Incremental Algorithms for the Design Of C^1-Triangular Spline Surfaces

Dianna Xu
Bryn Mawr College
Bryn Mawr, PA
dxu@brynmawr.edu

Jean Gallier
University of Pennsylvania
Philadelphia, PA
jean@cis.upenn.edu

April 21, 2006

Abstract

Spline surfaces consisting of triangular patches have a number of advantages over their rectangular counterparts, such as the ability to handle surfaces of arbitrary topology. Designing and interpolating triangular-based spline surfaces has been a research interest in the field of CAGD for some years. The problem of designing algorithms for triangular splines with local flexibility was proposed by Ramshaw [Ram87] in 1987*. To the best of our knowledge, although progress has been made, this problem is still open. Many approaches have been proposed in the following years, but none have quite achieved the elegance and flexibility of the algorithms for designing rectangular splines surfaces. We present a new de Boor-like algorithm to design open triangular C^1-splines surfaces based on general triangulations of the parameter plane. Through careful analysis of the continuity constraints based on polar forms, we discovered a way of choosing strategic control points, so that the remaining control points are computed using a simple propagation scheme. Due to its local nature, the algorithm can be easily made incremental. The algorithm operates in linear time and handles holes and sharp corners easily. Preliminary results also suggest that the algorithm can be extended to C^2-splines. As our algorithm leaves quite a bit of freedom around each vertex region, it is readily extendable to handle interpolation. However fairing methods are needed to improve the resulting surface quality.*

1 Introduction

Modeling smooth curves and surfaces continues to present many challenges and computer graphics research is still looking for the "ultimate solution". In general there are three approaches to shape representation: meshes, splines and subdivision surfaces.

The use of subdivision surfaces is a technique where polyhedra are subdivided repeatedly to approximate spline surfaces. There are rectangular (represented by Doo-Sabin [Doo78, DS78] and Catmull-Clark [CJ78] methods) as well as triangular (see Loop [Loo87]) subdivision surface algorithms. They adapt well to arbitrary topology and the implementations are straightforward. However, although it can been shown that the limit surface of a closed subdivided surface is G^1 continuous (see [BD88], [Zor96]), the continuity analysis is often complicated. But most importantly, there is no easy way to parameterize a subdivision surface for purposes such as texture-mapping.

Recent advances in the ability to process very dense point sets have led to the increased popularity of polygonal (triangular) meshes as shape representation. Meshes have no continuity requirements, and as processor power and memory start to allow dense enough meshes to closely approximate smooth surfaces, it is natural to see a preference for simple mesh-representation in many areas such as animation and gaming. In Computer Aided Geometric Design however, where manufacturability and practicality often require not just smooth-looking surfaces but surfaces with mathematical continuity and preferablly higher order continuity, smooth surface representations (mainly spline surfaces) are still very much present.

Among spline surfaces, tensor product surfaces (NURBS) remain popular because most of their properties are easy extensions from B-spline curves and well-developed algorithms that offer built-in continuity, incrementality and local control. However, the rectangular nature of these surfaces severely limits their ability to model abitrary topology. The alternative is triangular-based spline surfaces. Triangular spline surfaces have been the focus of CAGD research for some years, as they are the natural choice in a number of application areas, such as surfaces with arbitrary topology, surface fitting based on laser-scanned meshes and scatter data interpolation.

Despite the many nice properties of triangular spline surfaces, they have remained largely a research interest. Because the familiar B-spline curve framework does not carry over, they lack the simplity of tensor product surfaces. There are many different proposals to construct triangular Bézier spline surfaces or triangular B-spline surfaces, but while theoretically sound, they all failed to provide a simple and intuitive way of designing and manipulating the surface shape. This work was previously presented in Xu's dissertation thesis [Xu02]. The most relevant previous attempts are Loop [Loo94], which cannot handle interpolation and Hahmann et al [HBT00, HBT01, HB03], which extends Loop's method with a split-domain approach to achieve interpolation, and then further incorporates hierachical methods to achieve adaptive fitting. In short, an algorithm in the spirit of the elegant de Boor algorithm [Far92, Gal00a] for tensor product surfaces does not exist for triangular surfaces.

This work introduces a new algorithm to design smooth open triangular spline surfaces with quintic Bézier patches. We believe that our approach is simple, geometrically intuitive and computationally efficient. Our algorithm is designed to satisfy the following goals:

- Design of complex 3D shapes based on a parametrization of the plane.
- Shapes may have arbitrarily many holes and sharp corners.
- Shape modification by an incremental algorithm.
- Local control so that modifications only affect local neighborhood.
- Reasonable smoothness (C^1).

Preliminary results show that the same scheme may be extended to C^2, with higher degree patches. We have also extended this approach to handle closed surfaces with G^1 continuity, which will be presented in a separate paper. Our approach can also be extended to handle interpolation almost immediately, due to the freedom around each vertex region. However there is a degrading effect in the surface quality without further fairing methods. Note that the first and foremost aim of this work is to come up with an algorithm for designing triangular spline surfaces. Because our framework allows enough freedom for a theoretically easy extension to interpolation, it is always in the back of our mind. In our research, care has been taken to always take extension to interpolation into account. However, so far we have not conducted any substantial investigation to make claims for interpolation results.

2 Related Works

C^1 surface fitting methods can be divided into global and local categories. Global methods always involve solving systems of equations which generally become large and unmanageable as the data set gets bigger. Thus the majority of the methods are local, although many of them do resort to global measures for shape fairing.

An extensive survey of local methods for scattered data interpolation has been conducted by Franke [Fra82]. It is an excellent starting point for literature on this topic. According to Franke, the methods can be divided into inverse distance weighted methods due to Shepard [She68], rectangle based methods, triangle based methods and finite element based methods. We will only look at triangle based methods and finite element based methods because they are all based on a domain triangulation, and are, therefore, most relevant.

Early triangle based methods were all blending schemes with weight functions, such as Gold et al [GCR77] and Franke and Nielson [FN83]. There has been no recent literature in this category to our best knowledge, most likely due to the appearance and popularity of geometric continuity and G^1 surfaces.

Akima's method [Aki78] belongs to the finite element category, but it is in fact the closest in spirit to our approach. It fits a C^1 surface using quintic finite elements, and requires estimation of derivatives. In general estimating derivatives is a common theme in most finite element approaches, which is very similar to our tangent plane placement problem (see Section 7.2.3). There are also finite element approaches using split-domain schemes, such as Lawson [Law77].

Since the work we are presenting here works only on planar triangulations, it does not produce closed surfaces. However, since extensions to G^1 closed surfaces is the eventual goal, a survey of G^1 works will also be briefly presented. The majority of methods for fitting geometric-continuous surfaces differ in the way they simultaneously satisfy smoothness conditions for many patches around a common corner. This is also known as the "vertex consistency" or "twist compatibility" problem. There are subdivision methods, such as Loop [Loo87], Cavaretta et al [CDM91], Hoppe et al [HDD$^+$94] and Peters [Pet95], and more recently, [BLZ00] and [Kob01] and split-patch methods which are Clough-Tocher like split-domain schemes. Earlier split-patch approaches including Farin [Far82, Far83], Jensen [Jen87], Piper [Pip87] and Shirman and Sequin [SS88]. A more recent approach due to Hahmann et al [HBT00, HBT01] applies a new 4-split method. Split-domain schemes generally produce lower degree surfaces. Hahmann et al have also extended the approach to handle interpolation [HB03] and adaptive fitting [YHB05]. Convex combination schemes such as Hagen [Hag86] and Hagen and Pottmann [HH89] which construct higher degree triangular patches with a single patch per parameter triangle.

In addition to the above approaches, there are C^2-consistent boundary curve schemes such as Loop [Loo94] and Peters [Pet91] and variational methods using the theory of manifolds by Sarraga [Sar00] as well as B-patches by Seidel [GS93, PS95]. The surveys by Du and Schmitt [DS90] and Mann et al [MLM$^+$92] provide more detailed description of the above methods. It is pointed out in [MLM$^+$92] that most of these methods suffer from shape defects.

3 Polar Forms and Polynomial Surfaces

Our main results make heavy use of affine geometry and its properties. In particular, the concept of polar forms. The principle of (affine) "blossoming" was first introduced by de Casteljau [dC86] and Ramshaw [Ram88]. Blossoming can be viewed as a generalization of the well-known de Casteljau algorithm. In many cases, it is more convenient to study curves and surfaces from their polar forms or "blossoms"[Ram89], because it provides a way of labeling the Bézier or the de Boor control points with symmetric, multivariate labels which provide excellent geometric intuitions.

Since their introduction, polar forms have gained considerable popularity because of the geometric insights they provided to Bézier and B-splines related algorithms. The theory of blossoming has been applied to B-spline knot insertion [Sei88, Sei89], triangular B-spline surfaces [MS92] and geometrically continuous spline curves [Sei93]. For a more in-depth reading on the theory of blossoming, the readers are referred to Ramshaw [Ram87] and Gallier [Gal00a].

Since polar forms are in fact symmetric multiaffine maps, we first discuss the representation of polynomial maps in terms of multiaffine maps.

Definition 3.1 A function $f: \underbrace{\mathbb{A}^d \times \cdots \times \mathbb{A}^d}_{m} \to \mathbb{A}^n$ is a *multiaffine map* (or an *m-affine map*), iff it is affine with respect to each of its argument, that is, for every i, with $1 \leq i \leq m$, considering $a_1, \ldots, a_{i-1}, a_{i+1}, \ldots, a_n$ fixed, for all $a_i \in \mathbb{A}^d$, the map

$$a_i \to f(a_0, \ldots, a_i, \ldots, a_m)$$

is affine.

Definition 3.2 A *symmetric* map is a function that is invariant under permutation of its arguments, that is, the result of the function does not depend on any particular order of its arguments. In other words, given a map $f: \underbrace{\mathbb{A}^d \times \cdots \times \mathbb{A}^d}_{m} \to \mathbb{A}^n$,

$$f(a_{\pi(1)}, \ldots, a_{\pi(m)}) = f(a_1, \ldots, a_m),$$

for all $a_1, \ldots, a_m$, and all permutations π.

We can use multiaffine maps to define generalized polynomial maps between two affine spaces of arbitrary dimensions. In the special case where the multiaffine map maps from $\mathbb{A}^n$ to $\mathbb{A}$, it is equivalent to the notion of a polynomial function induced by a polynomial in n variables. In fact, every polynomial curve of degree m has a unique symmetric m-affine map associated with it. It is called the *m-polar* form of the curve, or its "blossom".

For example, consider the parabola $F: \mathbb{A} \to \mathbb{A}^2$, given by

$$\begin{aligned} x(t) &= 4t, \\ y(t) &= t^2 - 3t + 2. \end{aligned}$$

The polynomial map $F: \mathbb{A} \to \mathbb{A}^2$ comes from a unique symmetric biaffine map $f: \mathbb{A}^2 \to \mathbb{A}^2$, where

$$\begin{aligned} f_1(t_1, t_2) &= 2(t_1 + t_2), \\ f_2(t_1, t_2) &= t_1 t_2 - \frac{3}{2}(t_1 + t_2) + 2, \quad \text{such that} \\ F(t) &= f(t, t), \quad \text{for all } t \in \mathbb{A}. \end{aligned}$$

3.1 Polynomial Surfaces and Polar Forms

3.1.1 Polarizing polynomial surfaces

A polynomial curve is easily generalized to a polynomial surface by having the polynomials take two parameters. Thus, we have a polynomial map $F(u, v)\colon \mathbb{A}^2 \to \mathbb{A}^3$.

For example, the following polynomials define a bipolynomial surface of degree $(3, 3)$, known as *Enneper's Surface*:

$$\begin{aligned}
F_1(u, v) &= u - \frac{u^3}{3} + uv^2 \\
F_2(u, v) &= v - \frac{v^3}{3} + u^2v \\
F_3(u, v) &= u^2 - v^2.
\end{aligned}$$

Recall that symmetric multi-affine maps can be used to define the notion of a polynomial function between two affine spaces of arbitrary dimension. We have seen before that polynomial curves have associated polar forms. Similarly, we also have polar forms associated with polynomial surfaces.

To polarize a polynomial means to find the unique symmetric affine map associated with the given polynomial. Now, since we have polynomial surfaces, we have two parameter variables (u, v) involved in the polynomials, and there are two natural ways to polarize a given polynomial surface $F\colon \mathbb{A}^2 \to \mathbb{A}^3$.

The first approach is to polarize separately in the two variables, which yields bipolynomial surfaces, also commonly called tensor product surfaces. If p and q are the two degrees of the bipolynomial surface, then we get a $(p + q)$-multiaffine map which is symmetric in its first p arguments, and in its last q arguments, but not symmetric in all its arguments. This approach basically divides the parameter plane into rectangles.

