjustments, $\triangle F$ and $\triangle G$ should be on the same plane defined by the new normal vector $\vec{N}$, as shown in Figure 4.9(a). A 2D side-view is presented in Figure 4.9(b), where $\triangle F$ and $\triangle G$ are projected into the plane perpendicular with $\vec{u}$. To distribute the adjustments evenly, set the target normal $\vec{N}$ to bisect $\vec{N_F}$ and $\vec{N_G}$:

$$\vec{N} = \frac{\vec{N_F} + \vec{N_G}}{|\vec{N_F} + \vec{N_G}|}.$$

If we set $\vec{f}$ and $\vec{g}$ to be parallel to $\vec{N}$ to minimize the vector length, then $\vec{f}$ and $\vec{g}$ become $\vec{f} = a\vec{N}$, $\vec{g} = b\vec{N}$. By similar triangles shown in Figure 4.9(b), the ratio between the vectors length is

$$\frac{a}{b} = \frac{|\vec{v}|\sin(\delta_F)\cos\theta}{|\vec{w}|\sin(\delta_G)\cos\theta} = \frac{|\vec{u} \times \vec{v}|}{|\vec{w} \times \vec{u}|}. \tag{4.24}$$

Substituting Equation 4.24 into Equation 4.23 to have

$$\begin{aligned} &\vec{v'} \cdot (\vec{w'} \times \vec{u}) = \vec{v} \cdot \vec{N_G} + 12t^2(1-t)^2 a\vec{N} \cdot \vec{N_G} = 0 \\ \Longrightarrow \quad & a = -\frac{|\vec{v}|\sin(\delta_F)\sin(\theta)}{6t^2(1-t)^2}, \quad b = -\frac{|\vec{w}|\sin(\delta_G)\sin(\theta)}{6t^2(1-t)^2}. \end{aligned} \tag{4.25}$$

Equation 4.25 shows that for two patches meeting with ε-G^1 continuity, we can have equal normals at an additional interior point P on the boundary by adjusting central control points. If the two patches already meet with equal normals at P with initial positions of F_2 and G_2, then $\sin(\theta) = 0$ and $a = b = 0$, and the adjustment leaves F_2 and G_2 unchanged. Chapter 3 shows that having equal normals at $t = 0.5$ reduces the normal discontinuity significantly and the results are close to the value of t that minimize the normal discontinuities. With $t = 0.5$, Equation 4.25 simplifies to

$$a = -\frac{8}{3}|\vec{v}|\sin(\delta_F)\sin(\theta), \qquad b = -\frac{8}{3}|\vec{w}|\sin(\delta_G)\sin(\theta). \tag{4.26}$$

4.4 Results

I applied my quintic approximate G^1 scheme on the triangular meshes sampled on an icosahedron, a torus and the Stanford bunny, whose meshes are shown in Figure 4.10. I compared the resulting surfaces to those created by Loop's sextic G^1 scheme. I also implemented Loop's interpolating scheme using the new boundary construction method (Section 4.3.1), so that the ε-G^1 scheme and Loop's interpolating scheme construct the boundaries the same way as shown in Figure 4.5; this allows for a more reasonable comparison between Loop's G^1 scheme, and our approximate G^1 scheme and is an improvement over Loop's original scheme.

As the first test, I wanted to see if I could use a cubic boundary curve instead of a quartic curve. Figures 4.11 shows two interpolating surfaces on the icosahedron model. The surface on the left has cubic boundary curves with the vertex tangents constructed the same as in Loop's scheme (Section 4.3.1). The surface on the right of Figures 4.11 uses quartic boundaries constructed as in Section 4.3 with shape parameter γ set to 0.75. The degree of boundary curves of both surfaces were later elevated to quintic. Both surfaces were constructed using the quintic approximate G^1 scheme with equal normals at the middle boundary point. As shown in Figures 4.11, the discontinuity along the cubic boundaries (left) is noticeable; the surface with quartic boundaries (right) is visually smooth. Surfaces with different cubic boundary curves may have improved continuity conditions than the left surface of Figures 4.11, but my experiments show that they are still not as good as the surfaces with quartic boundary curves. The problem with using cubic boundary curves is that the construction of the twist control points (F_1 and G_1 in Figure 4.8) depends on the middle control point H_2. Using a cubic boundary curve does not admit a solution with consistent mixed partial derivatives at the vertex, leading to worse surface shape. Therefore, I will use quartic boundary curves for the rest of this chapter.

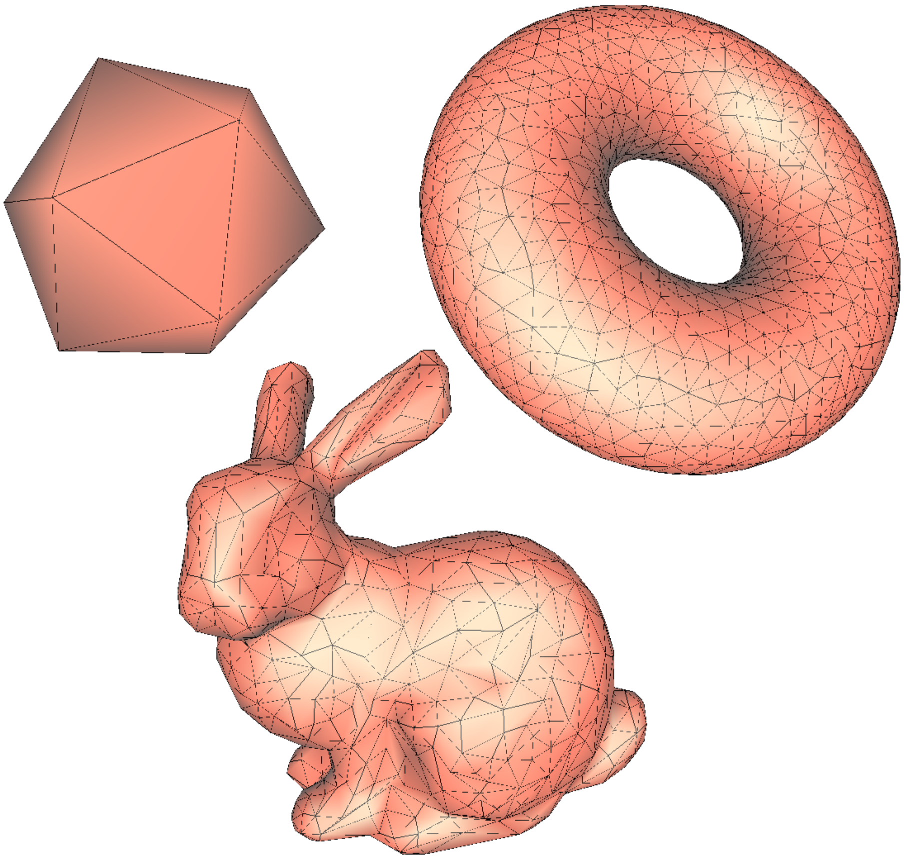


Figure 4.10: Data meshes.

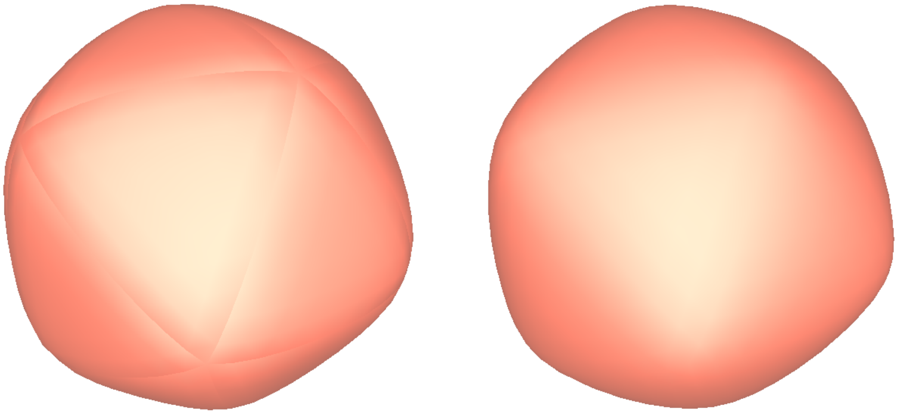


Figure 4.11: Icosahedral surfaces with cubic and quartic boundaries.

