Smooth Image Surface Approximation by Piecewise Cubic Polynomials

Oliver Matias van Kaick[1] and Helio Pedrini[2]

[1] School of Computing Science,
Simon Fraser University, Burnaby, BC, Canada, V5A 1S6
[2] Department of Computer Science
Federal University of Paraná, Curitiba, PR, Brazil, 81531-990

Abstract. The construction of surfaces from dense data points is an important problem encountered in several applications, such as computer vision, reverse engineering, computer graphics, terrain modeling, and robotics. Moreover, the particular problem of approximating digital images from a set of selected points allows to employ methods that are directed specifically to this task, which take advantage of the fact that all points belong to a common 2D domain. This paper describes a method for approximating images by fitting smooth surfaces to scattered points, where the smooth surfaces are constructed using piecewise cubic approximation. An incremental triangulation algorithm is used to iteratively refine a mesh until a specified error tolerance is achieved. The resulting surface is represented by a network of piecewise cubic triangular patches possessing C^1 continuity. The proposed method is compared against other surface approximation approaches and applied to several data sets in order to demonstrate its performance.

Keywords: Surface modeling, image surface, smooth interpolation.

1 Introduction

Approximation from unorganized data points arbitrarily distributed over 2D or 3D domains is a crucial problem in several scientific and industrial applications. Recent advances in acquisition of high-resolution images, associated with efficient modeling techniques, have allowed the construction of models with high degree of detail. Unless data reduction or compression methods are used, dense data sets cannot be stored, manipulated, or visualized efficiently.

Polygonal surfaces are often used to represent 3D data sets mainly because of their simplicity and flexibility. In the last few years, several polygonal surface algorithms have been proposed in the literature for generating a surface containing a small number of polygons. This is important for processing, visualizing, or transmitting larger surface data sets than the available capabilities of software, computers, and networks permit.

Although piecewise linear approximation approaches are simple in concept and generate compact surfaces, the generalization to a piecewise smooth representation is a natural and, in many cases, a necessary extension. Certain regions

L. Rueda, D. Mery, and J. Kittler (Eds.): CIARP 2007, LNCS 4756, pp. 261–270, 2007.

of interest may consist of smoothly curved areas that meet along sharp curves. Modeling such regions as piecewise linear surfaces usually requires a large number of polygons, whereas a curved surface can provide a more compact and accurate model of the surface. Smooth surfaces can also produce superior results for rendering purposes, reducing certain perceptual problems, for instance the appearance of Mach bands along element boundaries [1].

This paper describes a method for approximating digital images or other datasets constrained to a 2D domain, such as digital elevation models. The approximation is constructed by fitting a smooth surface to points selected from the dataset. An incremental triangulation algorithm is used to iteratively refine the mesh that describes the surface until a specified error tolerance is achieved. The approximating surface is represented by piecewise cubic triangular patches possessing C^1 continuity. Two strategies for the refinement method are investigated. The first creates the triangular patches only after the iterative triangulation algorithm is executed, and the second strategy already builds the mesh according to the error generated by the cubic patches. The proposed method is applied to several synthetic and natural data sets to demonstrate its robustness.

Section 2 briefly presents a review of some relevant surface fitting methods found in the literature. Section 3 describes the proposed method. Experimental results are given in Section 4. Some conclusions are summarized in Section 5.

2 Related Work

Techniques for piecewise linear approximation from data points have been proposed by several researchers. The resulting surface can be generally obtained by either refining a coarse triangulation or simplifying a fine triangular mesh until a given tolerance error is achieved.

As an alternative to planar polygonal models, smooth surfaces can be approximated more accurately with higher-order polynomials. Several adaptive methods have been developed for generating piecewise polynomial elements, such as subdivision surfaces [2,3,4,5], hierarchical splines [6], and models composed of triangular patches [1,7,8].

In 1974, Chaikin [9] introduced a method for generating a smooth curve from a control polygon by recursively cutting off the corners of the polygon. This is perhaps the first method of constructing smooth curves of arbitrary topological type. Catmull and Clark [2], and Doo and Sabin [10] generalized the idea to surfaces, where a subdivision surface is defined by repeatedly refining an initial control mesh to produce a sequence of meshes that converge to a limit surface.

The Loop scheme [4] is probably the simplest subdivision method for triangular meshes. Each edge of the mesh is split into two, and new vertices are reconnected to form four new triangles. Vertices are rearranged through an averaging step.

The Butterfly subdivision scheme, proposed by Dyn *et al.* [3], recursively subdivides each triangular face of the control polygon into four triangular faces interpolating the old control points. The subdivision step retains the existing vertices and splits each edge segment at its midpoint. The Butterfly scheme is proven to achieve tangent plane continuity when applied to regular meshes. A variant of the Butterfly scheme was proposed by Zorin [11], which guarantees that the subdivision produces C^1-continuous surfaces for arbitrary meshes.

