An Improved Parametric Side-Vertex Triangle Mesh Interpolant

Stephen Mann
Computing Science Department
University of Waterloo
200 University Ave W.
Waterloo, Ontario
Canada N2L 3G1

e-mail: smann@cgl.uwaterloo.ca

Abstract

There are many schemes for fitting triangular surface patches to a triangular net of data. In general, local schemes produce surfaces with poor surface quality. Although variational techniques construct surfaces of higher quality, such techniques tend to be computationally expensive. In this paper, I will present modifications to Nielson's side-vertex method that improve its surface quality without resorting to variational techniques.

Résumé

Il existe de nombreuses méthodes pour ajuster des pièces de surface triangulaire a une maille triangularisée. En général, les méthodes locales produisent des surfaces de mauvaise qualité. Bien que les techniques variationelles produisent de meilleurs résultats elles sont habituellement dispensieuses à calculer. Dans cet article, je vais présenter plusieurs modifications de la méthode de Nielson permettent d'augmenter la qualité des surfaces sans faire appel aux techniques variationelles.

Keywords: Parametric surfaces

Introduction

There are many variations of the data fitting problem. One class of techniques fit a surface to a triangulated set of data. This data may come from a surface that lies over a plane, in which case a function of the form $z = f(x, y)$ can be fit to the data. Often, however, the mesh of data will fold back on itself, necessitating the use of a *parametric scheme*, where we must find three functions of two parameters, $x(u, v)$, $y(u, v)$, and $z(u, v)$, that interpolate the data. Techniques for fitting parametric surfaces to a triangular mesh can be classified by the continuity of the surface they construct (e.g., C^0, C^1, C^∞) and by the amount of data considered when constructing a part of the surface (e.g., local schemes, which only consider data near the region being constructed, verses global schemes, which consider all of the data when constructing any part of the surface).

A large number of local parametric triangular surface schemes have been developed over the past fifteen years (see [6, 11] for a survey of such schemes). Surprisingly, all of these schemes exhibit similar shape defects. On closer inspection, it is seen that these schemes all have a large number of free parameters that are set using simple heuristics. By manually adjusting these parameters, one can improve the shape of the surfaces [8].

One way to improve the shape of the constructed surfaces automatically is to use variational methods. Several authors have used such schemes to improve the shape of the constructed surfaces, but usually at a high computational cost due to the global nature of the solution (e.g., [12]). Similarly, we can use *local optimization* methods to set the free parameters and improve the shape, although the results are not as good as the global methods [9].

In this paper, I will consider a local, parametric, triangular surface fitting scheme that constructs non-polynomial surface patches that meet with tangent plane continuity. My method is a modification of Nielson's side-vertex scheme [14], and relies heavily on the hybrid, scattered data fitting scheme of Foley and Opitz [3]. The resulting surfaces show large improvement in shape over other local, parametric, triangular surface fitting techniques. More precisely, given a triangle of data (a set of three vertices each with an associated normal), my scheme constructs a parametric patch that interpolates the positions and normals at the corners. When used to fill a triangular mesh, the resulting surface patches meet with tangent plane continuity.

The paper begins with a review of Bézier curves and triangular Bézier surfaces. Next, Nielson's side-vertex scheme is presented, followed by a description of the Foley-Opitz hybrid Bézier patch for functional data. In main section of the paper, I present improvements to

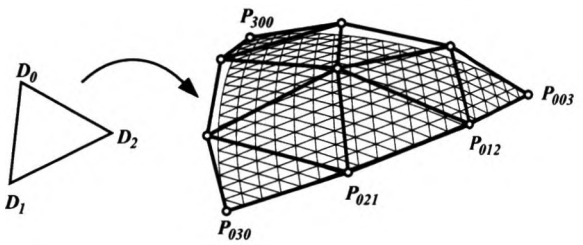

Figure 1: A triangular cubic Bézier patch

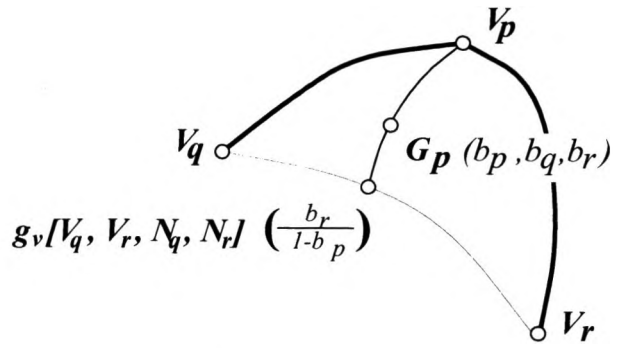

Figure 2: Side-vertex method.