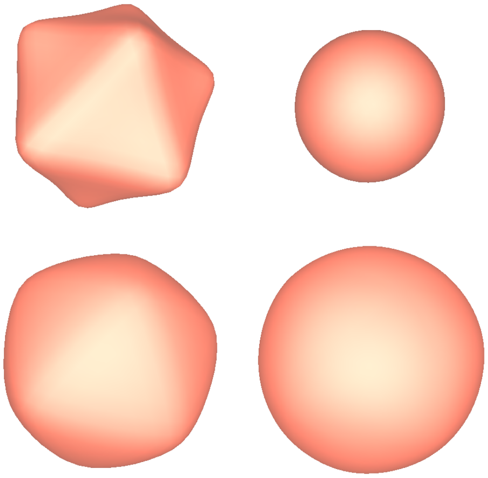


Figure 4.12: Icosahedral surfaces.

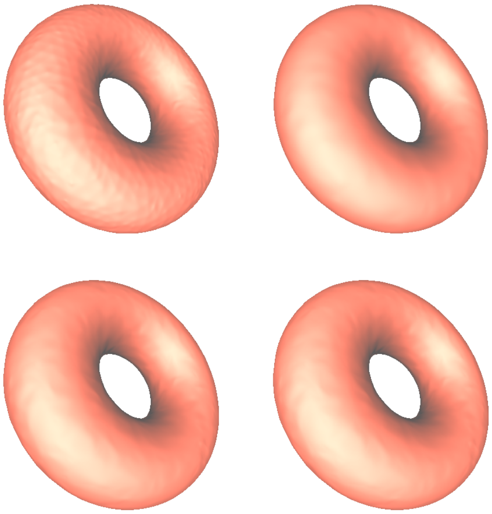


Figure 4.13: Torus surfaces.

In figures 4.12, 4.13, 4.14, 4.15 and 4.16, the surface on the upper left corner is generated by Loop's scheme with vertices interpolated; the upper right surface is the approximating result by Loop's scheme; the lower left surface is by Loop's interpolating scheme with the new boundaries; the lower right surface is my interpolating surface with approximate G^1 continuity. The shape parameter γ is set to 0.75 for the icosahedron in Figure 4.12, and 0.5 for all other models; these values were picked empirically.

The surface constructed for the icosahedron dataset by Loop's interpolating scheme with the original boundary construction shows poor shape quality, as shown in Figure 4.12. The approximating surface by Loop's scheme (upper right of Figure 4.12) has better shape, but the vertices are not interpolated. With the improved construction of boundary curves, Loop's scheme generates an interpolating surface with better shape (lower left of Figure 4.12), but not as good as the surface constructed by my scheme (lower right of Figure 4.12). With the maximum normal discontinuity lower than 0.1 degrees, we do not see a crease along patches boundaries on the surface constructed by my scheme.

As a second example, a torus surface was sampled randomly at 1371 vertices and the vertices were tessellated using a Voronoi triangulation method [1, 36], then surfaces were fit to this data whose mesh is shown in the upper right of Figure 4.10. The torus model used

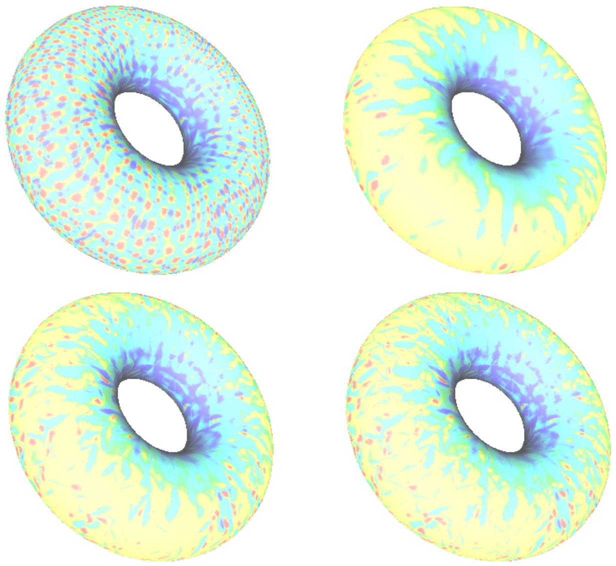


Figure 4.14: Gaussian curvature plots.

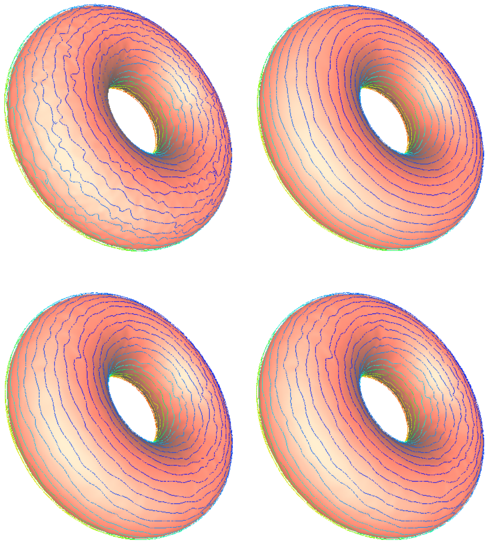


Figure 4.15: Isophotes lines of torus surfaces.

here has a major radius of 1 and a minor radius of 0.5. The maximum valence of a vertex in this mesh is 9, with the typical valence being 6. The ε-G^1 surfaces show the same level of visual smoothness as the revised interpolating surfaces and the approximating surface of Loop's scheme, with the maximum normal discontinuity lower than 0.7 degree.

To check the shape quality in more detail, I generated Gaussian curvature plots (Figure 4.14) and isophotes curves (Figure 4.15) for the resulting surfaces. In both plots, the surfaces constructed by the ε-G^1 scheme show similar shape as the surfaces constructed by Loop's scheme with revised boundaries. Although the isophotes shown on the surface that the ε-G^1 scheme constructed for the torus mesh look continuous in Figure 4.15, there are actually small breaks in these isophotes lines, because ε-G^1 surfaces are not G^1 continuous. Table 4.1 gives statistics about the gaps between the isophotes curves of the torus surface constructed by the ε-G^1 scheme. The greatest gap between isophotes segments is 0.066. Both schemes with revised boundaries show better shape than that of the original Loop interpolating scheme, but worse than the Loop's approximating scheme, which does not interpolate the data points.

The last model is the Stanford bunny, a triangular mesh with 752 vertices and 1500

Table 4.1: Isophotes gaps of torus surfaces.

Gaps	$(0, 10^{-3}]$	$(10^{-3}, 10^{-2}]$	$(10^{-2}, 10^{-1}]$
Number	2760	90	1
Percentage	97.81%	3.16%	0.03%

faces. As shown in Figure 4.16, my quintic ε-G^1 scheme generates a surface with the same level of smoothness as the revised Loop interpolating scheme. The highest normal discontinuity angle between two adjacent patches in ε-G^1 scheme is 65.7°. However, statistics show that boundaries with high normal discontinuity angles (more than 10 degrees) account for about 0.8% of all boundary curves (Table 4.2). The region having a discontinuity of 65.7° is circled in Figure 4.16 and Figure 4.17 shows a close up view of this region. Unlike the other figures, the upper left image in Figure 4.17 shows the data triangulation; the other three images are similar to the other figures. Among all the 27 edges with maximum normal discontinuity of more than 5 degrees, 22 edges are located around 5 vertices. As seen in the mesh in Figure 4.17, these vertices that cause problems for ε-G^1 scheme have anomalous locations, and cause modulations for most interpolating schemes, including Loop's. Table 4.2 shows that other than these particular areas, most parts of the ε-G^1 surface have low discontinuities.