The second approach treats the two variables (u, v) as a whole, namely, as coordinates of a point (u, v) in the affine plane. This approach produces total degree surfaces. If m is the degree of the total degree surface, then we get a m-affine map which is symmetric in all of its arguments. In this sense, this method is a natural generalization of the Bézier curves, and in fact, total degree surfaces (or Bézier triangular surfaces) were the first surfaces to be considered by de Casteljau himself. This approach triangulates the parameter plane.

A polynomial surface F of total degree m is completely specified by a (triangular) control net, or Bézier net consisting of control points.

3.1.2 Control points for triangular surfaces

Given an affine frame Δrst in the plane (where $r, s, t \in \mathcal{P}$ are affinely independent points), it turns out that any symmetric multiaffine map $f\colon \mathcal{P}^m \to \mathcal{E}$ is uniquely determined by a family $\mathcal{N} = (b_{i,\,j,\,k})_{(i,j,k)\in\Delta_m}$ of $\dfrac{(m+1)(m+2)}{2}$ points (where $\mathcal{E}$ is any affine space, say $\mathbb{R}^n$), known other wise as a *(triangular) control net, or Bézier net.*

For example, with respect to the frame $\Delta rst = ((1,0,0),(0,1,0),(0,0,1))$, we obtain 10 control points for the Enneper surface, as illustrated in Table 1.

			$f(r,r,r)$ $\left(\frac{2}{3},0,1\right)$			
		$f(r,r,t)$ $\left(\frac{2}{3},0,\frac{1}{3}\right)$		$f(r,r,s)$ $\left(\frac{2}{3},\frac{2}{3},\frac{1}{3}\right)$		
	$f(r,t,t)$ $\left(\frac{1}{3},0,0\right)$		$f(r,s,t)$ $\left(\frac{1}{3},\frac{1}{3},0\right)$		$f(r,s,s)$ $\left(\frac{2}{3},\frac{2}{3},-\frac{1}{3}\right)$	
$f(t,t,t)$ $(0,0,0)$		$f(s,t,t)$ $\left(0,\frac{1}{3},0\right)$		$f(s,s,t)$ $\left(0,\frac{2}{3},-\frac{1}{3}\right)$		$f(s,s,s)$ $\left(0,\frac{2}{3},-1\right)$

Table 1: Control points for Enneper's surface

4 Continuity Conditions of Triangular Spline Surfaces

4.1 Continuity conditions on polar forms

The most important issue in joining polynomial surfaces is continuity along the boundaries. The question of exactly what we mean by "smooth" is of central importance. Today, the well-accepted standard is to look at *parametric continuity.* The parametric functions that define the surfaces are considered smooth if and only if their respective derivatives are well-defined up to some order.

Definition 4.1 Given two surface patches F and G, for any point $a \in \mathbb{A}^2$, F and G are said to join with C^k-continuity at a if and only if

$$F^{(i)}(a) = G^{(i)}(a),$$

for all i, $0 \leq i \leq k$. That is, their derivatives at a agree up to kth order.

We shall look at conditions forced on polar forms if two surfaces are to join with C^k-continuity. We focus our interest on spline surfaces based on a triangulation of the plane.

Definition 4.2 Let A and B be two adjacent convex polygons in the plane, and let (r, s) be the line segment along which they are adjacent (where $r, s \in \mathbb{A}^2$ are distinct vertices of A and B). Given two polynomial surfaces F_A and F_B of degree m, *F_A and F_B join with C^k continuity along the line segment (r, s)* iff F_A and F_B agree to kth order for all $a \in (r, s)$. That is, for any point a along the segment (r, s), all derivatives of F_A and F_B up to the kth order must agree at a.

Lemma 4.3 *For any $a \in (r, s)$, F_A and F_B agree to kth order at a iff their polar forms $f_A \colon (\mathbb{A}^2)^m \to \mathbb{A}^d$ and $f_B \colon (\mathbb{A}^2)^m \to \mathbb{A}^d$ agree on all multisets of points that contain at least $m - k$ copies of a, that is, iff*

$$f_A(u_1, \ldots, u_k, \underbrace{a, \ldots, a}_{m-k}) = f_B(u_1, \ldots, u_k, \underbrace{a, \ldots, a}_{m-k}),$$

for all $u_1, \ldots, u_k \in \mathbb{A}^2$.

Using the above lemma, we can prove the following crucial result:

Lemma 4.4 *Let A and B be two adjacent convex polygons in the plane, and let (r, s) be the line segment along which they are adjacent (where $r, s \in \mathbb{A}^2$ are distinct vertices of A and B). Given two polynomial surface F_A and F_B of degree m, F_A and F_B join with C^k continuity along (r, s) iff their polar forms $f_A \colon (\mathbb{A}^2)^m \to \mathbb{A}^d$ and $f_B \colon (\mathbb{A}^2)^m \to \mathbb{A}^d$ agree on all multisets of points that contain at least $m - k$ points on the line (r, s), that is, iff*

$$f_A(u_1, \ldots, u_k, a_{k+1}, \ldots, a_m) = f_B(u_1, \ldots, u_k, a_{k+1}, \ldots, a_m),$$

for all $u_1, \ldots, u_k \in \mathbb{A}^2$, and all $a_{k+1}, \ldots, a_m \in (r, s)$.

Proof. As Lemma 4.3 states, for every $a \in (r, s)$, F_A and F_B agree to kth order at a iff

$$f_A(u_1, \ldots, u_k, \underbrace{a, \ldots, a}_{m-k}) = f_B(u_1, \ldots, u_k, \underbrace{a, \ldots, a}_{m-k}),$$

for all $u_1, \ldots, u_k \in \mathbb{A}^2$. However, if we consider

$$a \mapsto f_A(u_1, \ldots, u_k, \underbrace{a, \ldots, a}_{m-k})$$

and

$$a \mapsto f_B(u_1, \ldots, u_k, \underbrace{a, \ldots, a}_{m-k})$$

as affine polynomial functions $F_A(u_1, \ldots, u_k)$ and $F_B(u_1, \ldots, u_k)$, if these functions agree on all points in (r, s), because of the uniqueness of the polar form associated with a polynomial function, the corresponding polar forms $f_A(u_1, \ldots, u_k)$ and $f_B(u_1, \ldots, u_k)$ agree for all points $a_{k+1}, \ldots, a_m \in (r, s)$. Since this holds for all $u_1, \ldots, u_k \in \mathcal{P}$, we have shown that

$$f_A(u_1, \ldots, u_k, a_{k+1}, \ldots, a_m) = f_B(u_1, \ldots, u_k, a_{k+1}, \ldots, a_m),$$

for all $u_1, \ldots, u_k \in \mathbb{A}^2$, and all $a_{k+1}, \ldots, a_m \in (r, s)$, as desired. $\square$

4.2 Necessary and sufficient polar form conditions for C^1 continuity

As a consequence of Lemma 4.4, we obtain the necessary and sufficient conditions on triangular control nets for two surface patches F_A and F_B of degree m to join with C^n continuity along (r, s).

Let $A = \Delta prs$ and $B = \Delta qrs$ be two frames in the plane, sharing the edge (r, s).

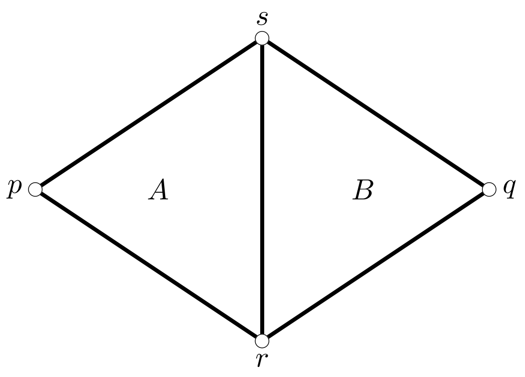

Figure 1: Two adjacent reference triangles

Then, Lemma 4.4 tells us that F_A and F_B join with C^k continuity along (r, s) iff

$$f_A(p^g q^h r^i s^j) = f_B(p^g q^h r^i s^j),$$

for all g, h, i, j such that $g + h + i + j = m$, and $i + j \geq m - k$ $(0 \leq k \leq m)$.

For C^0 continuity, we have $k = 0$. Thus, $i+j \geq m-0 \Rightarrow i+j \geq m$, since $g+h+i+j = m$, we have $g = h = 0$. Therefore, we just have

$$f_A(r^i s^{m-i}) = f_B(r^i s^{m-i}),$$

with $0 \leq i \leq m$, which means that the control points of the boundary curves along (r, s) must agree.

For C^1 continuity, we have $k = 1$. Thus, $i + j \geq m - 1$. Since $g + h + i + j = m$, we have three cases. Either $i + j = m$, then $g = h = 0$, or $i + j = m - 1$, in which case either $g = 0$ and $h = 1$, or $g = 1$ and $h = 0$. Thus, we have the conditions

$$\begin{aligned} f_A(r^i s^{m-i}) &= f_B(r^i s^{m-i}),\ 0 \leq i \leq m, && (1)\\ f_A(pr^i s^{m-i-1}) &= f_B(pr^i s^{m-i-1}),\ 0 \leq i \leq m-1, && (2)\\ f_A(qr^i s^{m-i-1}) &= f_B(qr^i s^{m-i-1}),\ 0 \leq i \leq m-1. && (3) \end{aligned}$$

This is a total of $3m + 1$ conditions. However, we will now show that in general, only $2m + 1$ of these conditions are independent. Let $q = \lambda p + \mu r + \nu s$, where $\lambda + \mu + \nu = 1$. Since we are assuming that A and B are distinct triangles, $\lambda \neq 0$, and

$$p = \frac{1}{\lambda} q - \frac{\mu}{\lambda} r - \frac{\nu}{\lambda} s,$$

and Condition 2

$$f_A(pr^i s^{m-i-1}) = f_B(pr^i s^{m-i-1})$$

becomes

$$f_A(pr^i s^{m-i-1}) = f_B\left(\left(\frac{1}{\lambda} q - \frac{\mu}{\lambda} r - \frac{\nu}{\lambda} s\right) r^i s^{m-i-1}\right),$$

and since f_B is multiaffine, this yields

$$f_A(pr^i s^{m-i-1}) = \frac{1}{\lambda} f_B(qr^i s^{m-i-1}) - \frac{\mu}{\lambda} f_B(r^{i+1} s^{m-i-1}) - \frac{\nu}{\lambda} f_B(r^i s^{m-i}),$$

or equivalently

$$\lambda f_A(pr^i s^{m-i-1}) + \mu f_B(r^{i+1} s^{m-i-1}) + \nu f_B(r^i s^{m-i}) = f_B(qr^i s^{m-i-1}),$$

for $0 \leq i \leq m - 1$.

Now using Condition 1

$$f_A(r^i s^{m-i}) = f_B(r^i s^{m-i}),$$

we get

$$(*) \qquad \lambda f_A(pr^i s^{m-i-1}) + \mu f_A(r^{i+1} s^{m-i-1}) + \nu f_A(r^i s^{m-i}) = f_B(qr^i s^{m-i-1}),$$

for $0 \leq i \leq m-1$. Similarly, since $q = \lambda p + \mu r + \nu s$, from Condition 3

$$f_A(qr^i s^{m-i-1}) = f_B(qr^i s^{m-i-1}),$$

we get

$$f_A((\lambda p + \mu r + \nu s) r^i s^{m-i-1}) = f_B(qr^i s^{m-i-1}),$$

which yields

$$\lambda f_A(pr^i s^{m-i-1}) + \mu f_A(r^{i+1} s^{m-i-1}) + \nu f_A(r^i s^{m-i}) = f_B(qr^i s^{m-i-1}),$$

for $0 \leq i \leq m-1$. These are the conditions $(*)$ that we found earlier, and thus, the conditions for C^1-continuity are indeed the $2m+1$ conditions

$$\begin{aligned} f_A(r^i s^{m-i}) &= f_B(r^i s^{m-i}),\ 0 \leq i \leq m, && (4)\\ f_B(qr^i s^{m-i-1}) &= \lambda f_A(pr^i s^{m-i-1}) + \mu f_A(r^{i+1} s^{m-i-1}) + \nu f_A(r^i s^{m-i}) && (5)\\ 0 \leq i \leq m-1&, \textit{ with } q = \lambda p + \mu r + \nu s. \end{aligned}$$

These conditions show that the control points $f_A(pr^i s^{m-i-1})$, $f_A(r^{i+1} s^{m-i-1})$, $f_A(r^i s^{m-i})$ and $f_B(qr^i s^{m-i-1})$ satisfy the same affine relation that p, r, s, q satisfy, i.e., the diamond formed by these control points is an affine image of the diamond (p, r, s, q) under some unique affine map h.