Nielson's scheme (with elaborations on some important details appearing in the following section), and in the final section I present some results.

Background

A parametric, cubic Bézier curve parameterized over the interval $[0, 1]$ is given by the formula

$$\boldsymbol{B}(t) = \sum_{i=0}^{3} \boldsymbol{P}_i B_i^3(t),$$

where the $\boldsymbol{P}_i$ are control points, and $B_i^3(t) = \binom{3}{i}(1-t)^{3-i}t^i$. The two important properties of Bézier curves for this paper are (i) they interpolate their end points (i.e., $\boldsymbol{B}(0) = \boldsymbol{P_0}$ and $\boldsymbol{B}(1) = \boldsymbol{P_3}$); (ii) the difference of the first two (last two) control points gives the derivatives at the ends of the curve (i.e., $\boldsymbol{B'}(0) = 3(\boldsymbol{P_1} - \boldsymbol{P_0})$ and $\boldsymbol{B'}(1) = 3(\boldsymbol{P_3} - \boldsymbol{P_2})$).

There are two common generalizations of Bézier curves to surfaces: Tensor product surfaces and triangular Bézier patches. In this paper, we are concerned with the latter generalization. A cubic triangular Bézier patch is, again, a weighted combination of control points:

$$\boldsymbol{B}(\boldsymbol{t}) = \sum_{\vec{i}} \boldsymbol{P}_{\vec{i}} B_{\vec{i}}^3(\boldsymbol{t}).$$

Here, $\vec{i} = (i_0, i_1, i_2)$, where i_0, i_1, i_2 are non-negative integers that sum to 3. The domain of a triangular Bézier patch is a triangle, and we use the barycentric coordinates of a domain point $\boldsymbol{t} = (t_0, t_1, t_2)$ to evaluate the generalized Bernstein polynomials:

$$B_{\vec{i}}^3(\boldsymbol{t}) = \frac{n!}{i_0!i_1!i_2!} t_0^{i_0} t_1^{i_1} t_2^{i_2}.$$

The control points of a triangular Bézier patch are arranged in a triangular net (Figure 1). The important properties of triangular Bézier patches for this paper are (i) they interpolate their corner control points; (ii) the boundary curves of the patch are cubic Bézier curves whose control points are the boundary points of the triangular control net (e.g., the boundary $t_0 = 0$ is the Bézier curve whose control points are $\boldsymbol{P}_{\vec{i}}$ with $\vec{i} = (0, i_1, i_2)$); (iii) the cross-boundary derivatives along one boundary are given by two layers of control points (e.g., for $t_0 = 0$, the cross-boundary derivatives are given by the control points $\boldsymbol{P}_{\vec{i}}$ with $\vec{i} = (0, i_1, i_2)$ and $\vec{i} = (1, i_1, i_2)$).

Although in this paper we are concerned with parametric surfaces, I will use ideas from a functional surface fitting scheme to modify an existing parametric scheme. In the functional setting, we can write z as a function of x and y (e.g., $z = f(x, y)$). Bézier curves and surfaces can be used to represent functional polynomials. In the case of surfaces, the x-y coordinates of the control points are distributed uniformly over the domain triangle. The only unknowns are the z coordinates of these control points. To distinguish between the scalar and parametric cases, I will use italic symbols for scalars, and bold-italic for tuples.

For a further description of Bézier curves and surfaces, see any text on Computer Aided Geometric Design (e.g., Farin's book [2]).