Table 4.2: Discontinuity statistics of Bunny

Discontinuity	$[0, 5]$	$(5, 10]$	$(10, 20]$	> 20
Boundary Number	2223	12	10	5
Percentage	98.1%	1.1%	0.5%	0.3%

I have presented the construction of a piecewise quintic surface that interpolates the data positions and normals. The patches meeting at a data point have a consistent mixed partial derivative, and each pair of adjacent patches meet with ε-G^1 continuity along the common boundary. In summary, the new quintic approximate G^1 scheme in this chapter

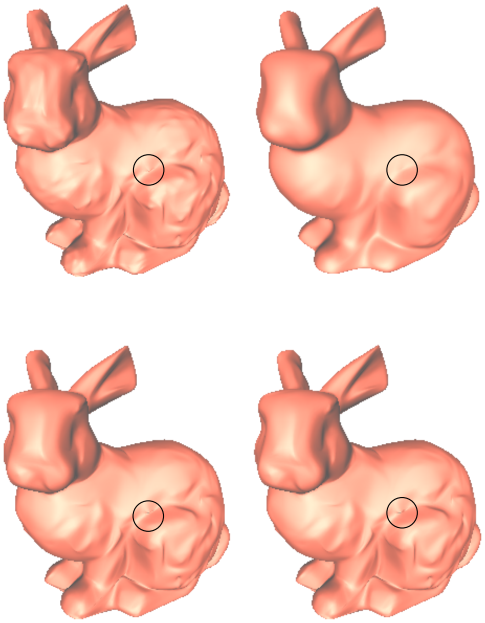


Figure 4.16: Bunny surfaces.

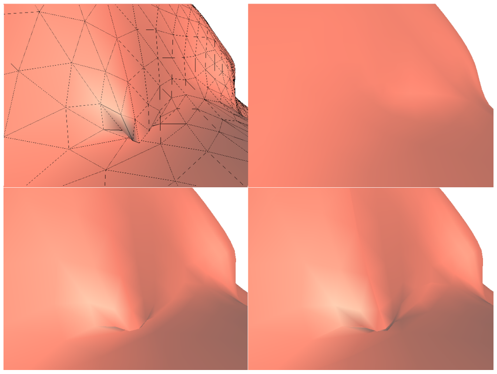


Figure 4.17: Close up view of bunny surfaces.

has following contributions:

- By using approximate continuity, I am able to reduce from six to five the degree of a triangular Bézier patch interpolant.
- By using a lower degree patch than Loop's scheme, the new scheme has fewer degrees of freedom remaining after achieving the interpolation and continuity conditions. This lower number of degrees of freedom avoids steps like the averaging used in step 6 of Loop's scheme (as mentioned in Section 4.2). The expectation is that this will result in a better shape, although such shape improvements may only appear in sparse data sets like the icosahedron of Figure 4.12.
- By constructing a virtual mesh, I was able to improve the shape of the surfaces constructed by Loop's scheme when it is used to interpolate data.

For all the models (icosahedron, torus and bunny) shown in figures 4.12, 4.13, 4.14, 4.15 and 4.16, the approximating surfaces displayed better shape quality than the interpolating surfaces created by the same scheme. The main reason for such a difference is that both Loop's scheme and my quintic ϵ-G^1 scheme optimize the data meshes for approximating surfaces. This translation process filters out the "bad shape features" on the original data mesh and generates surfaces with better shape as further discussed in the next section.

4.5 Data mesh optimization

To confirm the smoothing effect of transforming data vertices, I applied another two surface schemes with domain splitting to the optimized data meshes: Shirman-Séquin's G^1 quartic scheme [38] and an ϵ-G^1 cubic scheme [23]. I used these two schemes to construct both interpolating and approximating surfaces on the bunny model (experiments on other models lead to the same conclusion). The cubic scheme is based on the scheme introduced in Chapter 3, with the tangent planes at each data vertex constructed the same way as in Loop's scheme and the quintic ε-G^1 scheme. Each cubic boundary curve is then built using the tangent points at both ends of the boundary. After the three micro-patches are constructed, the center control points are adjusted to have more equal-normal points along each boundary to reduce the cross boundary discontinutiy.

In Figure 4.18, the surfaces in the left column are interpolating surfaces and those in the right column are approximating surfaces. The two surfaces in the first row are constructed using Loop's sextic scheme; the second row by Shirman-Séquin's quartic scheme; the third row by the cubic ϵ-G^1 scheme. For Loop's interpolating surface, I did not use the optimization on the middle control point of the boundary curve. Since Shirman-Séquin's is an interpolating scheme, the right surface in the second row is generated on a "new" model that is composed of data vertices after optimization (Equation 4.10), *i.e.*, applying the interpolating scheme to the approximating data set. The three approximating surfaces in the right column of Figure 4.18 interpolate the same set of data vertices.

For the interpolating surfaces, both Shirman-Séquin's and the cubic ϵ-G^1 scheme generated numerous shape artifacts, but the shape quality of the two approximating surfaces is much better than that of the corresponding interpolating surfaces. This observation confirms that the transformation of data vertices has a great impact on the surface shape quality. Moreover, if we compare the surfaces of Shirman-Séquin's scheme and the cubic ϵ-

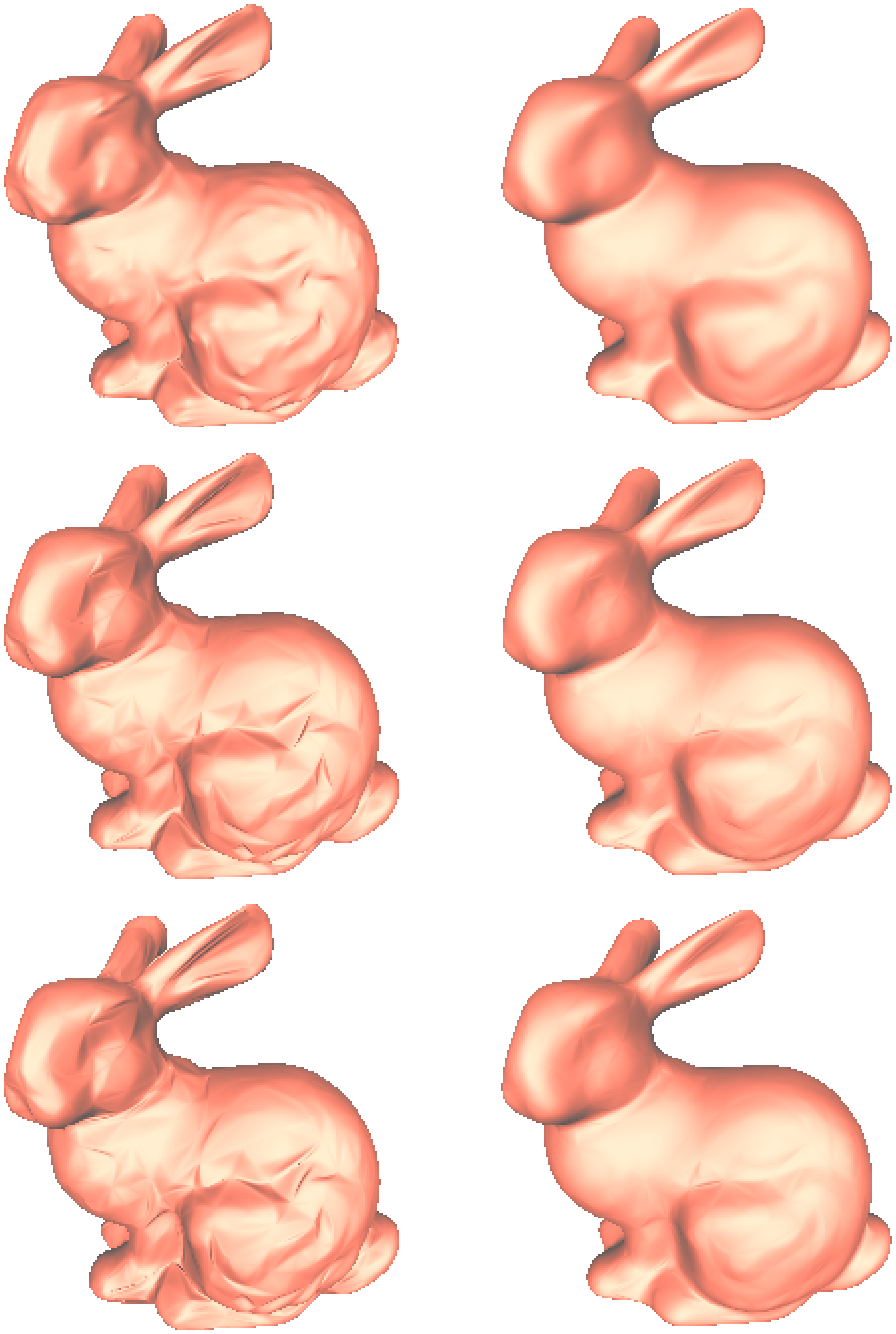


Figure 4.18: Influence of mesh optimization on surface quality.