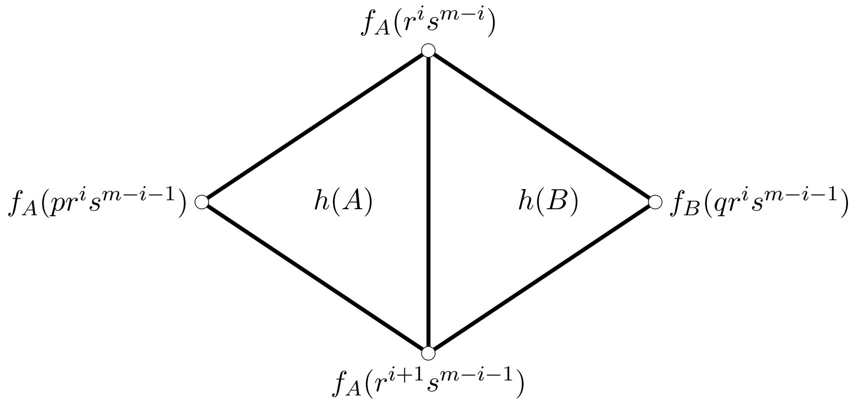

Figure 2: C^1-continuity conditions

In the special case where $q = r + s - p$, or equivalently $p + q = r + s$, which means that the line segments (p, q) and (r, s) have the same midpoint, the above Conditions 5 become

$$\frac{1}{2}(f_A(pr^i s^{m-i-1}) + f_B(qr^i s^{m-i-1})) = \frac{1}{2}(f_A(r^{i+1}s^{m-i-1}) + f_A(r^i s^{m-i})), \quad 0 \le i \le m-1. \tag{6}$$

Now we will look at C^1 continuity constraints in more detail. We start from the special case where adjacent triangles form a parallelogram. This is true if the all triangles in the parameter plane are equilateral.

4.3 Around a vertex

Assume that we have a (finite) triangulation of the parameter plane $\mathbb{A}^2$ consisting of equilateral triangles. We know that to achieve C^1 continuity, the line segment between the apexes of any two triangular patches sharing an edge must have the same midpoint as the edge itself, as stated by equation 6. Let us take a closer look at what this means when three or more adjacent triangular patches have a common vertex. First of all, we introduce a definition that will simplify our notations somewhat.

Definition 4.5 Given $n \ge 2$ triangular patches of degree m that share a common vertex v_1, let $T_1, \ldots, T_n$ be the n triangles in the planar domain and $f_1, \ldots, f_n$ be the n polar forms associated with the patches. Let $v_1, \ldots, v_{n+2}$ denote the vertices of the domain triangles, so that $T_i = (v_1, v_{i+1}, v_{i+2})$, with $1 \le i \le n$. Let $z_1 = f_i({v_1}^m)$, then the control points $z_{i+1} = f_i({v_1}^{m-1}, v_{i+1})$, *with* $1 \le i \le n+1$, are called the *star of control points* around z_1, or simply, the *star* of z_1 (See Figure 3).

We call the set $\{z_2, \ldots, z_{n+2}\}$ a *complete star* if z_1 is completely surrounded by patches. A complete star has n control points. An incomplete star has $n + 1$ control points.

Notice that vertex v_{n+2} coincides with v_2 if the star is complete.

We should also point out the v_i are in the planar domain and the z_i are control points in space. However, the corner point and its star, i.e. the z_i, are indeed coplanar and their convex hull is an affine image of the convex hull given by the v_i. One can think of Figure 3 as having the z_i superimposed onto the domain triangulation given by the v_i. Also, recall that the notation $f_i({v_j}^m)$ indicates that there are m copies of v_j in f_i's arguments, that is:

$$f_i({v_j}^m) = f_i(\underbrace{v_j, \ldots, v_j}_{m}).$$

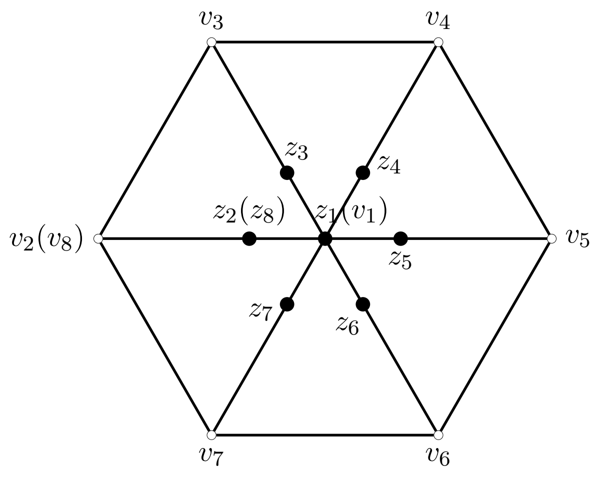

Figure 3: $z_2, \ldots, z_7$ is the complete star of z_1

The star completely determines the continuity conditions around a vertex, and therefore we will look at it closely. The necessary and sufficient polar form Conditions 6 given in 4.2 for C^1 continuity apply, as we have an equilateral template triangulation.

To simplify the notation for the equations given by Conditions 6, we would like to drop $\frac{1}{2}$. It turns out that it is all right to do so, because there is a contruction that will embed an affine space in a vector space. Thus, affine points can be turned into vectors, and then back to points. For more information on such a "homogenizing" construction, please refer to Chapter 4 of Gallier [Gal00b], or Berger [Ber90]. We begin with three patches sharing a common vertex.

4.3.1 Case 1: Three triangular patches.

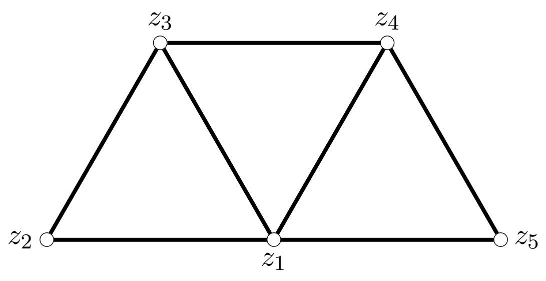

Figure 4: Three adjacent triangular patches

The two C^1-equations are

$$z_1 + z_3 = z_2 + z_4,$$
$$z_3 + z_5 = z_1 + z_4.$$

Subtracting the second equation from the first, we get $z_1 - z_5 = z_2 - z_1$, or

$$z_2 + z_5 = 2z_1.$$

Thus, z_1 is the middle of the line segment (z_2, z_5). There are three degrees of freedom, for example, z_1, z_3, z_4.

To avoid redundancy, we will skip over the next two cases, e.g. four and five triangular patches, and go straight to six patches.

4.3.2 Case 2: Six triangular patches.

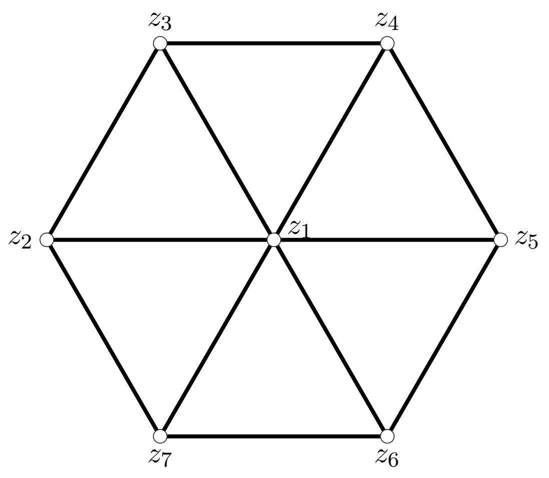

Figure 5: Six adjacent triangular patches

There are two more equations, and the six C^1-equations are

$$z_1 + z_3 = z_2 + z_4,$$
$$z_3 + z_5 = z_1 + z_4,$$
$$z_1 + z_5 = z_4 + z_6,$$
$$z_5 + z_7 = z_1 + z_6,$$
$$z_2 + z_1 = z_3 + z_7,$$
$$z_2 + z_6 = z_1 + z_7.$$

Remarkably, the last two equations follow from the first four. Indeed, as in the previous case, we get

$$\begin{aligned} z_2 + z_5 &= 2z_1, \\ z_3 + z_6 &= 2z_1, \\ z_4 + z_7 &= 2z_1, \end{aligned}$$

and by adding the two equations

$$\begin{aligned} z_1 + z_3 &= z_2 + z_4, \\ z_4 + z_7 &= 2z_1, \end{aligned}$$

we get

$$z_1 + z_3 + z_4 + z_7 = 2z_1 + z_2 + z_4$$

which reduces to

$$z_3 + z_7 = z_1 + z_2,$$

that is,

$$z_2 + z_1 = z_3 + z_7.$$

Similarly, by adding the two equations

$$\begin{aligned} z_2 + z_1 &= z_3 + z_7, \\ z_3 + z_6 &= 2z_1, \end{aligned}$$

we get

$$z_2 + z_1 + z_3 + z_6 = 2z_1 + z_3 + z_7$$

which reduces to

$$z_2 + z_6 = z_1 + z_7.$$

Thus, the rank of the system is four, and it is easily seen that the equations

$$\begin{aligned} z_2 + z_5 &= 2z_1, \\ z_3 + z_6 &= 2z_1, \\ z_4 + z_7 &= 2z_1, \\ z_2 + z_4 + z_6 &= 3z_1, \end{aligned}$$

are linearly independent. There are three degrees of freedom, for example, z_2, z_4, z_6 (or z_3, z_5, z_7).

In all cases, there are three degrees of freedom among the vertex and its star, that is, any three control points that are not collinear completely determine the continuity condition around that vertex.

4.3.3 Along an edge

When two degree m patches meet, the C^1 continuity constraints along the common edge are completely determined by the parallelograms formed by the control points across the common edge. Figure 6 shows two patches of degree four, which contains three such parallelograms. In general, there are $m-1$ such parallelograms for any two degree m patches that share an edge.

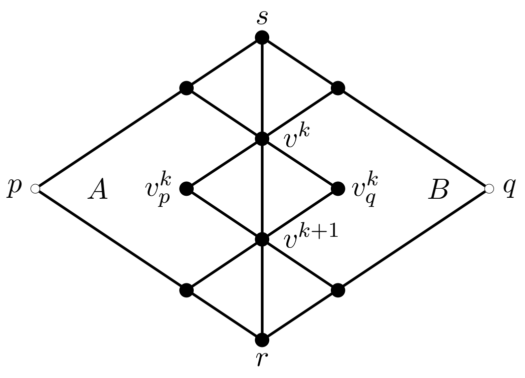

Figure 6: Two cubic patches that share an edge

We must have

$$v_A^k = v_B^k$$

along the common edge (r, s) and every diamond $(v_p^k, v^k, v_q^k, v^{k+1})$ must be the image of (p, s, q, r) under a bijective affine map, as shown in Figure 2. We can pick any three non-collinear points among (p, q, r, s). Assuming (p, r, q) are not collinear and $s = \lambda p + \mu r + \nu q$, we get

$$v^k = \lambda v_p^k + \mu v^{k+1} + \nu v_q^k.$$

as shown in Conditions 5 in Chapter 4.

Similarly, in the special case where (p, s, q, r) is a parallelogram, Conditions 5 become Conditions 6, that is, the line segments (p, q) and (r, s) have the same midpoint, and we get

$$\frac{1}{2}(v^k + v^{k+1}) = \frac{1}{2}(v_p^k + v_q^k),$$

written simply as:

$$v^k + v^{k+1} = v_p^k + v_q^k.$$

Thus, any three non-collinear control points among the four control points that make up a parallelogram will determine the fourth.

4.3.4 Arbitrary triangulation

So far, we have seen that with a equilateral triangulation, there are three degrees of freedom around a vertex control point and its star, and also in any parallelogram across any shared edge. This is a nice result, but in obtaining the result, we relied on diagonals of the parallelograms having the same midpoint. This is no longer true when we have an arbitrary triangulation.

Much to our delight, the case of an arbitrary triangulation is not much more difficult. It turns out that we still have three degrees of freedom among the vertex and its star, regardless of how many patches come together at that vertex.

Recall Conditions 5 given in 4.2 for the general case:

$$f_B(qr^is^{m-i-1}) = \lambda f_A(pr^is^{m-i-1}) + \mu f_A(r^{i+1}s^{m-i-1}) + \nu f_A(r^is^{m-i}),$$

where $\lambda + \mu + \nu = 1$.

These conditions show that the control points $f_A(pr^is^{m-i-1})$, $f_A(r^{i+1}s^{m-i-1})$, $f_A(r^is^{m-i})$, and $f_B(qr^is^{m-i-1})$, satisfy the same affine relation that p, r, s, q satisfy, i.e., the diamond formed by these control points is an affine image of the diamond (p, r, s, q) under some affine map h.