Nielson's Scheme

Nielson [14] developed a parametric *side-vertex* method to fit a piecewise smooth surface to a triangulated set of data. This method is a generalization of an earlier functional side-vertex interpolant [13]. Nielson's parametric scheme proceeds by first constructing three boundary curves, one corresponding to each edge of the input triangle. Next, three patches are created, one for each boundary/opposite-vertex pair. The interior of each patch is constructed by passing curves from points along the boundary (or "side") to the opposite vertex. Hence the name "side-vertex," as shown in Figure 2. The three patches are then blended to form the final patch (Figure 3).

All curves are constructed using a curve construction

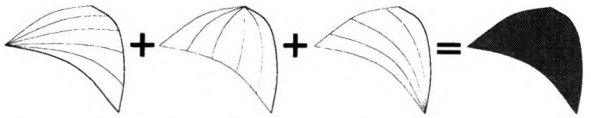

Figure 3: Nielson's surface is a blend of three side-vertex surfaces.

operator g_v that takes two vertices with normals and constructs a curve

$$g_v[V_0, V_1, N_0, N_1](t)$$

such that $g_v(0) = V_0, g_v(1) = V_1, <g_v'(0),\ N_0> = 0$, and $<g_v'(1),\ N_1> = 0$. Nielson's method also uses a normal field constructor g_n that constructs a continuous normal field along the curve g_v, where g_n is required to interpolate N_0 and N_1 at the endpoints and to be perpendicular to g_v at corresponding parameter values.

For a triangle $\triangle V_p V_q V_r$ with normals $N_p N_q N_r$, Nielson's scheme proceeds by building three patches, G_i, $ijk \in \{pqr, qrp, rpq\}$ defined as:

$$\begin{aligned} &G_i(t_p, t_q, t_r) = \\ &\quad g_v\Big[V_i,\ g_v[V_j, V_k, N_j, N_k]\left(\tfrac{t_k}{1-t_i}\right), \\ &\qquad N_i,\ g_n[V_j, V_k, N_j, N_k]\left(\tfrac{t_k}{1-t_i}\right)\Big](1-t_i), \end{aligned}$$

where t_p, t_q, t_r are the barycentric coordinates of the domain point. Nielson notes the following two properties of G_i:

1. G_i interpolates all three of the boundaries.
2. G_i interpolates the tangent plane field of the boundary opposite vertex V_i.

The three surfaces are blended with rational functions to yield the final surface:

$$G[V_p, V_q, V_r, N_p, N_q, N_r] = \beta_p G_p + \beta_q G_q + \beta_r G_r,$$

where

$$\beta_i = \frac{t_j t_k}{t_p t_q + t_q t_r + t_r t_p}. \qquad (1)$$

Nielson shows that if three surfaces having properties 1 and 2 are blended with these β_i, then the resulting surface will interpolate all the boundary curves and tangent fields. The theorem is true for a large class of g_v and g_n. Note that for any g_v and g_n, this construction has a removable singularity at the corners of the patch. Also, the mixed partial derivative at each corner is ill-defined.

The selection of g_v and g_n has a large influence on the shape of the surface. Nielson's g_v operator constructs the cubic Hermite curve that interpolates the two points V_0 and V_1, and the two derivative vectors

$$g_v'(0) = aN_0 \times (V_1 - V_0) \times N_0,$$

$$g_v'(1) = bN_1 \times (V_1 - V_0) \times N_1,$$

where a and b are scalar degrees of freedom. Selection of these a and b are a critical part of obtaining good shape. Unfortunately, there are no good criteria for setting these degrees of freedom.

For the comparisons in this paper, I used an alternative g_v operator for the representative Nielson scheme. This alternative g_v constructs the plane containing the edge V_1V_2 and the vector $N_0 + N_1$. Let N be the unit vector perpendicular to this plane. My alternative g_v operator then constructs a cubic Bézier curve with the following control points:

$$\begin{aligned} \overline{V}_0 &= V_0 \\ \overline{V}_1 &= V_0 + \frac{|V_1 - V_0|}{3} \cdot \frac{N \times N_0}{|N \times N_0|} \\ \overline{V}_2 &= V_1 - \frac{|V_1 - V_0|}{3} \cdot \frac{N \times N_1}{|N \times N_1|} \\ \overline{V}_3 &= V_1 \end{aligned}$$

This is roughly equivalent to setting Nielson's a and b shape parameters so that the length of $g_v'(0)$ and $g_v'(1)$ are $|V_1 - V_0|$. However, the alternative method constructs a planar curve, which yields surfaces with slightly better shape than Nielson's original method.