G^1 scheme, they have the same shape quality for both interpolating and the approximating surfaces. This experiment shows that the fulfillment of G^1 continuity constraints is not a prerequisite condition to construct surfaces with good shape. When building surfaces with approximate continuity, we should integrate techniques such as data mesh optimization and better boundary curve construction to create surfaces with better surface shape.

Chapter 5

Discontinuity bound for parametric surfaces

On surfaces with ϵ-G^1 continuity, we allow a small amount of discontinuity between normals across patch boundaries. To guarantee the maximum discontinuity angle within the tolerance of an application, we need to find the upper bound for the discontinuity. This discontinuity upper bound should be calculated solely using the control points from adjacent patches. By analyzing the definition of this upper bound, we can also develop methods to reduce the discontinuity.

5.1 Definitions

I define the tangent points and vectors in this section similarly to those used in chapters 3 and 4. The only difference is that I use degree n patches as the general case in this chapter. For two degree n triangular Bézier patches F and G meeting with ϵ-G^1 continuity, only the boundary control points H_i and the second row of control points from each patch are considered, as shown in Figure 5.1(a). For a given point P on the boundary curve, we perform de Casteljau's algorithm on patches F and G individually. At the last step of the evaluation, we have two tangent triangles $\triangle F$ and $\triangle G$ spanned by vectors $\vec{u}$, $\vec{v}$, $\vec{w}$, as

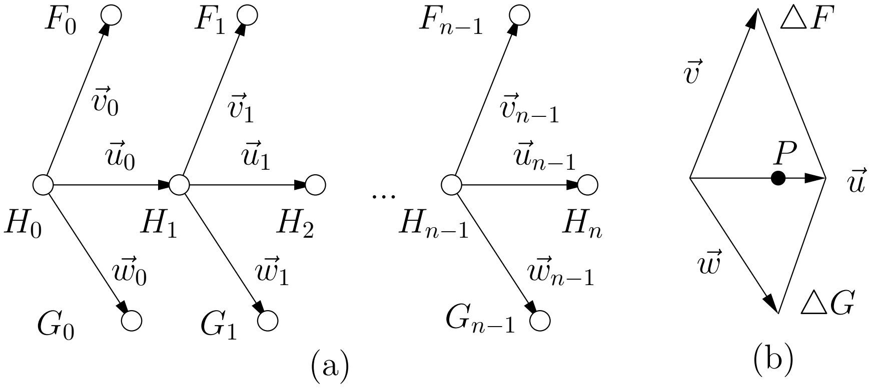


Figure 5.1: Definition of tangent vectors

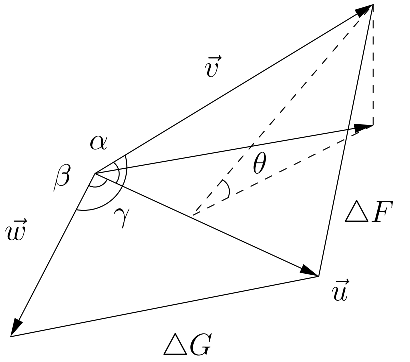


Figure 5.2: Definition of the discontinuity angle

shown in Figure 5.1(b). For the vectors shown in Figure 5.1(a), if we define

$$\vec{u}_i = H_{i+1} - H_i, \quad \vec{v}_i = F_i - H_i, \quad \vec{w}_i = G_i - H_i,$$

then the tangent vectors $\vec{u}$, $\vec{v}$, $\vec{w}$ are degree $n-1$ Bézier functions:

$$\vec{u} = \sum_{i=0}^{n-1} \vec{u}_i B_i^{n-1}(t), \qquad \vec{v} = \sum_{i=0}^{n-1} \vec{v}_i B_i^{n-1}(t), \qquad \vec{w} = \sum_{i=0}^{n-1} \vec{w}_i B_i^{n-1}(t).$$

Another view of $\triangle F$ and $\triangle G$ and the corresponding angles are shown in Figure 5.2. The angle θ between the normals of $\triangle F$ and $\triangle G$ is the discontinuity angle that we need to bound from above. In Chapter 2, the discontinuity angle was calculated by calculating its sine:

$$\theta = \arcsin\left(\frac{|(\vec{w}\times\vec{u})\times(\vec{u}\times\vec{v})|}{|\vec{w}\times\vec{u}|\;|\vec{u}\times\vec{v}|}\right). \tag{5.1}$$

In Equation 5.1, each of the normals is defined as the cross product of two tangent vectors. Because this was the functional case, the xy-coordinates of all the control points were distributed uniformly in the plane. For each pair of tangent vectors that define a surface normal, I decomposed them along the xyz-axes and calculate the cross product of two components along each axis independently. By calculating the max/min length of normals in Equation 5.1, I computed an upper bound for the discontinuity angle for functional surface patches meeting with approximate G^1 continuity.

In the parametric case, we cannot use the same method as in the functional case to calculate the upper bound using Equation 5.1. Since the topology of parametric surfaces is arbitrary, the normals cannot be decomposed uniquely and therefore the max/min length of the normals cannot be calculated as in the functional case in Chapter 2. To have a definition of θ that is easier to bound, I use $\arctan\theta$ instead of $\arcsin\theta$. As shown in Figure 5.2, after projecting $\vec{v}$ to the plane defined by $\vec{w}$ and $\vec{u}$, the discontinuity angle θ is defined as

$$\begin{aligned}\theta &= \arctan\left(\frac{|\vec{v}\cdot(\vec{w}\times\vec{u})|\;|(\vec{u}\times(\vec{u}\times\vec{w}))|}{|\vec{v}\cdot(\vec{u}\times(\vec{u}\times\vec{w}))|\;|(\vec{w}\times\vec{u})|}\right)\\ &= \arctan\left(\frac{|\vec{v}\cdot(\vec{w}\times\vec{u})\;|\;|\vec{u}|}{|(\vec{u}\cdot\vec{v})(\vec{u}\cdot\vec{w})-(\vec{v}\cdot\vec{w})(\vec{u}\cdot\vec{u})|}\right).\end{aligned} \tag{5.2}$$

In Equation 5.2, each term in the dot product between two tangent vectors is a scalar valued, degree $2(n-1)$ spline. The term $\vec{v}\cdot(\vec{w}\times\vec{u})$ is a scalar valued, degree $3(n-1)$ spline. To avoid the denominator being zero, we make following assumptions:

1. Angles α and β are not zero.
2. The planes of $\triangle F$ and $\triangle G$ are not perpendicular to each other.

As shown in Figure 5.2, α is spanned by vectors $\vec{v}$ and $\vec{u}$, and β by $\vec{w}$ and $\vec{u}$. The first assumption is easy to fulfil as long as there is no singular data triangle. For the second assumption, we assume all the boundaries are constructed free of "razor" edges, *i.e.*, the value of $\vec{v}\cdot(\vec{u}\times(\vec{u}\times\vec{w}))$ never changes sign along the boundary. Since we require the interpolation of normals at the data vertices, the two pairs of end panels are assumed to be coplanar, therefore $\vec{v}\cdot(\vec{u}\times(\vec{u}\times\vec{w}))\neq 0$ at the two end points. Therefore we have $\vec{v}\cdot(\vec{u}\times(\vec{u}\times\vec{w}))\neq 0$ for all the points along the boundary, which means $\theta\in[0,\pi/2)$.

5.2 Upper bound one

Since all the terms of the dot product in Equation 5.2 are scalar valued splines, an upper bound of θ can be calculated using the convex hull property of splines.