Lemma 4.6 *Given $n \geq 2$ triangular patches that share a common vertex, let $T_1, \ldots, T_n$ be the $n \geq 2$ triangles in the planar domain associated with the patches, with $T_i = (v_{i+1}, v_1, v_{i+2})$, so that T_i and T_{i+1} share the vertex v_1 and are adjacent along the edge (v_1, v_i) $(3 \leq i \leq n+1)$. Let z_1 be the control point associated to v_1, and $z_2, \ldots, z_{n+2}$ the star of control points of z_1 (note that $v_{n+2} = v_2$ and $z_{n+2} = z_2$ if the star is complete). Also let $F_1, \ldots, F_n$ be the polynomial maps associated with the patches. If the patches $F_i(T_i)$ and $F_{i+1}(T_{i+1})$ meet with C^1 continuity, for all $1 \leq i \leq n$, then there is a unique affine map $h\colon \mathbb{A}^2 \to \mathbb{A}^3$ such that*

$$h(v_i) = z_i,$$

with $1 \leq i \leq n+2$.

As a consequence, if (v_i, v_j, v_k) are any three affinely independent points, for any v_l with $l \neq i, j, k$, if $v_l = \lambda v_i + \mu v_j + \nu v_k$, we also have the equation

$$z_l = \lambda z_i + \mu z_j + \nu z_k,$$

where $\lambda + \mu + \nu = 1$.

Proof. We proceed with induction on n. Base case: $n = 2$. There are two triangles, $T_1 = (v_2, v_1, v_3)$, and $T_2 = (v_3, v_1, v_4)$ that meet with C^1 continuity. According to Conditions

5 obtained in Chapter 4, the points z_4, z_1, z_2 and z_3 are an image of v_4, v_1, v_2 and v_3 under a unique affine map.

Assume that Lemma 4.6 holds for n. Now, we prove it for $n+1$. According to the induction hypothesis, patches $F_i(T_i)$ and $F_{i+1}(T_{i+1})$ meet with C^1 continuity for all $1 \leq i \leq n$, thus there is an unique affine map h such that

$$h(v_i) = z_i,$$

with $1 \leq i \leq n+2$. Now, we have a new patch $F_{n+1}(T_{n+1})$ that meets with patch $F_n(T_n)$ with C^1 along (v_1, v_{n+2}). According to Conditions 5, there is also an unique affine map g such that $g(v_i) = z_i$, with $i \in [1, n+1, n+2, n+3]$. Thus we have the affine maps h and g overlap on the triangle $T_n = (v_1, v_{n+1}, v_{n+2})$. Because the triangle T_n is a proper triangle with three affinely independent vertices which form an affine frame, this forces h and g to be the same affine map. $\square$

Since any three affinely independent points form an affine basis, any other point is completely determined by a unique affine combination of these three points. When any three non-collinear control points among z_1 and its star are determined, to compute any other control point in the star, we can immediately find the corresponding vertices of the four control points involved, compute the barycentric coefficients for the unique affine combination between the vertices, and use the same coefficients to compute the fourth control point. Thus, regardless of how many patches come together, any three non-collinear control points among z_1 and its star determine the continuity conditions around a vertex completely. The same also holds for similar quadrilaterals along the edges.

5 An Algorithm for Designing C^1 Triangular Spline Surfaces

5.1 Algorithm to choose prescribed (free) control points based on equilateral triangulations

The challenge is to come up with a way to systematically prescribe a certain set of control points, so that the rest of the control points can be computed efficiently, and the resulting surface has guaranteed C^1 continuity. We restrict ourselves for now to the special case where all triangles in the template triangulation are equilateral. This guarantees that the two lines formed by the opposite apexes of any two adjacent triangles (line segments (p, q) and (r, s)) have the same midpoint. Notice that restricting all triangles to be isosceles, with the third, non-equilateral edge as the shared common edge between two triangles achieves the same effect, but the condition makes the template triangulation a lot less regular, and harder to draw.

5.1.1 Degree 5 patches

It is generally impossible to construct a triangular spline surface of degree m with C^k continuity, if $2m \leq 3k+1$. The proof of this is quite involved, and a detailed account can be found on pages $317-320$ in Gallier's *Curves and Surfaces in Geometric Modeling* [Gal00b].

Thus, we must have $2m \geq 3k+2$. For C^1 continuity, we have $k=1$, and therefore m must be at least three. Because there are three degrees of freedom in all cases where adjacent triangles share some common vertex, the systems of equations to solve for control points are over-determined for $m \leq 4$. It turns out that in order to have local flexibility and a reasonably symmetric method, $m=5$ is the smallest possible degree to allow enough freedom. Therefore we turn our attention to quintic patches.

Since we have degree five, each patch has 21 control points. We divide these control points into corner points, edge points and inner points. Recall that the control points

$$b_{i,j,k} = f(\underbrace{r,\ldots,r}_{i},\underbrace{s,\ldots,s}_{j},\underbrace{t,\ldots,t}_{t})$$

where $i+j+k=m$ can be viewed as an image of the triangular grid Δ_m defined by

$$\Delta_m = \{(i,j,k) \in \mathbf{N}^3 \quad | i+j+k=m\}.$$

The 21 control points of a quintic patch forms a triangular grid:

					500					
				401		410				
			302		311		320			
		203		211		221		230		
	104		113		122		131		140	
005		014		023		032		041		050

Table 2: Control points grid for a quintic triangular patch

Definition 5.1 Given a triangular control net of degree m,

1. *Corner points* are control points whose corresponding polar values have either $i=m$, $j=m$ or $k=m$, which means any two of i,j,k are 0.

2. *Edge points* are control points whose corresponding polar values have either $i+j=m$, $j+k=m$ or $i+k=m$, which means any one of i,j,k is 0.

3. *Inner points* are control points whose corresponding polar values have $i+j+k=m$ with $i \neq 0, j \neq 0, k \neq 0$.

When $m=5$, the corner points are $b_{0,0,5}$, $b_{5,0,5}$ and $b_{5,0,0}$, the edge points are $b_{4,0,1}$, $b_{3,0,2}$, $b_{2,0,3}$, $b_{1,0,4}$, $b_{0,1,4}$, $b_{0,2,3}$, $b_{0,3,2}$, $b_{0,4,1}$, $b_{1,4,0}$, $b_{2,3,0}$, $b_{3,2,0}$ and $b_{4,1,0}$ and the inner points are $b_{3,1,1}$, $b_{1,3,1}$, $b_{1,1,3}$, $b_{2,1,2}$, $b_{2,2,1}$, $b_{1,2,2}$. It is clear that the corner points correspond to points on the three corners of the triangular surface patch, the edge points correspond to points on the three edges of the triangular surface patch, and the inner points correspond to points not on the boundary of the triangular surface patch.

5.1.2 Corner point region

First of all, we need to specify a uniform way to prescribe the corner points. As we have seen earlier in Chapter 4, the continuity conditions around a corner point depends on the number of patches (or the number of edges) coming together at that particular point, although there are three degrees of freedom in all cases.

In fact, a corner point that is adjacent to exactly two edges is common to only one face, and therefore does not contribute to or constrain the continuity conditions of the patch. Such points and its two edges are commonly called an "ear" in many literatures on triangulation, and we will refer to such corner points and edge points on the two edges associated as *ear points*.

Because we are dealing with equilateral triangles, there can be at most six patches coming together at a corner point. Thus, we need only to specify prescription methods for corner points adjacent to two to six patches. They are shown in Figure 7, where a square represents a corner point, a circle represents an edge point and a green square or circle indicates that particular control point is prescribed.

In theory, there are three degrees of freedom in all of the above cases, and therefore we can pick any three control points that are not collinear. We chose the above schemes simply because they are more symmetric, and therefore are less likely to cause undesired compensation effect due to excessive tilting of the affine reference frame. In the case of a corner point adjacent to six faces, either of the two schemes is permitted. We simply pick any three alternate points out of the six adjacent to the corner point.

Once a corner point region is prescribed, propagation will determine the corner point and its star completely. Corner point region propagations are computed based on the fact that the midpoints of the two diagonals of the parallelograms must agree. Because of the regularity of the equilateral triangles, the corner point is the midpoint of three pairs of star

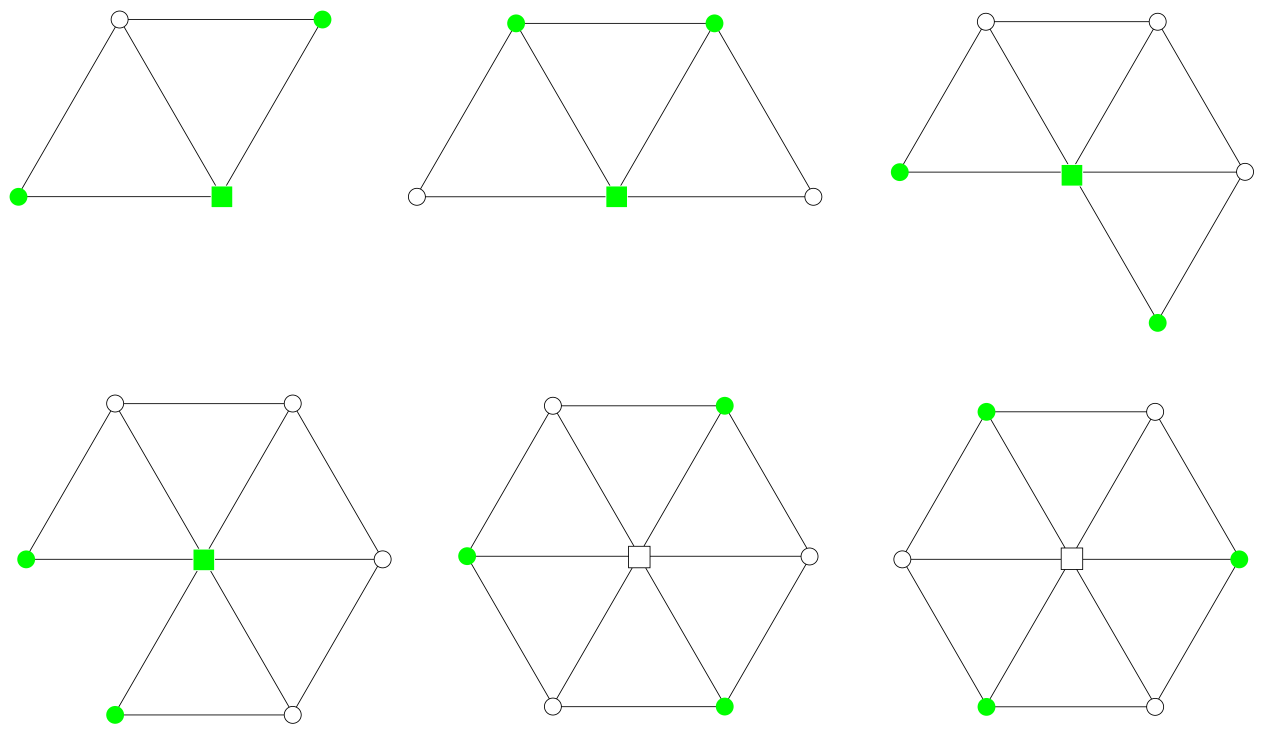

Figure 7: Prescribing corner points

points (in the case of six patches), and the center of gravity of any three alternate star points. These special properties simplify propagation somewhat and we take advantage of that. Propagation computations are simple. They consist of only one addition and one subtraction in most cases, and are instantaneous.

As shown in Figure 8, where a corner point region consisting of z_1 and its complete star $z_2, \ldots, z_7$ is given. We notice that z_2, z_4 and z_6 are prescribed. We first compute z_1. Since it is the center of gravity of z_2, z_4 and z_6,

$$z_1 = \frac{z_2 + z_4 + z_6}{3}.$$

Once z_1 is obtained, z_3, z_5 and z_7 follow immediately:

$$\begin{aligned} z_3 &= 2z_1 - z_6, \\ z_5 &= 2z_1 - z_2 \\ z_7 &= 2z_1 - z_4, \end{aligned}$$

z_1 is obtained with two additions and one division, and the rest one addition $(2z_1 = z_1 + z_1)$ and one subtraction.

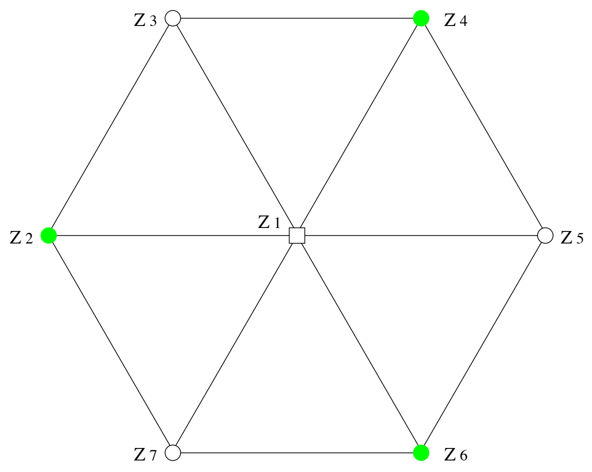

Figure 8: Prescribing corner points, example

5.1.3 Along the edges

What remains are continuity conditions along an edge, which are controlled by the parallelograms formed by inner points and edge points across that edge. Clearly all edge points directly affect the continuity conditions across that particular edge, but for every patch, only some of its inner points affect its C^1 continuity conditions. These are the inner points that form the aforementioned parallelograms.