In a later paper, Hamann, Farin, and Nielson used generalized conics as the g_v function [5]. In my tests I found that the g_v operator in the previous paragraph usually gives better results than generalized conics. For example, on the data sets used in this paper (two tori and a cat data set), the generalized conics gave slightly better results for the outside portions of the torus, but the g_v operator above gives better results for the insides of the torus and for the cat data.

Nielson used the following g_n:

$$\begin{aligned} &g_n(V_0, N_0, V_1, N_1) = \\ &\quad g_v'(t) \times \Big[(1-t)N_0 \times g_v'(0) \times tN_1 \times g_v'(1)\Big]. \end{aligned}$$

Again, because it gives slightly better results, I used the following variation of g_n for my comparisons:

$$\begin{aligned} &g_n(V_0, N_0, V_1, N_1) = \\ &\quad g_v'(t) \times \Big[(1-t)N_0 + tN_1\Big] \times g_v'(t). \end{aligned} \qquad (2)$$

While use of these operators yields a G^1 surface, the resulting surfaces have poor shape (left column of Figures 7, 8, 9). In the next section, I will present the method

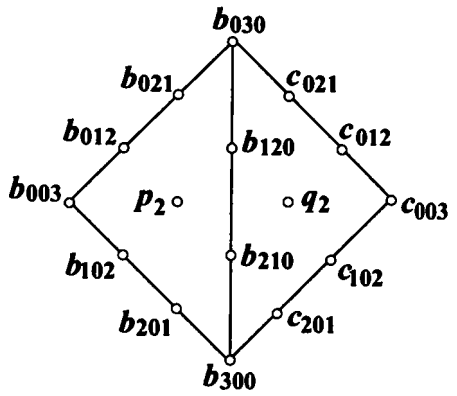

Figure 4: Domain control net for the Foley-Opitz hybrid Bézier patch

of Foley-Opitz; the main result of this paper is to use the Foley-Optiz techniques for new g_v and g_n operators that result in surfaces with better shape.

The Foley-Opitz Scheme

Foley and Opitz [3] developed a method for interpolation of scattered data above the plane using a "hybrid" cubic Bézier patch. A *hybrid cubic patch* is similar to a cubic Bézier patch, except the interior control point is a rational blend of three points. In the functional setting, the cubic patch boundaries are completely determined by the triangle vertices and normals. Foley and Opitz then construct three inner control points using a C^1 cross-boundary construction that gives the hybrid patch cubic precision.

Figure 4 shows the domain control net for two neighboring triangles. Since this is the functional setting, we need only find the z-coordinate of each control point. All of the boundary z-values are set to meet the interpolation requirements (i.e., each patch must interpolate three points and normals). The Foley-Opitz method then constructs three interior control points for each patch. In Figure 4, p_2 is one of the three interior control points associated with the left triangle and q_2 is one of the three interior control points associated with the right triangle.

Foley and Opitz compute p_2 as follows. Let r, s, and t be the barycentric coordinates of c_{003} with respect to b_{003}, b_{030}, and b_{003}. If both patches of Figure 4 form a single cubic, then from subdividing Bézier cubics it can be shown that the z-coordinate of p_2 is

$$\begin{aligned} p_2 = \; (\; & c_{102} + c_{012} - r^2(b_{300} + b_{210}) \\ & -2rs(b_{210} + b_{120}) - 2rtb_{201} - 2stb_{021} \\ & -s^2(b_{120} + b_{030}) - t^2(b_{102} + b_{012}) \\ ) \; & /(2(r+s)t). \end{aligned}$$

where only the z-coordinate of the control points is used in the calculation.