A set of spline patches can be derived to fit a smooth surface over irregular polygonal meshes, globally achieving some order of continuity [12,13,14,15,16].

3 Proposed Method

The method proposed for fitting surfaces to scattered points involves two distinct parts, which are the surface approximation method, that creates a triangle mesh from the provided set of points, and the interpolation method, which defines the error between the original data and the triangles. These parts are described in the next two sections.

3.1 Surface Approximation

A refinement method is used in order to generate a piecewise approximation of a certain surface. The algorithm starts with an initial triangulation that covers the boundary of the domain, and iteratively adds new points from the data set until a specified error tolerance is achieved. The resulting surface is formed by C^0 continuous triangular patches.

The Delaunay triangulation is used to construct the mesh, generating the triangulation that maximizes the minimum angle of all triangles. This helps to reduce the occurrence of thin and long triangles since they can lead to undesirable behavior, affecting numerical stability and producing visual artifacts. During the approximation method, the error for each triangle is computed as the sum of the squared difference between each original data point inside the domain of the triangle and its interpolated value. Each triangle has also an associated candidate point, which is the point of the triangle with the maximum difference. Figure 1 presents an example where the points involved in the computation of the triangle error are shown. The candidate point is highlighted in black.

The four steps of the refinement method can be seen in Figure 2, from (a) to (d). At each iteration of the method, for a given mesh (a), the candidate point

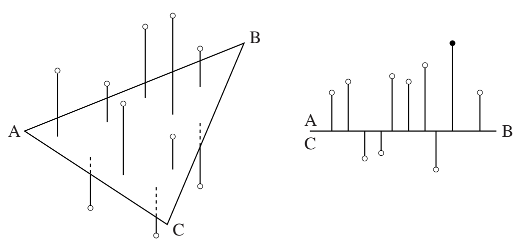


Fig. 1. Vertical error

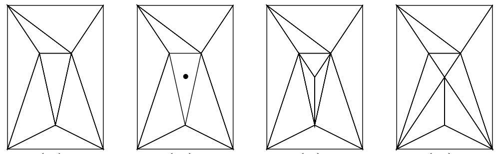


Fig. 2. Surface approximation method

of the triangle with maximum error is inserted into the mesh (b). Three new triangles are created with this vertex (c), and the modified portion of the mesh is retriangulated in order that the triangles maintain the Delaunay property (d).

The error evaluated for a given point clearly depends on the interpolation scheme that is being used. If a linear scheme is used, the value of a point is computed according to the plane defined by the three points of its containing triangle. In this work, we propose a surface approximation method using cubic interpolation, and compare it with linear interpolation. The smooth interpolation can be used either during the iterative construction of the mesh or only to fit a polynomial surface over an already created mesh. Our cubic interpolation method is described in the next section.

3.2 Smooth Surface Interpolation

Our interpolation method uses cubic polynomials to construct C^1 surfaces, which is based on a scheme originally described by Clough and Tocher [17]. It divides each triangle into subtriangles and fits an approximating function over each subtriangle. Certain continuity conditions must be satisfied at every boundary between two adjacent patches in order for the entire surface to be smooth.

Before describing the interpolation method, some preliminary concepts are introduced.

Mathematical Preliminaries

The use of *barycentric coordinates* is a natural way of representing triangular patches, since this guarantees a symmetric influence of all three triangle corners. Let T be a planar triangle defined by the vertices V_1, V_2, V_3. Any point V in T can be expressed in terms of the barycentric coordinates (r, s, t) defined by $V = rV_1 + sV_2 + tV_3$, where $r + s + t = 1$ and $0 \leqslant r, s, t \leqslant 1$.

Bernstein polynomials of degree n over a triangle T can be defined in terms of barycentric coordinates (r, s, t) expressed as

$$B^n_{i,j,k}(r, s, t) = \frac{n!}{i!\, j!\, k!}\, r^i s^j t^k \tag{1}$$

which form a basis for all bivariate polynomials of degree n. The parametric equation for a single triangular Bernstein-Bézier patch is

$$\mathbf{p}(r,s,t) = \sum_{\substack{i+j+k=n \\ i,j,k \geqslant 0}} \mathbf{b}_{i,j,k}\ B^n_{i,j,k}(r,s,t) \tag{2}$$

where the coefficients $\mathbf{b}_{i,j,k}$ are called the Bézier control points of $\mathbf{p}(r,s,t)$.