Theorem 5.2.1. *For parametric triangular Bézier patches F and G as defined in Section 5.1, the upper bound of the discontinuity angle θ is given by*

$$\theta \leq \arctan\left(\frac{\max(|\vec{v}\cdot(\vec{w}\times\vec{u})\,|)\ \max(\sqrt{|\vec{u}\cdot\vec{u}|})}{|\min(|\vec{u}\cdot\vec{v}|\ |\vec{u}\cdot\vec{w}|)-\min(|\vec{v}\cdot\vec{w}|\ |\vec{u}\cdot\vec{u}|)|}\right). \tag{5.3}$$

The theorem follows directly from Equation 5.2. In the remaining part of this thesis, I use the acronym of PB1 to denote the upper bound defined by Equation 5.3. All the max/min values in Equation 5.3 are computed separately. The calculation of PB1 (parametric surface upper bound 1) solely depends on the configuration of tangent vectors.

5.3 Upper bound two

Calculating the upper bound using the convex hull property of spline functions can only offer a coarse estimate of the discontinuity angle. To calculate a tighter upper bound, Equation 5.2 is further simplified. After cancelling the vectors' lengths both in the numerator and the denominator, we have

$$\begin{aligned}\theta &= \arctan\left(\frac{|\vec{v}\cdot(\vec{w}\times\vec{u})\,|\ |\vec{u}|}{|(\vec{u}\cdot\vec{v})(\vec{u}\cdot\vec{w})-(\vec{v}\cdot\vec{w})(\vec{u}\cdot\vec{u})|}\right)\\ &= \arctan\left(\frac{|\hat{v}\cdot(\hat{w}\times\hat{u})\,|}{|\cos\alpha\cos\beta-\cos\gamma|}\right).\end{aligned} \tag{5.4}$$

Here $\hat{u}$, $\hat{v}$ and $\hat{w}$ are normalized vectors. Angle α is spanned by $\vec{v}$ and $\vec{u}$, β by $\vec{w}$ and $\vec{u}$, and γ by $\vec{v}$ and $\vec{w}$,as shown in Figure 5.2. To have a tighter upper bound than Equation 5.3, we need to find a tighter upper bound for the numerator and a tighter lower bound for the denominator in Equation 5.4 individually.

For the denominator of Equation 5.4, its value only depends on the three angles α, β and γ. As the sampling rate is increased, these angles do not change much. Therefore the denominator of Equation 5.4 plays little part in finding a tighter upper bound for the angle θ.

Theorem 5.3.1. *For parametric triangular Bézier patches F and G as defined in Section 5.1, an upper bound on the discontinuity angle θ is given by*

$$\theta \leq \arctan\left(\frac{\max(|f^{(k)}(t)|)h^k \ \max(\sqrt{\vec{u}\cdot\vec{u}})}{4k\left(|\min(|\vec{u}\cdot\vec{v}| \ |\vec{u}\cdot\vec{w}|) - \min(|\vec{v}\cdot\vec{w}| \ |\vec{u}\cdot\vec{u}|)|\right)}\right). \tag{5.5}$$

Proof. For the numerator of Equation 5.4, we define a scalar valued function $f(t)$ as

$$f(t) = \vec{v}\cdot(\vec{w}\times\vec{u}), \qquad t\in[0,1]. \tag{5.6}$$

Since vectors $\vec{v}$, $\vec{w}$ and $\vec{u}$ are all degree $n-1$ polynomials, the function $f(t)$ is a degree $3(n-1)$ Bézier polynomial. Now we use a degree $k-1$, $k>1$, Lagrange polynomial $p(t)$ to interpolate $f(t)$ at $0=t_0<t_1\cdots<t_{k-1}=1$, to have $p(t_i)=f(t_i)$, $i=0,\ldots,k-1$. The error estimate of such an interpolation is [8, 35]

$$|f(t)-p(t)| \leq \frac{\max(|f^{(k)}(t)|)h^k}{4k}.$$

Here $h=\max(|t_{i+1}-t_i|)$, the maximum parameter interval. The function $f(t)$ must be differentiable to at least order k. If $f(t)$ and $p(t)$ are both zero at these points, *i.e.*, $p(t_i)=f(t_i)=0$, $i=0,\ldots,k-1$, then we have k zero-valued equations for the degree $k-1$ polynomial $p(t)$, which leads to

$$p(t)\equiv 0.$$

The error estimate then turns into the upper bound of $|f(t)|$ [35]:

$$|f(t)-p(t)| = |f(t)| \leq \frac{\max(|f^{(k)}(t)|)h^k}{4k}. \tag{5.7}$$

Here k represents the number of equal-normal points along the boundary. Replacing the corresponding term in Equation 5.3 with Equation 5.7, we have the second upper bound for θ, named PB2. The proof of this theorem is done by combining all the equations in this section with Equation 5.2. Here all the terms other than $f(t)$ are calculated the same way as in PB1. $\square$

The Taylor series of $\arctan(x)$ is

$$\arctan(x) = x - \frac{x^3}{3} + \frac{x^5}{5} - \frac{x^7}{7} + \cdots = \lim_{n\to\infty} \frac{(-1)^n x^{2n+1}}{2n+1}.$$

Thus, for small angles, we should expect θ to converge as $O(h^k)$.

5.4 Discontinuity reduction

In the surface fitting schemes introduced in Chapters 3 and 4, I adjusted the control points to reduce the discontinuity across each boundary. With the upper bounds developed in this chapter, now I can explain why this method works. In Equation 5.5, with other terms being constant, we can have a smaller value of θ by

1. Increasing the sampling rate,
2. Increasing k, the number of equal-normal points.

Reducing the discontinuity by increasing the sampling rate is not always possible especially when the surface being sampled is unknown. But we can generate more equal-normal points by adjusting the control points. Since we require that the data normals are interpolated, the two end points have equal normals before the adjustment, therefore $k = 2$. The extra equal-normal points should be chosen at the interior of the boundary curve. In Chapter 3 and 4, I discussed how to adjust the control points of an ε-G^1 surface to reduce the normal discontinuity. After the adjustments, we can have up to three additional equal-normal points on each boundary curve. In practice, I only used the one equal-normal point method. The parameter value t of this extra equal-normal point is set to be $t = 0.5$ to make a uniform partition.

Table 5.1: Upper bounds before adjustments.

Sample	Max	PB1	PB2	R1	R2
10	1.612928	1.799921	2.530955	1.12	1.57
20	0.221054	0.260774	0.257245	1.18	1.12
30	0.065661	0.078249	0.070457	1.19	1.07
40	0.028203	0.033712	0.029404	1.20	1.04
50	0.014293	0.017116	0.014675	1.20	1.03
60	0.008388	0.010049	0.008549	1.20	1.02
70	0.005229	0.006266	0.005951	1.20	1.14
80	0.003561	0.004269	0.003639	1.20	1.02

Table 5.2: Upper bounds after adjustments.

Sample	Max	PB1	PB2	R1	R2
10	0.117087	0.235954	0.295882	2.01	2.53
20	0.008262	0.015297	0.016588	1.85	2.00
30	0.001617	0.002941	0.002973	1.82	1.84
40	0.000485	0.000866	0.000787	1.79	1.62
50	0.000201	0.000357	0.000318	1.78	1.58
60	0.000098	0.000174	0.000157	1.78	1.60
70	0.000052	0.000093	0.000082	1.79	1.57
80	0.000031	0.000054	0.000046	1.74	1.50

5.5 Results

To test the upper bounds, I applied the ε-G^1 cubic interpolation scheme introduced in Chapter 3 on the data meshes sampled from a regular torus model. To reduce the discontinuity, the center control points of the micro patches are also adjusted to have equal normals at the middle point of each boundary.

The resulting surfaces have G^1 continuity at all the data vertices, approximate G^1 continuity on other boundary points and C^1 continuity along the internal boundaries. The sampling rate of the torus surfaces varies from 10 to 80 data triangles per ring and tube. Table 5.1 shows the discontinuity angles of the surfaces before the adjustments on control points, and Table 5.2 shows the data after the adjustments. All the angles in Tables 5.1 and 5.2 are in degrees. For each boundary curve, I sampled surface normals numerically at 100 points, and calculated the discontinuity angle between the two normals. The greatest discontinuity angle sampled on each surface is shown in the column "Max" in tables 5.1 and 5.2. Although this maximum angle is not the actual greatest discontinuity angle, it meets the precision requirement of my experiments. I used the maximum angle to measure the tightness of the upper bounds. For the boundary curve where the maximum angle is recorded, I also calculated the two upper bounds together with the ratios of them over the maximum angle, shown as "R1" and "R2" in Table 5.1 and 5.2. In Table 5.1 and 5.2, the step size in the sampling rate is 10, while in Figure 5.3, the step size is 2. For the plotted curves in Figure 5.3, I use solid line to depict the curve of the maximum angle, dashed line for PB1, and dotted line for PB2. The three curves in the right of Figure 5.3 are generated with the results before the adjustments to reduce the discontinuity, and the three curves in the left are plotted for the surfaces after the adjustments.