The z-coordinate of q_2 is forced by continuity conditions to be

$$q_2 = rp_2 + sb_{120} + tb_{210}.$$

When applied to data that does not come from a cubic, the Foley-Opitz construction of p_2 and q_2 ensures that the two triangles have a C^1 join along their common border. Identical calculations would be made to ensure C^1 continuity across the remaining two edges, giving three settings for the interior control points of each of the two patches.

The three interior points (p_0, p_1, p_2) are blended with Nielson's rational blend functions (Equation 1) giving

$$\begin{aligned} b_{111}(t_0,t_1,t_2) \; = \; & \beta_0(t_0,t_1,t_2)p_0 + \beta_1(t_0,t_1,t_2)p_1 \\ & +\beta_2(t_0,t_1,t_2)p_2. \end{aligned}$$

After blending, we are left with a 10 point cubic Bézier patch, which is evaluated at (t_0, t_1, t_2) in the standard way.

In this paper, our interest in the Foley-Opitz scheme is that surfaces created by their scheme have noticeably better shape than those produced by schemes such as the Clough-Tocher technique [3, 7]. The next section will show one way to incorporate the Foley-Opitz method into a parametric scheme, yielding similar improvements in shape.

A Modified Nielson Approach

Davidchuk [1, 10] incorporated the Foley-Opitz cross-boundary construction into a hybrid parametric scheme. The resulting surfaces showed significant improvement in shape. In this paper, I will incorporate the Foley-Opitz method into Nielson's scheme. This is done by changing the two Nielson operators g_v and g_n.

The Foley-Opitz cross-boundary construction relies on a natural parameterization between patch pairs, since the barycentric coordinates of neighboring patches with respect to each other are key in determining the tangent plane fields. In the parametric setting, there is no predefined association between patch domains. Therefore, to use Foley's tangent plane field construction, we must make an association between neighboring patch domains.

I chose to use Davidchuk's choice of a plane, and perform the construction over the plane defined by the normal $N = N_0 + N_1$ passing through the point V_0 (the choice of V_0 is arbitrary; any point may be chosen, as the Foley-Opitz construction is independent of this point). Given this plane, the new curve constructor builds a cubic Bézier curve $\overline{V}_0, \overline{V}_1, \overline{V}_2, \overline{V}_3$ as follows (Figure 5):

1. Set $\overline{V}_0 = V_0$ and $\overline{V}_3 = V_1$.

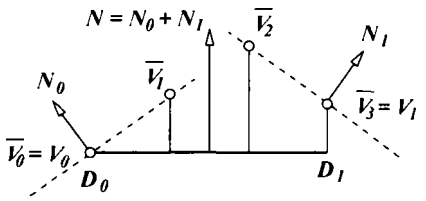

Figure 5: Functional g_v operator.

2. Project V_0 and V_1 into the plane given by V_0 and N, giving the points D_0 and D_1.

   The straight line connecting D_0 and D_1 forms the domain for the curve.

3. (a) Intersect the line given by $(2D_0 + D_1)/3$ and N with the plane given by V_0, N_0 to get the point $\overline{V}_1$.

   (b) Similarly, the line given by $(D_0 + 2D_1)/3$ and N is intersected with the plane given by V_1, N_1 to get the point $\overline{V}_2$.

For the normal operator g_n, I use the Foley-Opitz construction when possible. Since their construction only works for functional data, I reparameterize the data to be relative to the plane given in my g_v construction. From the Foley-Opitz construction, I keep two layers of control points. The first layer contains the points b_{030}, b_{120}, b_{210}, b_{300} (which are the same points that g_v constructs) and the second layer contains the points b_{021}, p_2, and b_{201}. I then use these two layers to compute the normal for g_n.

Problems and Solutions

On implementing and testing the scheme described in the previous section, I encountered several problems. Although in most areas, surfaces constructed using the above g_v, g_n operators have better shape than surfaces constructed using the original Nielson operators, for certain data, the new operators will construct surfaces of extremely poor surface quality. Fortunately, this problem data is easily detected, and the Nielson side-vertex construction is flexible enough to allow us to use different operators on different parts of the surface. The following is a list of problems with the new operators, and a discussion of how to detect and correct these problems.