For every two patches that share an edge, once the two corner point regions of the shared edge are computed, we need only to prescribe two inner points per patch, plus one inner point per shared edge to determine all the edge points.

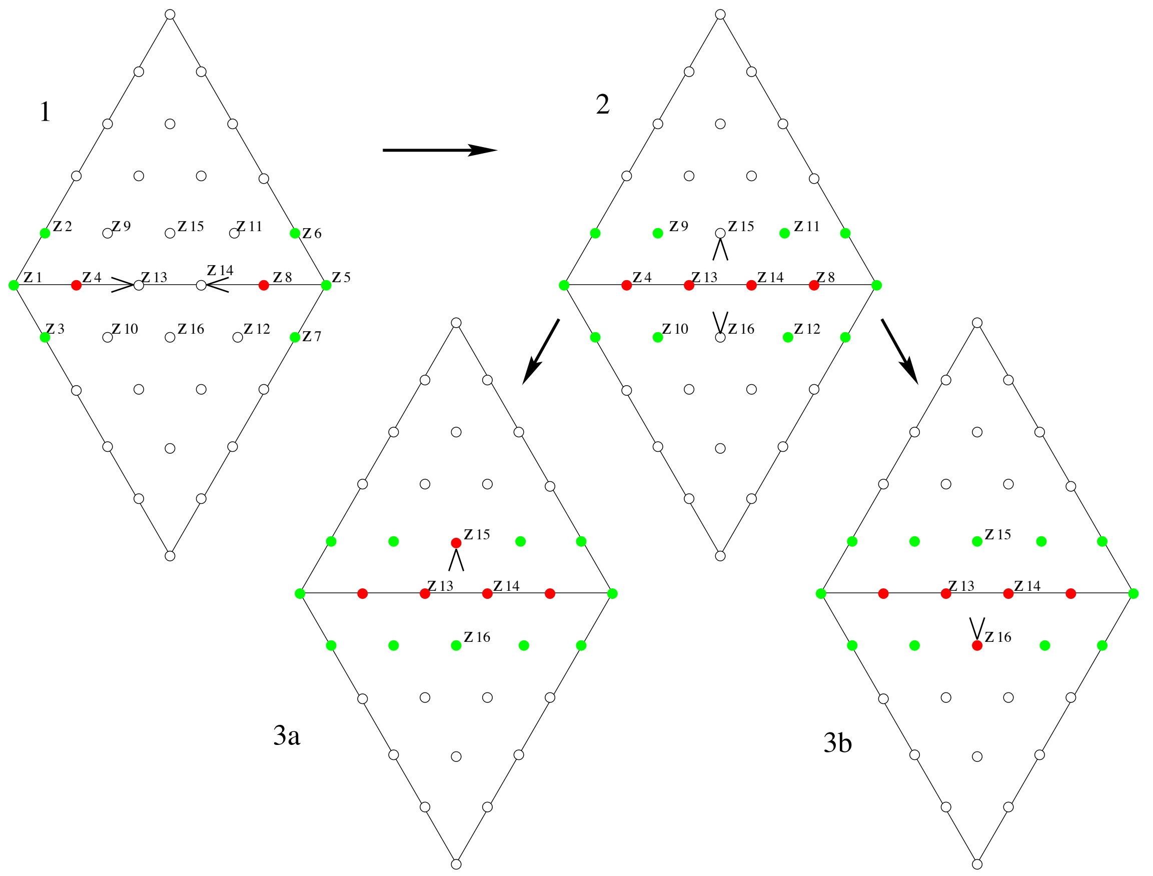

Figure 9: Prescriptions and propagations along the edges

Edge propagations are computed the same way as corner point region propagations. We force propagations along the shared edge from both ends to the center as shown in step $1 - 2$. Step 1 shows an edge whose two endpoint corner-point-region propagations $(z_1, z_2, z_3 \rightarrow z_4)$, $(z_5, z_6, z_7 \rightarrow z_8)$ are already completed. We adopt the convention of drawing prescribed control points in green, and propagated control points in red.

Next, following the directions of propagation as indicated by the two arrows, we prescribe two inner points at each side of the edge, namely, (z_9, z_{10}) and (z_{11}, z_{12}). Edge points z_{13} and z_{14} are then immediately computed via propagation in the following way:

$$z_{13} = z_9 + z_{10} - z_4,$$
$$z_{14} = z_{11} + z_{12} - z_8$$

At this stage we have z_{15} and z_{16} still to be propagated. It's clear that one of them needs to be prescribed, and the other propagated. Step 3_a and 3_b are both legal moves allowed by the algorithm. Prescribing either one of the inner points will lock the other in. We therefore call them *flip points*. The flip point is propagated either as $z_{15} = z_{13} + z_{14} - z_{16}$, or $z_{16} = z_{13} + z_{14} - z_{15}$, depending on which one we choose to prescribe. The two directions of propagations are caught and stopped by the flip points in the middle of the shared edge. All control points remaining (only inner and ear points) are free and they do not constrain continuity conditions.

5.1.4 Six patches

When we have a large surface with many patches, the case where six patches share a common corner point is typical except on the boundaries. In this case, three inner points per patch plus one flip point every two adjacent patches will suffice to determine continuity conditions across the patches. We require that corner point region propagations always precede edge propagations. All propagations are computed based on diagonal mid-point agreement of affected parallelograms.

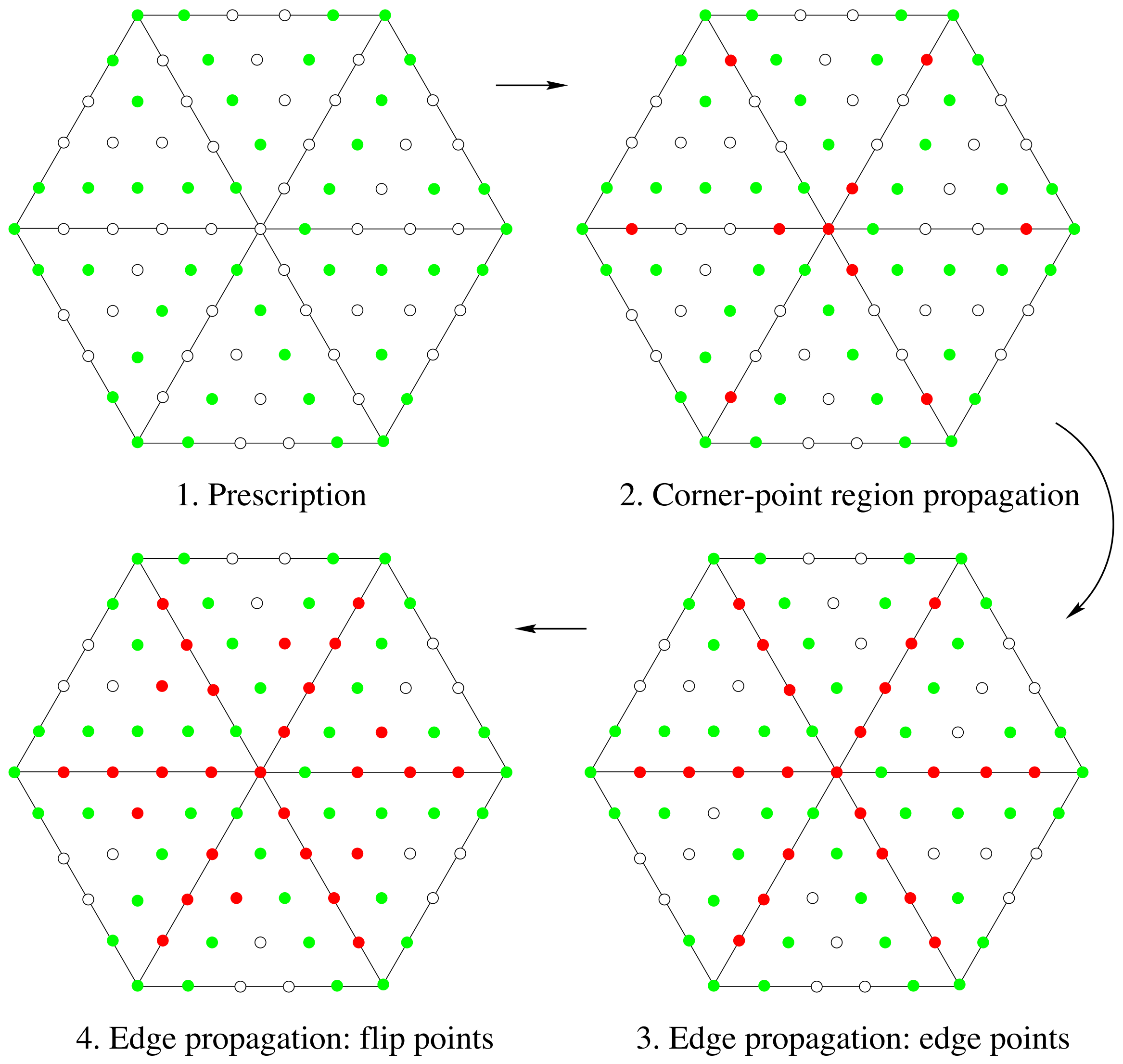
1. Prescription
2. Corner-point region propagation
3. Edge propagation: edge points
4. Edge propagation: flip points

Figure 10: When six patches come together

5.2 Irregular triangulations

So far, we have based our algorithm on equilateral triangulations, which are quite restrictive. The kind of surfaces they can represent tend to be geometrically regular and are not very realistic. We started with equilateral triangulations due to the nice properties brought by their regularity, but as we shall see, the properties of the equilateral triangles are not necessary, and our algorithm needs minimal modifications to adapt to irregular triangulations. In fact, equilateral triangulation is just a special case of irregular triangulation.

First, let's look at corner point region. From Lemma 4.6, we know that around the corner point region, there are always three degrees of freedom regardless how many patches come together. Therefore any three non-collinear control points among the corner point and its star can be prescribed. The three prescribed control points then form an affine frame. Therefore, propagation for any fourth point in this corner region is computed based on affine relations with respect to that particular frame.

Along the edges, we have similar quadrilaterals instead of parallelograms because the triangulation is no longer regular. Propagations still proceed in the same manner and directions, and involve control points with the same positions (or polar values) in their respective patches. The only thing that is different is in the actual computation. Just like in the corner point region, instead of mid-point calculations, we compute using affine relations.

The complexity of our algorithm stays the same, because the only thing that is different in the irregular case is the actual computation step of the propagation. The exact same sets of control points are prescribed and propagated, respectively. The computation based on affine relations is only slightly more complicated, which requires the computation of 2×2 determinants.

In Figure 11, following our convention, all the colored points, some labeled z_i, are control points in space, and the v_i belong to the domain triangulation, and are therefore in the plane.