Before the adjustments to have equal normals at the middle boundary point, the value of PB2 is greater than that of PB1 for low sampling rates, as shown in right of Figure 5.3. As we increase the sampling rate, PB2 converges more quickly and soon gets smaller than PB1. Equation 5.5 shows that the convergence speed of the upper bound should be order of $O(h^k)$. For each boundary curve, we have equal normals at the two end points before the adjustments, *i.e.*, $k = 2$, therefore the convergence speed should be $O(h^2)$. However, in the experiments, the convergence speed of the discontinuity angles, including the maximum angle and two upper bounds, relative to the sampling rate are observed to be $O(h^3)$.

The discontinuity angle and the upper bounds were re-calculated after the adjustments and the plotted curves are shown in the left in Figure 5.3. After the adjustments, there are three equal-normal points along each boundary curve and values of PB1, PB2 and the max angle dropped tremendously for all surfaces at different sampling rates. Moreover, the convergence speed of the angles (sampled angle, PB1 and PB2) after adjustments is observed to $O(h^4)$, although Equations 5.5 shows that it should be $O(h^3)$ (for $k = 3$). After adjustments, the value of PB2 is greater than PB1 at low sampling rates. As the sampling rate is increased, PB2 converges faster than PB1 and soon has smaller values than PB1, as shown in the left of Figure 5.3. It is worth noting that usually the discontinuity is not

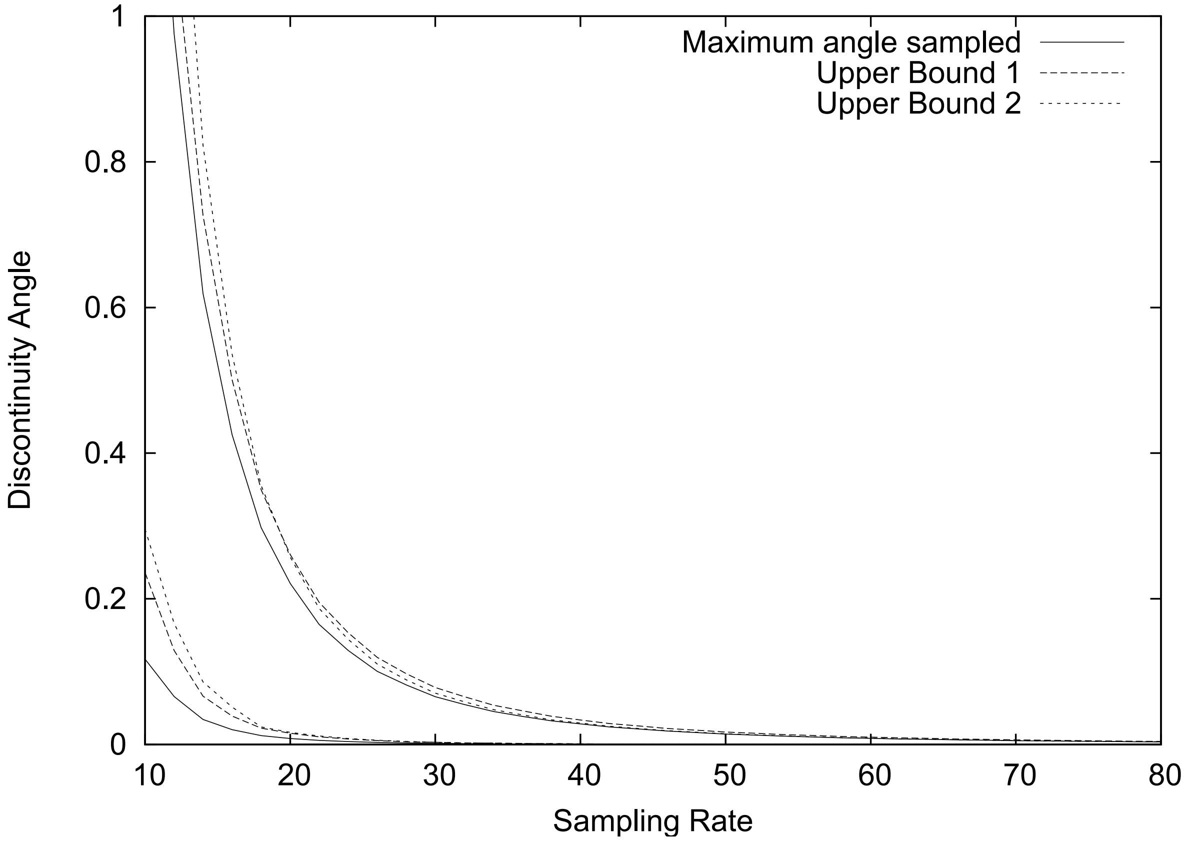


Figure 5.3: Upper bounds plot.

observable when the discontinuity angle is lower than one degree.

I also applied PB1 and PB2 to the functional surfaces introduced in Chapter 2 and compared the results with the normal discontinuity upper bounds on functional surfaces *i.e.*, FB1 and FB2. In Table 5.3, columns "PB1" and "PB2" show the results of the two upper bounds introduced in this chapter; all other data are the same as in the Table 2.1. Generally, the two parametric upper bounds are larger than the functional bounds. For most surfaces, the optimal (smallest) parametric bound is greater than the optimal functional bound, except for some surfaces where the discontinuity is small and all the bounds have similar values. Functional bounds on these surfaces are tighter because they use properties of a functional surface to greatly simplify the bounding equations (Chapter 2).

Table 5.3: Discontinuity of Franke function surfaces.

Functions	Grid	Discontinuity	FB 1	FB 2	PB 1	PB 2
F1	10×10	26.487492	67.894637	35.814840	85.634356	84.765676
	20×20	8.254011	11.494557	8.597684	18.127269	15.259764
	40×40	1.226258	1.469088	1.266205	1.997695	1.664952
F2	10×10	8.059139	13.315756	11.347997	15.449778	12.970033
	20×20	2.676044	3.660120	3.431178	4.380602	3.652674
	40×40	0.438722	0.515085	0.541983	0.618069	0.515064
F3	10×10	1.430003	1.630511	1.498616	2.283164	1.902944
	20×20	0.231934	0.240315	0.235121	0.296013	0.246678
	40×40	0.030659	0.030940	0.030764	0.037127	0.030939
F4	10×10	0.512760	0.545877	0.519773	0.655014	0.545852
	20×20	0.069462	0.070669	0.069708	0.084802	0.070669
	40×40	0.008862	0.008901	0.008870	0.010681	0.008901
F5	10×10	5.426008	9.497992	6.355084	12.080647	10.112753
	20×20	0.988369	1.235251	1.039557	1.491006	1.242591
	40×40	0.137503	0.146953	0.139415	0.176416	0.147013
F6	10×10	0.432433	0.503140	0.506612	0.603738	0.503121
	20×20	0.081190	0.089400	0.097062	0.107280	0.089400
	40×40	0.013420	0.014184	0.016166	0.017021	0.014184

Chapter 6

Conclusions

6.1 Contributions

When constructing surfaces using triangular Bézier patches to interpolate data meshes, control points are constrained to satisfy continuity conditions. Strict constraints require more control points, therefore we have to use high degree patches. Setting these extra degrees of freedom in high degree patches during surface construction is complicated, and the shape quality of resulting surfaces is sometimes unsatisfying.

The question is, do we have to satisfy continuity conditions to have a visually smooth surface with good shape? For many applications, what we need is a visually smooth surface, not necessarily meeting strict C^1/G^1 conditions. Moreover, fulfillment of continuity conditions is also not a prerequisite of having a good shape quality (Section 4.4).