- One problem with the new g_v operator is that in the limit, as a normal becomes parallel to an edge, the intermediate control points move to infinity. To avoid this problem, I modified the new method to check the dot product of the normal with an edge, and if the dot product is greater than a threshold value, I adjust the position of the corresponding interior control point to shorten the length of the first derivative. More precisely, at step (3a) of the above construction, if $|N_0 \cdot (V_0 - V_1)|/|V_0 - V_1| > C_m$, then in place of $\overline{V}_1$, g_v constructs the point $\hat{V}_1$ as follows:

$$\hat{V}_1 = V_0 + \frac{\overline{V}_0 - V_0}{|\overline{V}_0 - V_0|} \cdot \frac{|D_1 - D_0|}{3\sqrt{1 - C_m^2}} \qquad (3)$$

  Here, C_m is the maximum cosine (of the angle between N_0 and the edge V_0V_1) for which we will use the Foley-Opitz construction. If the cosine of this angle is greater than C_m, then we construct the tangent vector to have the same length as one the one that would be constructed if the angle was exactly C_m. A similar change is made at step (3b). For the images in this paper, I used a value of $C_m = 0.8$.

- A second problem is that the normal $N = N_0 + N_1$ may prove inadequate, since the neighboring triangles may overlap one-another in this plane. Likewise, if one of the one of the neighboring triangles is nearly perpendicular to N, then the Foley-Opitz cross-boundary construction has numerical problems, and produces interior control points that are far from the data triangle (which results in unacceptable bumps in the surface). A simple solution to this problem is to detect it by checking the dot product of N with each triangle normal, and use the modified Nielson g_n operator (Equation 2) for edges where either dot product is too small (I used a minimum of 0.2). Alternatively, we could find a different plane into which the triangle have a projection, and use the g_v and g_n operators over this plane.

- A further problem with the g_n operator occurs for a face that lies on the boundary of a mesh, since such a face does not have neighbors across some of its edges. In such cases, while the g_v operator still works, a different g_n operator must be used. Also, if one of the interior control points of a boundary curve is computed using Equation 3, then that boundary does not lie above a plane, and the Foley-Opitz cross-boundary construction can not be used. For both these boundary edges and these non-functional, I use Equation 2 for g_n.

Results

To test my scheme, I fit surfaces to two samplings of a torus, and to a cat data set (Figure 6). For the torus data sets, the normals were sampled directly from the torus. For the cat data set, the normal at each vertex was estimated to be the average of the unit face normals for all faces surrounding the vertex.

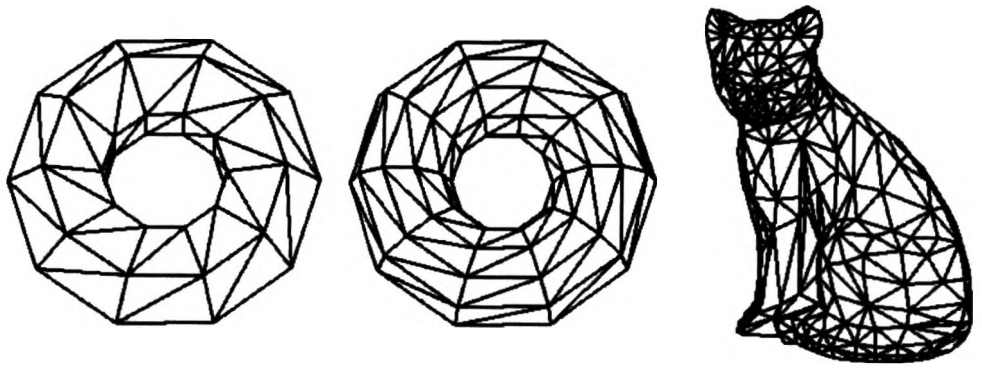

Figure 6: Data sets

Note that normal estimation is a difficult problem. For the cat data set, the normal estimation scheme used in this paper was chosen for its simplicity. However, this method does not produce particularly good normals. For an example of a more sophisticated normal estimation scheme, see Nielson's paper [15].