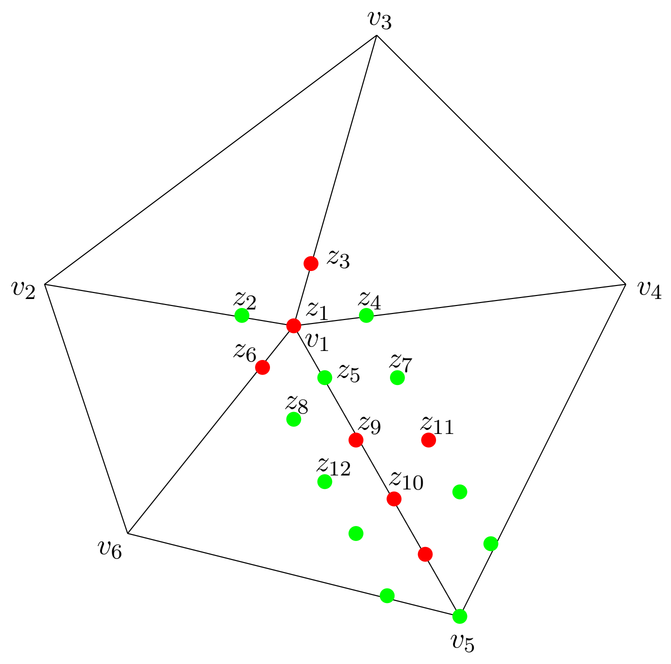

Figure 11: Irregular triangulation

In general, it is easily shown that given three affinely independent point (a, b, c) in $\mathbb{A}^2$, for any point $x \in \mathbb{A}^2$, if $x = \lambda a + \mu b + \nu c$, where λ, μ, ν are the barycentric coordinates of x with respect to (a, b, c),

$$\lambda = \frac{det(\overrightarrow{xb}, \overrightarrow{bc})}{det(\overrightarrow{ab}, \overrightarrow{ac})}, \quad \mu = \frac{det(\overrightarrow{ax}, \overrightarrow{ac})}{det(\overrightarrow{ab}, \overrightarrow{ac})}, \quad \nu = \frac{det(\overrightarrow{ab}, \overrightarrow{ax})}{det(\overrightarrow{ab}, \overrightarrow{ac})}.$$

As shown in the above example, the corner point region of z_1 and its star $z_2, \ldots, z_6$ has the three points (z_2, z_4, z_5) chosen as the affine frame. Computations for the rest of the control points in the corner point region is given as follows:

$$z_1 = \frac{det(\overrightarrow{v_1v_4}, \overrightarrow{v_4v_5})}{det(\overrightarrow{v_2v_4}, \overrightarrow{v_2v_5})} z_2 + \frac{det(\overrightarrow{v_2v_1}, \overrightarrow{v_2v_5})}{det(\overrightarrow{v_2v_4}, \overrightarrow{v_2v_5})} z_4 + \frac{det(\overrightarrow{v_2v_4}, \overrightarrow{v_2v_1})}{det(\overrightarrow{v_2v_4}, \overrightarrow{v_2v_5})} z_5,$$
$$z_3 = \frac{det(\overrightarrow{v_3v_4}, \overrightarrow{v_4v_5})}{det(\overrightarrow{v_2v_4}, \overrightarrow{v_2v_5})} z_2 + \frac{det(\overrightarrow{v_2v_3}, \overrightarrow{v_2v_5})}{det(\overrightarrow{v_2v_4}, \overrightarrow{v_2v_5})} z_4 + \frac{det(\overrightarrow{v_2v_4}, \overrightarrow{v_2v_3})}{det(\overrightarrow{v_2v_4}, \overrightarrow{v_2v_5})} z_5,$$
$$z_6 = \frac{det(\overrightarrow{v_6v_4}, \overrightarrow{v_4v_5})}{det(\overrightarrow{v_2v_4}, \overrightarrow{v_2v_5})} z_2 + \frac{det(\overrightarrow{v_2v_6}, \overrightarrow{v_2v_5})}{det(\overrightarrow{v_2v_4}, \overrightarrow{v_2v_5})} z_4 + \frac{det(\overrightarrow{v_2v_4}, \overrightarrow{v_2v_6})}{det(\overrightarrow{v_2v_4}, \overrightarrow{v_2v_5})} z_5,$$

Along the edge, (z_5, z_7, z_8) will be the affine frame used to compute z_9, and similarly, (z_9, z_{10}, z_{11}) will give us the flip point z_{12}.

We can handle most irregular triangulations, but not arbitrary triangulations. We are limited by one case. We will discuss that in some detail in 7.

5.3 Local control

Locality is an important property for interactive design systems. The patches affected by the modification of a single control vertex, the less recomputation is needed, and therefore the faster the rendering speed. When it comes to locality, *complete locality* is highly desired, which means modification of a control vertex only affects patches that are incident to it.

Our algorithm generates propagations that are completely local. Local control is achieved because propagations always end at the flip points. Modification of any control point on any patch F will affect at most those control points associated with the patches that are adjacent to F. Only some of the control points of these incident patches are affected at all times, and furthermore, not all incident patches are affected under most circumstances.

Depending on the nature of the modified control point (corner, inner or flip), more or less repropagation is needed. Only control points that were prescribed in the original design are allowed to be modified.

5.3.1 Corner point region

Modification of a corner point or any of its star will result in repropagations of the entire corner point region and along all of its affected adjacent edges.

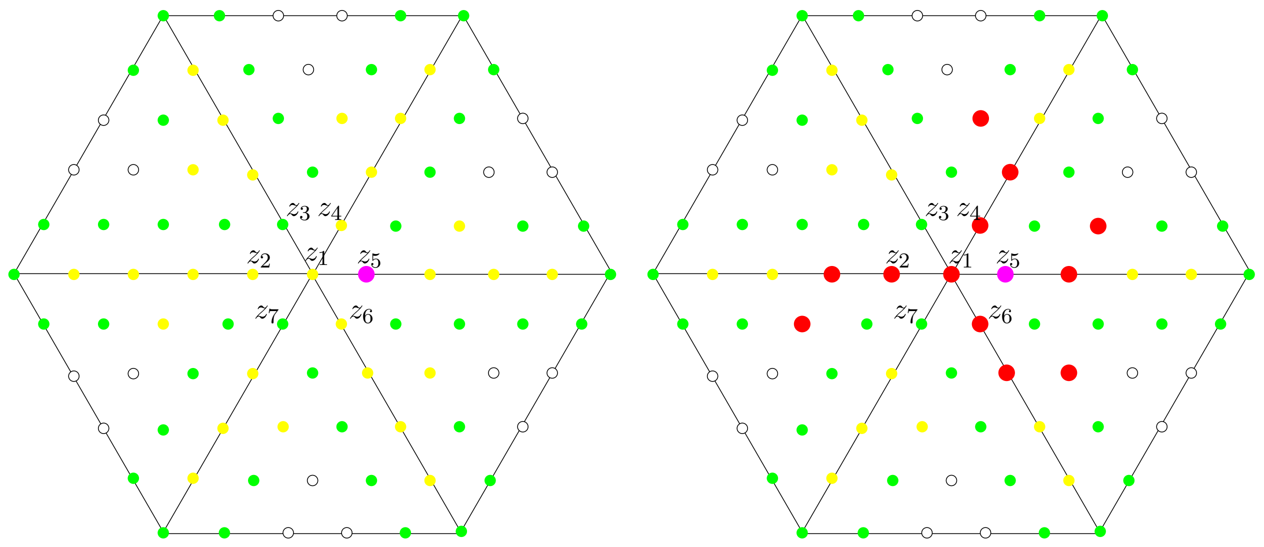

Figure 12: Local propagation in corner point region

In Figure 12, control point z_5 is modified. We adopt the convention of drawing modified control points in magenta. Since z_5 was one of the three points that formed the affine frame in that particular corner point region, once the affine frame changed, all the other control points in the same corner point region that do not belong to the affine frame, namely z_1, z_2, z_4 and z_6 need to be recomputed with respect to the new frame.

Notice that the example shows only one modified control point. All three control points that make up the frame are allowed to be modified. The changes made to z_2, z_4, z_6 and also the original z_5 will in turn affect the edges associated to them, e_2, e_4, e_6 and e_5 respectively. Edges e_3 and e_7 are not affected, as z_3 and z_7 are not modified.

Repropagation along any of the above edges is in fact only propagated through half of the edge. We know that there are two directions of propagation along any edge, but from any one corner point region, only one side of an edge is affected. Either direction is stopped by the flip point, and cannot proceed further.

Repropagation of any one corner region and its adjacent edges does not affect any other corner regions, as the flip points stops the propagation before it can reach any other corner region. This means that all corner point region propagations can be carried out in parallel, or when programmed sequentially, in any arbitrary order.

5.3.2 Inner points

Modification of an inner point (not a flip point) will result in repropagations along its two incident edges, as illustrated in Figure 13.

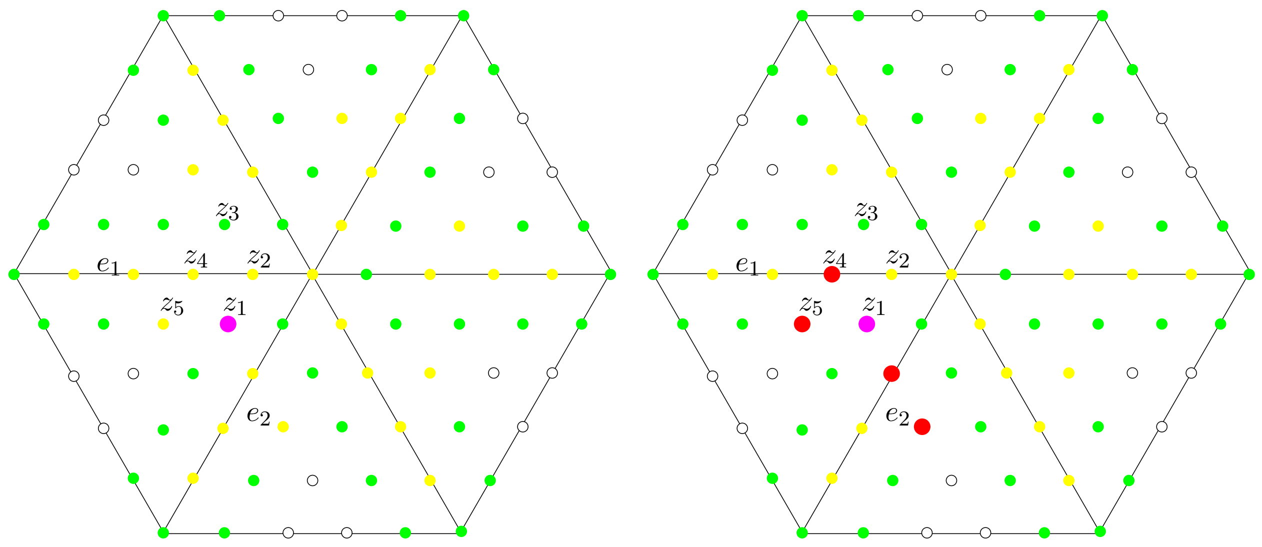

Figure 13: Local propagation along the edges

In Figure 13, inner point z_1 is modified. As a result, on edge e_1, the affine frame (z_1, z_2, z_3) has changed, and therefore z_4 needs to be recomputed. The changes to z_4 in turn affect the affine frame used to compute flip point z_5, therefore z_5 has to be recomputed as well. No other changes occurred on e_1. Similarly on edge e_2, one edge point and one flip point need to be recomputed.

Repropagation along an edge will not affect any other edges or corner point regions, again because the flip points "sinks" the propagations from both ends of the edge. All edge propagations can also be carried out in parallel, or in any arbitrary order.

5.3.3 Flip points

Modification of a flip point will result in repropagation of the other flip point across the edge, as illustrated in Figure 14.

In Figure 14 flip point z_1 is modified. As a result, the affine frame (z_1, z_2, z_3) is changed, and therefore z_4 needs to be recomputed with respect to the new frame. No other control points are affected.

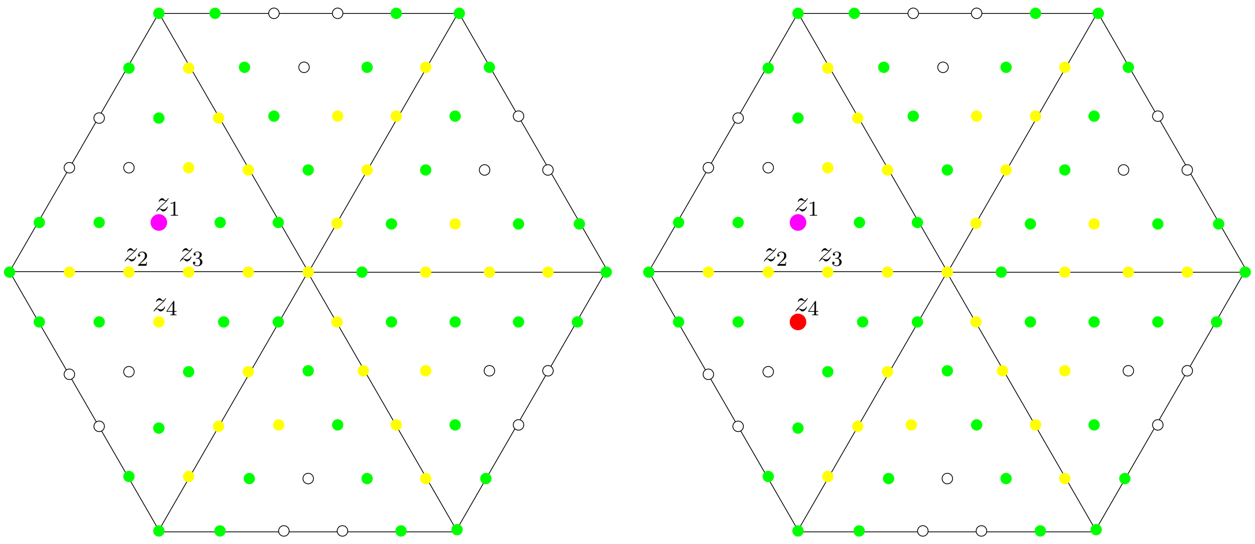

Figure 14: Local propagation of flip points

Repropagation of a flip point does not affect any corner point regions or any edges. Therefore, when a set of control points is modified, first repropagate all affected corner regions in any order, then along the affected edges in any order, finally recompute affected flip points in any order. This order (corner first, edge second, flip last) assures that all modifications take effect as they should.