Taking $n = 3$, Equation 2 gives

$$\begin{aligned}\mathbf{p}(r,s,t) = r^3\ \mathbf{b}_{3,0,0} + s^3\ \mathbf{b}_{0,3,0} + t^3\ \mathbf{b}_{0,0,3} + 3r^2s\ \mathbf{b}_{2,1,0} + 3r^2t\ \mathbf{b}_{2,0,1} + \\ 3rs^2\ \mathbf{b}_{1,2,0} + 3s^2t\ \mathbf{b}_{0,2,1} + 3rt^2\ \mathbf{b}_{1,0,2} + 3st^2\ \mathbf{b}_{0,1,2} + 6rst\ \mathbf{b}_{1,1,1}\end{aligned} \tag{3}$$

where (r,s,t) are the barycentric coordinates of a point (x,y) relative to the subtriangle. Figure 3 shows an example of a triangular patch and its corresponding Bernstein polynomials.

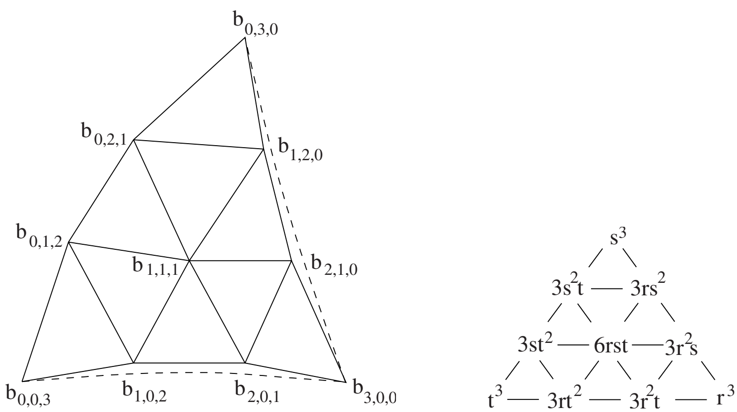


Fig. 3. Cubic triangular Bernstein-Bézier patch and its corresponding Bernstein polynomials

Cubic Interpolant

The Clough-Tocher [17] interpolation scheme, originally developed as a technique in finite element analysis, was used to produce a piecewise cubic polynomial surface during the approximation method. For the surface construction, each triangle is subdivided at the centroid into three subtriangles, and a cubic Bernstein-Bézier polynomial is defined over each subtriangle. Figure 4 illustrates the Clough-Tocher interpolation scheme.

Farin [7] provides a comprehensive description of the conditions for derivative continuity on the common boundary between two adjacent triangular patches. To ensure C^1 continuity, the first derivatives of two adjacent patches $\mathbf{p}$ and $\mathbf{q}$ must join continuously across the shared edge. Our interpolation method is based on the work developed by Quak and Schumaker [18]. Their paper provides a construction such that derivative continuity is achieved on each shared triangle edge.

Our interpolation method is relatively simple to implement since it computes the coefficients of the polynomial for each triangle based only on the elevation values and the estimated values of the first partial derivatives (tangent vectors)

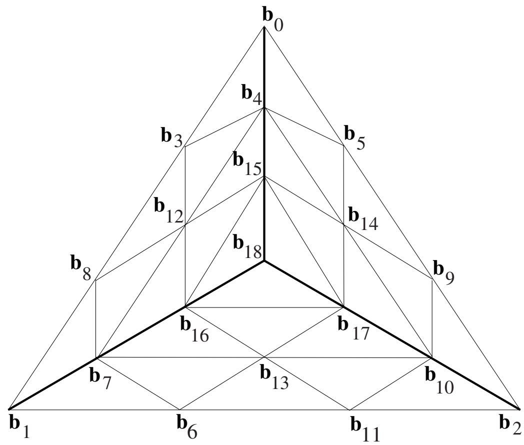


Fig. 4. Clough-Tocher subdivision

at the three vertices of the triangle. The derivative at a vertex is computed as the weighted average of the normals of the triangles adjacent to that vertex.

The 19 Bernstein-Bézier coefficients of the Clough-Tocher interpolation scheme are given by the following equations