The objectives of using approximate continuity is to relax the continuity constraints so as to use low degree patches and simple constructions without introducing too many extra control points. We also expect approximately continuous surfaces to have shape quality as good as the high degree surfaces with C^1/G^1 continuity. I first applied the idea of approximate continuity to interpolate functional data (Chapter 2). The cross boundary C^1 constraints were simplified and an upper bound of the discontinuity angle between surface normals was formulated. To test the idea, a simple scheme was designed to fit surfaces to a data mesh (Chapter 2). In this functional scheme, only one cubic functional patch is used for each data triangle. The resulting surfaces displayed the same level of smoothness as the G^1 scheme using three cubic patches per data triangle [28]. Although I did not expect this scheme to replace existing C^1 schemes, it provides a good example that it is possible to use a simple patch construction to generate visually smooth surfaces, sometime with a better shape quality than C^1 schemes.

However, the smooth surfaces constructed by the ε-C^1 scheme in Chapter 2 may be due to the simplicity of the functional data sets. The uniform distribution of control points in the plane helps limit the normal discontinuity, especially when the ε-C^1 scheme uses local averaging methods to build patches. So I looked at parametric surfaces to further test the idea of approximate continuity. For parametric surfaces, using only one cubic patch cannot provide enough freedom to maintain acceptable smoothness along all patch boundaries. Therefore I performed a Clough-Tocher like domain split on each data triangle, with each sub-patch affecting one corresponding macro-boundary (Chapter 3). This ε-G^1 scheme provides a cubic solution for arbitrary data sets, including those without a cubic G^1 solution. The resulting surfaces are as smooth as Shirman-Séquin's quartic scheme, and have better shape quality than some cubic schemes for nearly singular data sets.

The drawback of using a Clough-Tocher split is that the sub-patches can be long and narrow, generating unwanted undulations on the surfaces. Although this problem happens to all domain split schemes, it is worth further analysis to devise a better ε-G^1 scheme without a domain split. To use one patch for each data triangle, a higher degree patch is required to allow more freedom and specially designed boundary curves for optimized surface shape. In Loop's scheme, the optimization on the data meshes helps to generate surfaces with high smoothness. My quintic ε-G^1 scheme uses the same method, together with a revised configuration of the middle point of each quartic boundary curve (Chapter 4). Thus one patch ε-G^1 scheme constructs surfaces using quintic patches with quartic boundary curves. The resulting surfaces exhibit the same level of smoothness as G^1 continuous sextic schemes for data sets from regular primitives such as the torus and octahedron, and on the widely used data model of the Stanford bunny.

For both of my parametric schemes, the center control panel along the boundary curve can be adjusted so as to have equal normals at the middle point. This adjustment can be applied on any ε-G^1 surface as long as it only affects the continuity along one boundary curve and the adjustment is limited to the center panel of the boundary. The motivation of this adjustment is from the analysis of the upper bound of the discontinuity angle between surface normals (Chapter 5).

In summary, the essential idea of approximate continuity is to relax the strict C^1/G^1 constraints and manipulate control points to reduce the normal discontinuity and improve surface shape. Surface fitting with approximate continuity is not merely a removal of the continuity conditions, it includes different techniques to guarantee that the discontinuity is under control. The maximum discontinuity for surfaces is bounded from above.

6.2 Future work

I have shown the potentials of constructing surfaces with the approximate continuity, but there are still many possible improvements remaining. The first possible improvement is to construct better boundary curves. An interpolating surface with good shape quality depends heavily on the construction of good boundary curves. This has been identified as a key issue for all surface fitting schemes [27]. In this thesis, all my ε-C^1/ε-G^1 schemes use the same boundary construction as the C^1/G^1 schemes, requiring the interpolation of the positions and normals at data vertices. To make the construction simple, the boundary curve is usually built as a planar curve. In Loop's scheme and my ε-G^1 scheme in Chapter 4, the tangent control points are created as a regular "n-gon" at the data vertices and all the boundary curves are not coplanar. The resulting surfaces show that these 3D boundary curves can lead to better surface shape than the traditional 2D curves. In the future, designing boundary curves for better surface shape of ε-C^1/ε-G^1 continuous isurfaces should be studied.

For the parametric scheme introduced in Chapter 4, a quartic patch is first created for each data triangle and later degree elevated to quintic to create more freedom to adjust the control points. The quartic patch is created using a revised boundary curve construction of Loop's G^1 scheme, interpolating the twist terms calculated for G^1 conditions, as shown in the Figure 6.1. Although setting the twist terms the same as Loop's scheme helps to improve the shape quality for my quintic ε-G^1 scheme, it is worth mention that a solution to the twist compatibility problem is not required in an ε-G^1 scheme. In the future work, it would be good to set the twist points optimized for better surface quality other than

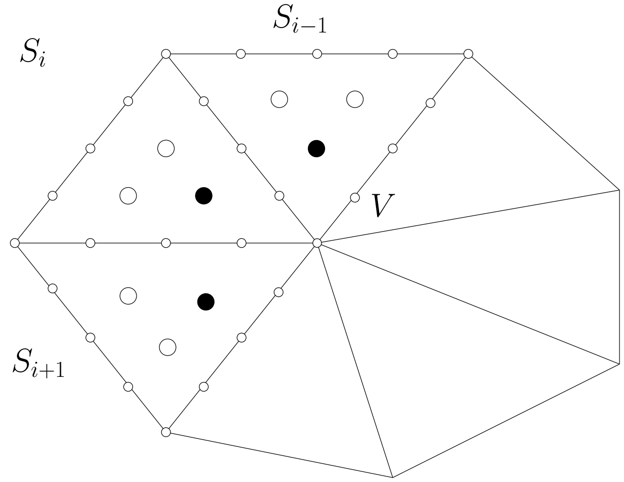


Figure 6.1: Quartic patches surrounding a vertex.

interpolating twist terms calculated for ε-G^1 continuous surfaces.

In this dissertation, all the surface fitting schemes use ε-C^1/ε-G^1 continuity, but no higher continuity conditions. Actually, the surface shape is also closely related to the curvature distribution on the resulting surfaces, therefore approximate continuity of higher orders should also be analyzed. I hope to use the freedom released from the relaxation of C^1/G^1 constraints to gain smoother curvature. Ideally, we can configure the control points per a user's request to generate surfaces with smoother curvature than C^1/G^1 surfaces. In the results shown in Chapters 2, 3 and 4, ε-C^1/ε-G^1 continuous surfaces sometimes show a better Gaussian curvature distribution than the corresponding C^1/G^1 surfaces. In the future, I will do further analysis to optimize curvature distribution.

The data triangulation affects the quality of resulting surfaces tremendously. Loop presented a method to optimize the data mesh for triangular subdivision surfaces in his thesis [24]. Later, this method is used in his interpolating scheme and my quintic ε-G^1 scheme, filtering out small "bad" features such as cusps, in the data mesh that might ruin the smoothness of the resulting surface.

The reason why this optimization works can be found by analyzing the curvature value around vertices. An easy method to approximate the curvature value on the data mesh

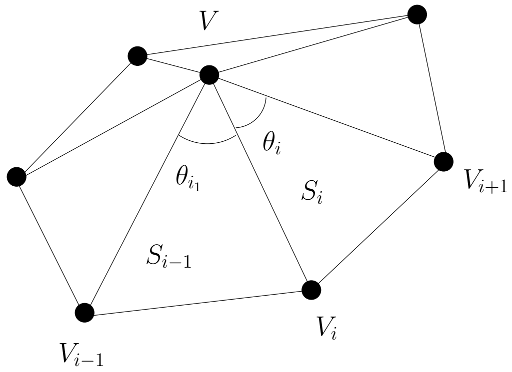


Figure 6.2: Curvature approximation of data mesh.

is the deficit method. For the example shown in Figure 6.2, if the given vertex V and its surrounding vertices V_i, $i = 0, \ldots, n-1$ are not coplanar, then using the angle deficit method, the curvature at vertex V can be approximated as

$$Cur = \frac{2\pi - \sum_{i=0}^{n-1} \theta_i}{\frac{1}{3}\sum_{i=0}^{n-1} S_i} \tag{6.1}$$

Here angle θ_i is the angle spanned by V_iV_i and V_iV_{i+1}, S_i is the area of the triangle $\triangle VV_iV_{i+1}$. In the optimization method, we translate V toward the averaged center of all the neighbours. Since we focus on the surface shape, the size of the triangles in the data mesh is assumed to be not changed much during this vertex translation process. Therefore, as the vertex V is translated toward the averaged center, the numerator of Equation 6.1 is getting smaller, and the denominator is relatively invariant. Therefore Cur is decreased after translation. This is why the approximating surface has a better surface shape than the interpolating surface in Loop's scheme [25]. In the future, I will explore other optimization methods on data meshes to improve the shape quality of ε-C^1/ε-G^1 surfaces.