6 Results

Software is written in C and OpenGL to fully implement the algorithm. Tests are conducted on with hand-built triangulations and some randomly chosen initial values for the control points that need to be prescribed. Often the choices are simply a weighted interpolation of the corner points based on their individual polar coordinates. The constrained control points are then propagated according to the algorithm outlined in the previous section.

If testing on a given mesh then the control points that need prescription will take on the coordinates from the original mesh. It is difficult to find open meshes that are freely available to test on. Most of the public repositories contain only closed meshes. Thus most of the testing were done with hand-built examples and they are therefore small. However we believe that the effectiveness of the algorithm can nevertheless be demonstrated and the care we have take to minimize the computation time for control point modification and propagation assures that the method will work readily on larger models.

6.1 Six patches

Show in the following figures (Figure 15) is a sample modification process of six equilateral triangular patches. In the original configuration, control points shown in green are free and can be modified. Those shown in yellow are constrained. In the second figure, control points modified are shown in magenta, those that are propagated are show in red.

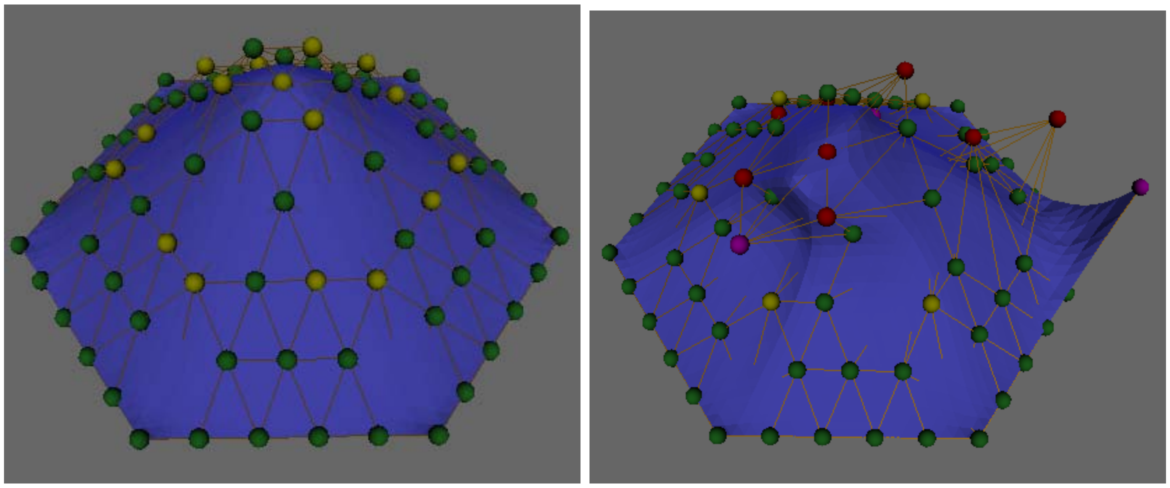

Figure 15: 6 patches modified

All modifications are done in real time and require very little computation time. Patches are rendered with two levels of subdivision iteration. This coarse resolution is used to facilitate on-screen real-time modification of large surface models. Once changes are finalized, more levels of iteration can be introduced to produce a final smooth model.

6.2 Six irregular patches

Show in the Figures 16 and 17 is an irregular triangulation with six patches. The smooth models are rendered with four iteration levels of subdivision.

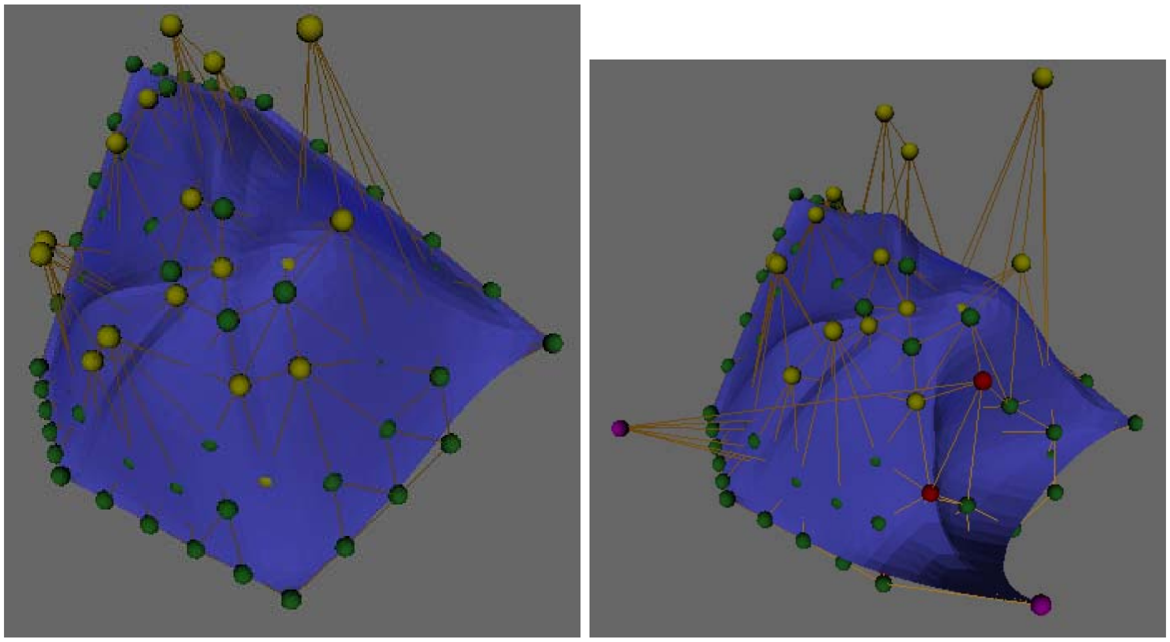

Figure 16: 6 irregular patches modified

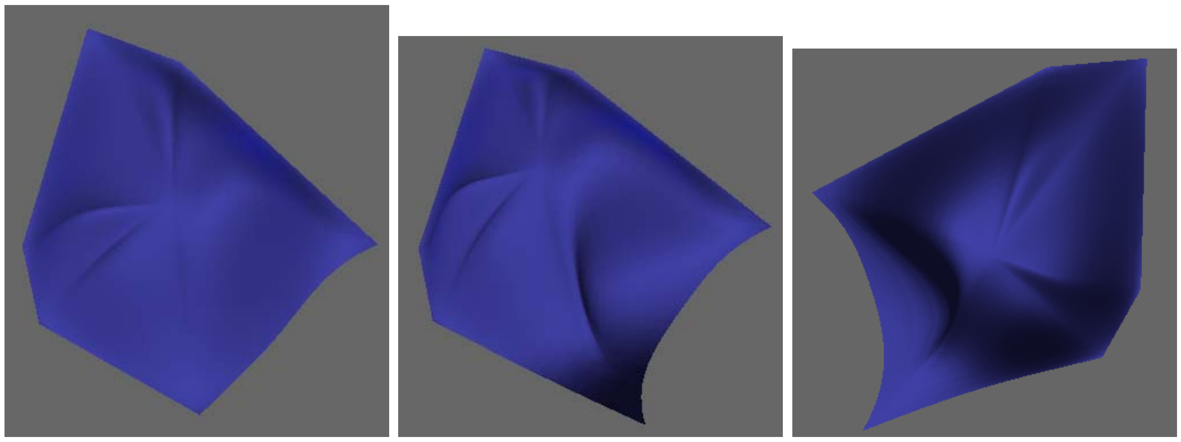

Figure 17: Smooth rendering of these patches

6.3 Nine patches with a hole

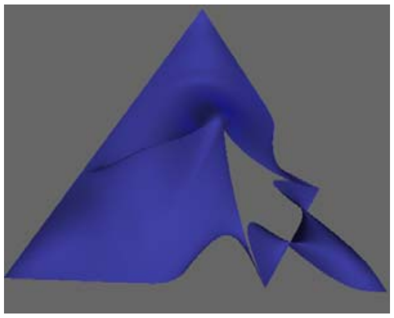

Figure 18: Smooth rendering of nine patches with a hole

7 Summary and future work

We believe that our algorithm provides a simple, fast way to design surfaces with built-in C^1 continuity. Since our method is based on triangular patches, there is no restriction on how many patches may come together at a control point, and thus it offers more freedom in terms of modeling capability. Furthermore, there are enough free (manipulatable) control points per patch (typically about $5 - 6$) such that very local changes might be made.

The triangular control net the algorithm generates lends very well to the subdivision version of the de Casteljau algorithm for evaluation. Given the minimum degree we require, 5, usually two iterations are enough to generate a good approximation of the surface. To manipulate and modify surface splines, it suffices to locally move control points. Recomputation of control points is limited to the local vicinity, and is therefore simple and easily implemented.

Complex topologies involving holes or sharp corners are automatically handled by the algorithm without extra work, because they do not arise as special cases. As long as the template triangulation reflects the existence of such topologies, they are handled properly. All the computations involved in the algorithm are just the propagations of the control points, which in most cases consist of just one addition and one subtraction.

Bellow are some directions we are taking to extend this work.

7.1 C^2

Preliminary experiment results have shown that our algorithm can likely be extended to construct C^2 triangular surfaces. The extension is in theory straight-forward, but working out an actual local incremental algorithm analogous to what we have shown so far is not trivial.

It turns out that at least degree 8 patches are needed everywhere for C^2, and in fact we need to move to degree 9 to have enough freedom locally to design a propagation scheme. Here, in addition to prescribing the first row cross-boundary control points, some second row cross-boundary control points will need to be prescribed as well. A balanced prescription scheme needs to be worked out to avoid shape defects.

7.2 Interpolation

The extension of our scheme to interpolate instead of approximate is almost instantaneous. Because we have three degrees of freedom around any corner-point region, including the corner point itself, we have enough freedom to simply force the corner point to pass through desired coordinates.

However, this does have the potential to lead to shape defects due to the problem mentioned in the last subsection, tilting of local affine frame, that is, if the choice of the corner-point's placement is unfortunate, it will result in undesired undulations in that corner point region. We will discuss this more in Subsection 7.2.3.

Also, recall that our C^1 algorithm requires the existence of a parameterization in the plane. If we assume that such a parameterization has been given, the extension to handle interpolation is indeed just as described above. We have also run some successful tests based on this assumption. However, in most cases, it is unrealistic to assume a given parameterization for interpolation purposes. The input will be a collection of data points given in space, although it is probably reasonable to assume that it is already meshed. We will need to come up with a reasonable parameterizaion of this mesh in the plane.

This is a non-trivial problem. A simple projection to a plane is not enough. Even with planes that are reasonably fitted to the data points, such as a least-squares fitting, it is still not good enough to generate a parameterizaion with no self-intersection. The problem arises when the mesh describe a surface that has folds (but no self-intersection). Any projection will not deal with the folds correctly. A reasonable parameterization in the plane will need to stretch out the folds in the mesh.

Gu, Gortler and Hoppe [GGH02] worked on remeshing irregular triangular meshes with meshes with (semi)-regular connectivity. In their work of representing surfaces with geometry images, they have worked on methods of cutting a mesh to open it into a topological disk, which then gives a parameterization of that mesh within this disk. This is quite possibly exactly what we can use to find a parameterization of our input mesh. Closer examination of Gu et al's methods is needed.

Of course, we are also aware of some restrictions of our method, some of which we are working on to resolve in the near future.

7.2.1 Shape parameters and automated shape control

Our algorithm is highly local and leaves quite a bit of freedom in the interior of the patches and the boundaries. The disadvantage of this is that, in general, a fair number of control points need to be modified before a change to a single patch will be uniformly distributed. This is definitely not a task a user should handle. An automatic method is needed to prescribe a set of appropriate control points, given a request for shape change. That is, some type of shape parameter needs to be set up.

A related issue is to develop an automatic method to prescribe all the control points needed for propagation when given large sets of data to approximate. In this case, it is important to set good initial values for optimal first approximation so that less modification is needed from the user. It is still current research to set the initial free parameters to incorporate as much of the known discrete surface geometry as possible.

7.2.2 Degenerate frames along the edges

First, we mentioned in Section 5.2 that we cannot handle just any arbitrary triangulation, we are limited by one case. Recall that all propagations need the existence of at least three affinely independent points. Around a corner point region, this is guaranteed. But along the edges, because we have fixed directions of propagation, we do not have the freedom to choose which three points from the four control points that form the similar quadrilaterals will be our affine frame. They are fixed. In the unfortunate cases when these three points are collinear, the algorithm breaks down.