I have compared my new scheme against several local schemes; however, none of these other local schemes performed significantly better than (and several constructed worse surfaces than) Nielson's original scheme. Thus, in this paper, I make comparisons to Nielson's original scheme, and consider it as representative of local, parametric triangular schemes.

In Figures 7, 8, 9, on the left is the surface constructed by Nielson's scheme; in the middle is the surface constructed using the new $\boldsymbol{g_v}$ operator; and on the right is the surface constructed using both the new $\boldsymbol{g_v}$ operator and the $\boldsymbol{g_n}$ operator. The middle images indicate that there is little change in the surface quality when you use the new $\boldsymbol{g_v}$ operator without the new $\boldsymbol{g_n}$ operator.

In Figures 7 and 8, I have plotted isophotes [4] atop the surfaces, as the shading artifacts are somewhat subtle (a discontinuity in the isophote indicates a wrinkle in the surface). In Figure 7, we see that the new scheme constructs a worse surface than the original scheme. However, as we increase the sampling of the torus, we see in Figure 8 that the new scheme constructs a higher quality surface. This rapid improvement as we increase the sampling of the torus suggests that the new scheme has a higher order of convergence than the original Nielson scheme.

In Figure 9, we see that the new scheme performs significantly better on parametric scattered data than the original side-vertex method; many of the wrinkles in the surface have disappeared (e.g., on the tail and back of the cat).

Note that basic construction (described in the section **A Modified Nielson Approach**) was used for most parts of these surfaces (both tori and the cat); the only spots in which the alternative boundaries and cross-boundaries (described in the section **Problems and Solutions**) were used were on the boundaries of the cat and on the cat's ears.

As a final comment, tests with other data sets indicate that if the triangulation of the data is "poor," then the resulting surface will strongly reflect this poor triangulate. This appears to be a problem inherent in Nielson's scheme, as it occurs with all sets of $\boldsymbol{g_v}/\boldsymbol{g_n}$ operators used in this paper. While some improvements can by achieved by modifying the triangulation, there will be data for which neither my or the original Nielson scheme will be able to construct good surfaces.

Analysis and Conclusions

Methods for fitting G^1 surfaces to triangulated, parametric data have existed for many years. The early schemes uniformly constructed surfaces that had severe shape defects. The flaws in these early schemes were due to poor settings of degrees of freedom. These degrees of freedom occur in the boundary curve constructions, in the cross-boundary construction, and elsewhere in the patch construction. While using good settings of the boundary curve degrees of freedom is a necessary condition for obtaining good shape, it is not a sufficient condition – one must also find good settings for the other degrees of freedom in the construction.

Other researchers have used variational techniques to set extra degrees of freedom in surface construction methods. In this paper, I have shown that the extra degrees of freedom can be set using local, geometric constructions based on functional constructions that have high degree polynomial precision. While the surfaces produced by my scheme are not as good as those produced by variational techniques, my scheme is an improvement over

other local parametric schemes, the computational cost of my technique is lower than that of variational methods, and the ideas in this paper should prove useful to finding good starting points for variational techniques.

Acknowledgments

Many thanks to Angelika Engelmann who implemented the g_v operator and developed the basic code structure.

This research was funded by the Natural Sciences and Engineering Research Council of Canada.

References

[1] Matthew Davidchuk. A parametric hybrid triangular bézier patch. Master's thesis, University of Waterloo, 1997. ftp://cs-archive.uwaterloo.ca/cs-archive/CS-97-29/.

[2] Gerald Farin. *Curves and Surfaces for CAGD*. Academic Press, 1993.

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

[4] Hans Hagen, Stephanie Hahmann, Thomas Schreiber, Yasuo Nakajima, Burkard Wördenweber, and Petra Hollemann-Grundetedt. Surface interrogation algorithms. *Computer Graphics and Applications*, 12:53–60, 1992.

[5] Bernd Hamann, Gerald Farin, and Gregory Nielson. A parameteric triangular patch based on generalized conics. In G. Farin, editor, *NURBS for Curve and Surface Design*, pages 75–85. SIAM, 1991.