With the two upper bounds of discontinuity angle introduced in Chapter 5, we can estimate the smoothness of resulting surfaces. In the future, I will try to refine the upper bounds on the normal discontinuity. Moreover, for a given discontinuity tolerance, I need to find a way to show how to guarantee this order of smoothness limited by the upper bounds. If we have an application that has a strict limit on the discontinuity, what methods should we use and what will the quality of the resulting surfaces be? In this case, the constraints of control points should be solved in reverse order for a given discontinuity value. These constraints will later be configured during the construction to guarantee the surface quality.

In my work, all the ε-C^1 and ε-G^1 surfaces here are constructed using the triangular Bézier patches. However, tensor product surfaces, particularly NURBS surfaces, are the industrial standard. It would be an interesting topic to migrate approximate continuity techniques to tensor product surfaces.

Bibliography

[1] *CGAL User and Reference Manual*, 2006.

[2] Mark Belcarz. Isophotes of approximately continuous surfaces. Technical report, 2006.

[3] R Clough and J Tocher. Finite element stiffness matrices for analysis of plates in bending. In *Proceedings of Conference on Matrix Methods in Structural Analysis*, 1965.

[4] Philip Davis. *Circulant Matrices*. Wiley, New York, 1979.

[5] Paul de Faget de Casteljau. Outillage méthodes cacul. Technical report, Institute National de la Proprieté Industrielle, Paris, 1959.

[6] T.D. Derose. *Geometric Continuity: a parameterization independent measure of continuity for computer aided geometric design*. PhD thesis, 1985.

[7] Tony DeRose and Stephen Mann. An approximately G^1 surface interpolant. In T. Lyche and L. Schumaker, editors, *Mathematical Methods in Computer Aided Geometric Design II*, pages 185–196. Academic Press, 1992.

[8] Francois Dubeau. Explicit error bounds for spline interpolation on a uniform partition. *Journal of Approximation Theory*, 1995.

[9] Gerald Farin. A construction for visual C^1 continuity of polynomial surface patches. *Computer Aided Geometric Design*, 20:272–282, 1982.

[10] Gerald Farin. Triangular Berstein-Bézier patches. *Computer Aided Geometric Design*, 3(2):83–127, 1986.

[11] Gerald Farin. *Curves and Surfaces for CAGD: A Practical Guide.* Academic Press, fifth edition, 2002.

[12] Thomas A. Foley, David A. Lane, Gregory M. Nielson, Richard Franke, and Hans Hagen. Interpolation of scattered data on closed surfaces. *Computer Aided Geometric Design*, 7:303–312, 1990.

[13] Thomas A. Foley and Karsten Opitz. Hybrid cubic Bézier triangle patches. In T Lyche and L Schumaker, editors, *Mathematical Methods for Computer Aided Geometric Design II*, pages 275–286. Academic Press, 1992.

[14] Thomas Grandine. An iterative method for computing multivariate C^1 piecewise polynomial interpolants. *Computer Aided Geometric Design*, 4:307–319, 1987.

[15] Alfred Gray. *Modern Differential Geometry of Curves and Surfaces with Mathematica.* Academic Press, second edition, 1997.

[16] Hans Hagen, Stefanie Hahmann, and Thomas Schreiber. Surface interrogation algorithms. *Computer Graphics and Applications*, 7:53–60, 1992.

[17] Stefanie Hahmann. Triangular G^1 interpolation by 4-splitting domain triangles. *Computer Aided Geometric Design*, 17(8):731–757, 2000.

[18] Stefanie Hahmann and Georges-Pierre Bonneau. Polynomial surfaces interpolating arbitrary triangulations. *IEEE Transactions on Visualization and Computer Graphics (TVCG)*, 9(1):99–109, 2003.

[19] Josef Hoschek, Dieter Lasser, and Larry L. Schumaker. *Fundamentals of Computer Aided Geometric Design.* A K Peters, 1993.

[20] J.M.Hahn. Triangular patches within a geometric continuous patch complex. Technical report, Brunel University, Uxbridge, England, 1987.

[21] Ming-Jun Lai. Geometric interpretation of smoothness conditions of triangular polynomial patches. *Computer Aided Geometric Design*, 14:191–199, 1997.

[22] Yingbin Liu and Stephen Mann. Approximate continuity for functional, triangular Bézier patches. *Presented on SIAM-Geometric Modeling 2005, submitted for review for conference proceeding*, 2005.

[23] Yingbin Liu and Stephen Mann. Approximate G^1 cubic surfaces for data approximation. *Third International Conference on Computer Graphics Theory and Applications*, pages 39–44, 2008.

[24] Charles Loop. Smooth subdivision surfaces based on triangles. Master's thesis, 1987.

[25] Charles Loop. A G^1 triangular spline surface of arbitrary topological type. *Computer Aided Geometric Design*, 11(3):303–330, 1994.

[26] Michael Lounsbery, Stephen Mann, and Tony DeRose. Parametric surface interpolation. *Computer Graphics and Applications, IEEE*, 12(5):45–52, Septemper 1992.

[27] Stephen Mann. *Surface Approximation Using Geometric Hermite Patches.* PhD thesis, University of Washington, 1992.

[28] Stephen Mann. Cubic precision Clough-Tocher interpolation. *Computer Aided Geometric Design*, 16(2):85–88, February 1999.

[29] Stephen Mann. Continuity adjustments to triangular Bézier patches that retain polynomial precision. Technical Report Research Report, Computer Science Department, University, April 2000.

[30] Stephen Mann. Adjusting control points to achieve continuity. *Computer Aided Geometric Design*, 19(7):589–602, 2002.

[31] Stephen Mann, Michael Lounsbery, Charles Loop, David Meyers, James Painter, Tony DeRose, and Kenneth Sloan. *Curve and Surface Design, Chapter 8, A Survey of Parametric Scattered Data Fitting Using Triangular Interpolants.* SIAM, 1992.

[32] Gregory M. Nielson. A transfinite, visually continuous, triangular interpolant. In G. Farin, editor, *Geometric Modeling: Algorithms and New Trends*, pages 235–246. SIAM, 1987.

[33] Jörg Peters. Smooth mesh interpolation with cubic patches. *Computer-aided Design*, 22/2:109–120, 1990.

[34] Bruce P. Piper. Visually smooth interpolation with triangular Bézier patches. In Genald Farin, editor, *Geometric Modeling: Algorithms and New Trends*, pages 221–234. SIAM, 1987.

[35] P.M.Prenter. *Splines and variational methods.* John Wiley & Sons Inc, 1975.

[36] Laurent Rineau and Mariette Yvinec. A generic software design for delaunay refinement meshing. Technical report, 2006.

[37] R F Sarraga. G^1 interpolation of generally unrestricted cubic Bézier curves. *Computer Aided Geometric Design*, 4(1-2):23–40, 1987.

[38] L.A. Shirman and C.H. Séquin. Local surface interpolation with Bézier patches: Errata and improvements. *Computer Aided Geometric Design*, 8(3):217–221, August 1991.

[39] Holger Theisel. Are isophotes and reflection lines the same? *Computer Aided Geometric Design*, 18(7):711–722, 2001.

[40] van Wijk. Bicubic patches for approximating non-rectangular control-point meshes. *Computer Aided Geometric Design*, 3(1):1–13, 1988.

[41] M A Watkins. Problems in geometric continuity. *Computer Aided Design*, 20(8):499–502, 1988.