As shown in Figure 19, the points v_2, v_1, v_4 and v_3, v_1 v_5 are respectively collinear in the domain triangulation. The corner point region of z_1 and its star $z_2, \ldots, z_5$ is not affected, and in the example, z_3, z_4, z_5 are chosen as the affine frame, and z_1 and z_2 are then obtained through propagation. But then, the edge propagations immediately present a problem. For example, along e_1, the algorithm dictates that z_6, z_2, z_9 shall form an affine frame, resulting in z_{10}'s propagation. However, z_6, z_2, z_9 are given by the same affine relations as v_3, v_1, v_5, which are collinear. So are frames (z_6, z_3, z_7), (z_7, z_4, z_8), (z_8, z_5, z_9) along edges e_2, e_3 and e_4, respectively. Thus, control points z_{10}, z_{11}, z_{12} and z_{13} cannot be computed. In this case, we have two straight lines going through v_1/z_1. In general, any straight line that goes through a corner point will cause similar problem for two edge propagations adjacent to that particular corner point.

There are two possible fixes for this problem. We can detect it and then change the direction of propagations locally where such degenerate frames exist. Notice that because our domain is a triangulation, there is always at least one direction along an edge that does

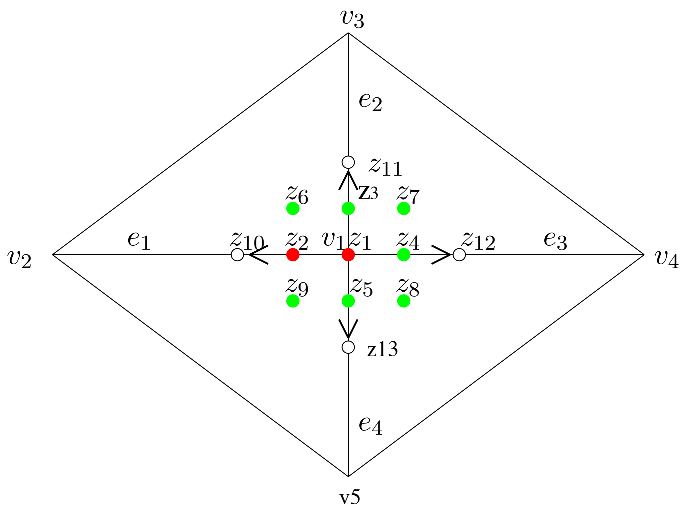

Figure 19: Frames along the edges are collinear

not contain degenerate frames.

Another approach is to leave the algorithm alone, but preprocess to remove the degenerate frames. We simply need to move v_1 slightly off to the side, to disrupt the straight lines. The move is therefore a small one, and will not cause noticeable difference in the final surface generated. Any irregular triangulation will need to be preprocessed to remove such degenerate cases.

7.2.3 Excessive tilting of the affine frames

This is the most serious problem that we have encountered so far. Recall that both around the corner point region and across the edges, propagation places control points on a plane determined by the affine frame formed by the three affinely independent control points that we prescribed. This means that the placement of this plane relative to the normals of the surface in that region is extremely important to the final shape of the surface.

In general, we want this plane to be tangent to the surface as much as possible. That is why in the G^1 literature it is often called the “cross-boundary tangent plane”. If by some unfortunate choice (most likely due to interpolation), the affine frame is tilted towards the normal of the surface, then the resulting surface will contain severe undulations.

Fortunately, even in the case of interpolation, we still have two other control points which we can prescribe to mitigate the tilt, if it occurs. The best method of fairing is currently being researched.

References

[Aki78] H. Akima. A method of bivariate interpolation and smooth surface fitting for irregularly distributed data points. *ACM Transaction on Mathematical Software*, 4:148–159, 1978.

[BD88] A. A. Ball and Storry D.J.T. Conditions for tangent plane continuity over recursively generated B-spline surfaces. *ACM Transactions on Graphics*, 10(6):113–119, 1988.

[Ber90] Marcel Berger. *Géométrie 1.* Nathan, 1990. English edition: Geometry 1, Universitext, Springer Verlag.

[BLZ00] H. Biermann, A. Levin, and D. Zorin. Piecewise smooth subdivision surfaces with normal control. In *SIGGRAPH '00 Conference Proceedings, Annual Conference Series*, pages 113–119. ACM SIGGRAPH, 2000.

[CDM91] A.S. Cavaretta, W. Dahmen, and C.A. Micchelli. Stationary subdivision. *Memoirs American Mathematical Society*, 93:1–186, 1991.

[CJ78] E. Catmull and Clark J. Recursively generated B-spline surfaces on arbitrary topological meshes. *Computer Aided Design*, 10(6):350–355, 1978.

[dC86] Paul de Faget de Casteljau. *Shape Mathematics and CAD.* Hermes, first edition, 1986.

[Doo78] D. Doo. A subdivision algorithm for smoothing down irregularly shaped polyhedrons. In *Proceedings on Interactive Techniques in Computer Aided Design*, pages 157–165, 1978.

[DS78] D. Doo and M. Sabin. Analysis of the behavior of recursive division surfaces near extraordinary points. *Computer Aided Design*, 10(6):356–360, 1978.

[DS90] W.H. Du and F. Schmitt. On the G^1 continuity of piecewise Bézier surfaces: A review with new results. *Computer-aided Design*, 22:556–573, 1990.

[Far82] G. Farin. A construction for the visual C^1 continuity of polynomial surface patches. *Computer Graphics and Image Processing*, 20:272–282, 1982.

[Far83] G. Farin. Smooth interpolation to scattered 3D data. In R.E. Barnhill and W. Boehm, editors, *Surfaces in CAGD*, pages 43–63. North-Holland, 1983.

[Far92] Gerald Farin. *Curves and Surfaces for CAGD.* Academic Press, third edition, 1992.

[Far95] Gerald Farin. *NURB Curves and Surfaces, from Projective Geometry to practical use.* AK Peters, first edition, 1995.

[FN83] R. Franke and G. Nielson. Surface approximation with imposed conditions. In R.E. Barnhill and W. Boehm, editors, *Surfaces in CAGD*, pages 135–146. North-Holland, 1983.

[Fra82] R. Franke. Scattered data interpolation: Tests of some methods. *Mathematics of Computation*, 38:181–200, 1982.

[Gal00a] Jean H. Gallier. *Curves and Surfaces in Geometric Modeling.* Morgan Kaufman, first edition, 2000.

[Gal00b] Jean H. Gallier. *Geometric Methods and Applications.* Springer, first edition, 2000.

[GCR77] C.M. Gold, J.D. Charters, and J. Ramsden. Automated contour mapping using triangular element data structures and an interpolant over each irregular triangular domain. *Computer Graphics*, 11:170–175, 1977.

[GGH02] X. Gu, S. Gortler, and H. Hoppe. Geometry images. In *SIGGRAPH '02 Conference Proceedings, Annual Conference Series.* ACM SIGGRAPH, 2002. To appear in SIGGRAPH '02.

[GR74a] W.J. Gordon and R.F. Riesenfeld. B-splines curves and surfaces. In R.E. Barnhill and R.F. Riesenfeld, editors, *Computer Aided Geometric Design*, pages 95–126. Academic Press, New York, 1974.

[GR74b] W.J. Gordon and R.F Riesenfeld. Bernstein-Bézier methods for the computer aided design of free form curves and surfaces. *ACM*, 21:293–310, 1974.

[GS93] Günther Greiner and Hans-Peter Seidel. Modeling with triangular b-splines. In *2nd ACM Symposium on Solid Modeling*, pages 211–220, 1993.

[Hag86] H. Hagen. Geometric surface patches without twist constraints. *Computer Aided Geometric Design*, 3:179–184, 1986.

[HB03] S. Hahmann and G.P. Bonneau. Polynomial surfaces interpolating arbitrary triangulations. *IEEE Transactions on Visualization and Computer Graphics*, 9:99–109, 2003.

[HBT00] S. Hahmann, G.P. Bonneau, and R. Taleb. Localizing the 4-split method for G^1 free-form surface fitting. Technical report, University of Grenoble, 2000. To appear in Computing, 2001.

[HBT01] S. Hahmann, G.P. Bonneau, and R. Taleb. Modélisation de surfaces de topologie arbitraire. In *SIAM Conference on Geometric Design and Computing*, page 37, 2001. Abstract only.

[HDD$^+$94] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Piecewise smooth surface reconstruction. In *SIGGRAPH '94 Conference Proceedings, Annual Conference Series*, pages 295–302. ACM SIGGRAPH, 1994.

[HH89] Hagen H. and Pottmann H. Curvature continuous triangular interpolants. In T. Lyche and L. Schumaker, editors, *Mathematical Methods in Computer Aided Geometric Design*, pages 373–384. Academic Press, 1989.

[HL93] J. Hoschek and D. Lasser. *Computer Aided Geometric Design.* AK Peters, first edition, 1993.

[Jen87] T. Jensen. Assembling triangular and rectangular patches and multivariate splines. In G. Farin, editor, *Geometric Modeling: Algorithm and New Trends*, pages 203–222. SIAM, 1987.

[Kob01] L. Kobbelt. $\sqrt{3}$-subdivision. In *SIGGRAPH '01 Conference Proceedings, Annual Conference Series*, pages 103–112. ACM SIGGRAPH, 2001.

[Law77] C.L. Lawson. Software for C^1 surface interpolation. In J.R. Rice, editor, *Software III*, pages 159–192. Academic Press, 1977.

[Loo87] Charles Loop. *Smooth Subdivision Surfaces based on triangles.* PhD thesis, University of Utah, Department of Mathematics, Salt Lake City, Utah, 1987. Masters's thesis.

[Loo94] Charles Loop. A G^1 triangular spline surface of arbitrary topological type. *Computer Aided Geometric Design*, 11:303–330, 1994.

[MLM$^+$92] S. Mann, C. Loop, Lounsberry M., D. Meyers, Painter J., DeRose T., and K. Sloan. A survey of parametric fitted data: Fitting using triangular interpolants. In H. Hagen, editor, *Curve and Surface Design*, pages 145–172. SIAM, 1992.

[MS92] D. Micchelli and H.-P. Seidel. Blossoming begets B-spline bases built better by B-patches. *Mathematics of Computation*, 59:97–115, 1992.

[Pet91] J. Peters. Smooth interpolation of a mesh of curves. *Constructive Approximation*, 7:221–246, 1991.

[Pet95] J. Peters. Biquartic C^1-surface splines over irregular meshes. *Computer Aided Geometric Design*, 27:895–903, 1995.

[Pip87] B. Piper. Visually smooth interpolation with triangular Bezier patches. In G. Farin, editor, *Geometric Modeling: Algorithm and New Trends*, pages 221–234. SIAM, 1987.

[PS95] P Pfeifle and H.-P. Seidel. Fitting triangular B-splines to functinal scattered data. In *Graphics Interface '95*, pages 80–88, 1995.

[Ram87] L. Ramshaw. Blossoming: A connect-the-dots approach to splines. Technical report, Digital SRC, Palo Alto, CA 94301, 1987. Report No. 19.

[Ram88] L. Ramshaw. Bézier and B-splines as multiaffine maps. In RA Earnshaw, editor, *Theorectical Foundations of Computer Graphics and CAGD*, pages 757–776. Springer, 1988.

[Ram89] L. Ramshaw. Blossoms are polar forms. *Computer Aided Geometric Design*, 13:38–46, 1989.

[Ris92] J.-J. Risler. *Mathematical Methods for CAD*. Masson, first edition, 1992.

[Sar00] R. F. Sarraga. A variational method to model G^1 surfaces over triangular meshes of arbitrary topology in R^3. *ACM Transactions on Graphics*, 19:279–301, 2000.

[Sei88] H.-P. Seidel. Knot insertion from a blossoming point of view. *Computer Aided Geometric Design*, 5:81–86, 1988.

[Sei89] H.-P. Seidel. A new multiaffine approach to B-splines. *Computer Aided Geometric Design*, 6:23–32, 1989.

[Sei93] H.-P. Seidel. Polar forms for geometrically continuous spline curves of arbitrary degree. *ACM Transactions on Graphics*, 12(1):1–34, 1993.

[She68] D. Shepard. A two-dimentional interpolation function for irregularly spaced data. In *Proc. 23rd Nat. Conf. ACM*, pages 517–523, 1968.

[SS88] L. Shirman and C. Sequin. Local surface interpolation with Bezier patches. *Computer Aided Geometric Design*, 4:279–296, 1988.

[Wey46] Hermann Weyl. *The Classical Groups. Their Invariants and Representations.* Princeton Mathematical Series, No. 1. Princeton University Press, second edition, 1946.

[Xu02] Dianna Xu. *INCREMENTAL ALGORITHMS FOR THE DESIGN OF TRIANGULAR-BASED SPLINE SURFACES*. PhD thesis, University of Pennsylvania, Department of Computer and Information Science, Philadelphia, PA, 2002.

[YHB05] A. Yvart, S. Hahmann, and G.P. Bonneau. Smooth adaptive fitting of 3d models using hierachical triangular splines. In *Shape Modeling International 2005 Conference Proceedings*, pages 13–22, 2005.

[Zor96] D. Zorin. *C^k Continuity of Subdivision Surfaces.* PhD thesis, California Institution of Technology, 1996. Ph.D. thesis.