[6] Michael Lounsbery, Stephen Mann, and Tony DeRose. Parameteric surface interpolation. *Computer Graphics and Applications*, pages 45–52, 1992.

[7] Stephen Mann. Cubic precision clough-tocher interpolation. submitted for publication.

[8] Stephen Mann. *Surface Approximation Using Geometric Hermite Patches*. PhD thesis, University of Washington, 1992. ftp://ftp.cs.washington.edu/tr/1992/11/UW-CSE-92-11-06.tar.

[9] Stephen Mann. Using local optimization in surface fitting. In M. Dæhlen, T. Lyche, and L. Schumaker, editors, *Mathematical Methods for Curves and Surfaces*, pages 323–332. Vanderbilt University Press, 1995.

[10] Stephen Mann and Matthew Davidchuk. A parametric hybrid triangular bézier patch. to appear in Mathematical Methods for Curves and Surfaces II.

[11] Stephen Mann, Charles Loop, Michael Lounsbery, David Meyers, James Painter, Tony DeRose, and Kenneth Sloan. A survey of parametetric scattered data fitting using triangular interpolants. In H. Hagen, editor, *Curve and Surface Design*, pages 145–172. SIAM, 1992.

[12] Henry Moreton and Carlo Séquin. Functional optimization for fair surface design. In *SIGGRAPH '92*, pages 167–176, 1992.

[13] Gregory Nielson. The side-vertex method for interpolation in triangles. *Journal of Approximation Theory*, 25:318–336, 1979.

[14] Gregory Nielson. A transfinite, visually continuous, triangular interpolant. In G. Farin, editor, *Geometric Modeling: Algorithms and New Trends*, pages 235–245. SIAM, 1987.

[15] Gregory Nielson. Iterative surface design using triangular network splines. In S. Slaby and H. Stachel, editors, *Third International Conference on Engineering Graphics and Descriptive Geometry*, volume 2, pages 70–77, 1988.

Figure 7: Surfaces fit to a 10x5 sampling of a torus; Left: Nielson's scheme; Middle: Nielson's scheme using the new g_v operator; Right: Nielson's scheme using the new g_v and g_n operators.

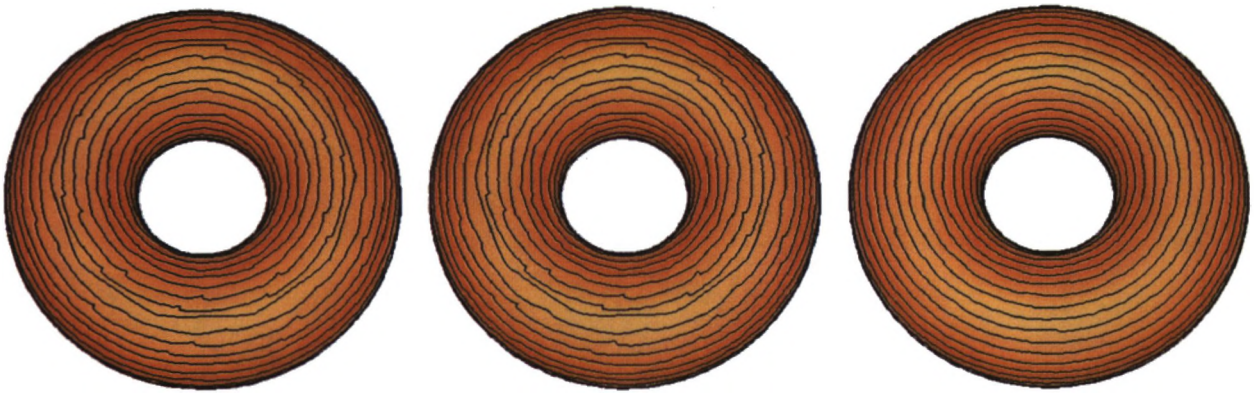

Figure 8: Isophote plots of surfaces fit to a 10x10 sampling of a torus.

Figure 9: Nielson cat, Nielson cat with Foley-Opitz boundaries, Nielson cat with Foley-Opitz cross-boundaries.