$$\begin{aligned}
\mathbf{b}_0 &= z_0 \\
\mathbf{b}_1 &= z_1 \\
\mathbf{b}_2 &= z_2 \\
\mathbf{b}_3 &= ((x_1 - x_0)z_0^x + (y_1 - y_0)z_0^y)/3 + z_0 \\
\mathbf{b}_4 &= ((x_c - x_0)z_0^x + (y_c - y_0)z_0^y)/3 + z_0 \\
\mathbf{b}_5 &= ((x_2 - x_0)z_0^x + (y_2 - y_0)z_0^y)/3 + z_0 \\
\mathbf{b}_6 &= ((x_2 - x_1)z_1^x + (y_2 - y_1)z_1^y)/3 + z_1 \\
\mathbf{b}_7 &= ((x_c - x_1)z_1^x + (y_c - y_1)z_1^y)/3 + z_1 \\
\mathbf{b}_8 &= ((x_0 - x_1)z_1^x + (y_0 - y_1)z_1^y)/3 + z_1 \\
\mathbf{b}_9 &= ((x_0 - x_2)z_2^x + (y_0 - y_2)z_2^y)/3 + z_2 \\
\mathbf{b}_{10} &= ((x_c - x_2)z_2^x + (y_c - y_2)z_2^y)/3 + z_2 \\
\mathbf{b}_{11} &= ((x_1 - x_2)z_2^x + (y_1 - y_2)z_2^y)/3 + z_2 \\
\mathbf{b}_{12} &= (\mathbf{b}_4 + \mathbf{b}_7 + (\theta_0 - 1)\mathbf{b}_0 + \quad (2 - 3\theta_0)\mathbf{b}_3 + (3\theta_0 - 1)\mathbf{b}_8 - \theta_0\mathbf{b}_1)/2 \\
\mathbf{b}_{13} &= (\mathbf{b}_7 + \mathbf{b}_{10} + (\theta_1 - 1)\mathbf{b}_1 + \quad (2 - 3\theta_1)\mathbf{b}_6 + (3\theta_1 - 1)\mathbf{b}_{11} - \theta_1\mathbf{b}_2)/2 \\
\mathbf{b}_{14} &= (\mathbf{b}_{10} + \mathbf{b}_4 + (\theta_2 - 1)\mathbf{b}_2 + \quad (2 - 3\theta_2)\mathbf{b}_9 + (3\theta_2 - 1)\mathbf{b}_5 - \theta_2\mathbf{b}_0)/2 \\
\mathbf{b}_{15} &= (\mathbf{b}_{14} + \mathbf{b}_4 + \mathbf{b}_{12})/3 \\
\mathbf{b}_{16} &= (\mathbf{b}_{12} + \mathbf{b}_7 + \mathbf{b}_{13})/3 \\
\mathbf{b}_{17} &= (\mathbf{b}_{13} + \mathbf{b}_{10} + \mathbf{b}_{14})/3 \\
\mathbf{b}_{18} &= (\mathbf{b}_{17} + \mathbf{b}_{15} + \mathbf{b}_{16})/3
\end{aligned} \tag{4}$$

where

$$\begin{aligned}\theta_0 &= \frac{(x_c - x_0)(x_1 - x_0) + (y_c - y_0)(y_1 - y_0)}{(x_1 - x_0)^2 + (y_1 - y_0)^2} \\ \theta_1 &= \frac{(x_c - x_1)(x_2 - x_1) + (y_c - y_1)(y_2 - y_1)}{(x_2 - x_1)^2 + (y_2 - y_1)^2} \\ \theta_2 &= \frac{(x_c - x_2)(x_0 - x_2) + (y_c - y_2)(y_0 - y_2)}{(x_0 - x_2)^2 + (y_0 - y_2)^2}\end{aligned} \tag{5}$$

The coordinates of points $\mathbf{b}_0$, $\mathbf{b}_1$, and $\mathbf{b}_2$ are (x_0, y_0, z_0), (x_1, y_1, z_1) and (x_2, y_2, z_2), respectively, and the derivatives at these points are (z_0^x, z_0^y), (z_1^x, z_1^y) and (z_2^x, z_2^y). The point (x_c, y_c) is the centroid of the triangle ($\mathbf{b}_0$, $\mathbf{b}_1$, $\mathbf{b}_2$).

Initially, the derivatives at each data vertex are computed by estimating the normals for each adjacent triangle. Then, three cubic triangular Bézier patches are constructed over each subtriangle. A cubic Bézier patch is defined by 10 control points as shown in Figure 3. The 10 control points of each subtriangle provide the degrees of freedom required to ensure continuity across the element boundaries.

The Bernstein-Bézier representation for subtriangle $(\mathbf{b}_0, \mathbf{b}_1, \mathbf{b}_{18})$ is

$$\begin{aligned}\mathbf{p}(r, s, t) = r^3\, \mathbf{b}_{18} + s^3\, \mathbf{b}_0 + t^3\, \mathbf{b}_1 + 3r^2 s\, \mathbf{b}_{15} + 3r^2 t\, \mathbf{b}_{16} + \\ 3rs^2\, \mathbf{b}_4 + 3s^2 t\, \mathbf{b}_3 + 3rt^2\, \mathbf{b}_7 + 3st^2\, \mathbf{b}_8 + 6rst\, \mathbf{b}_{12}\end{aligned} \tag{6}$$

The C^1 cubic surfaces for subtriangles $(\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_{18})$ and $(\mathbf{b}_0, \mathbf{b}_2, \mathbf{b}_{18})$ are computed analogously.

4 Experimental Results

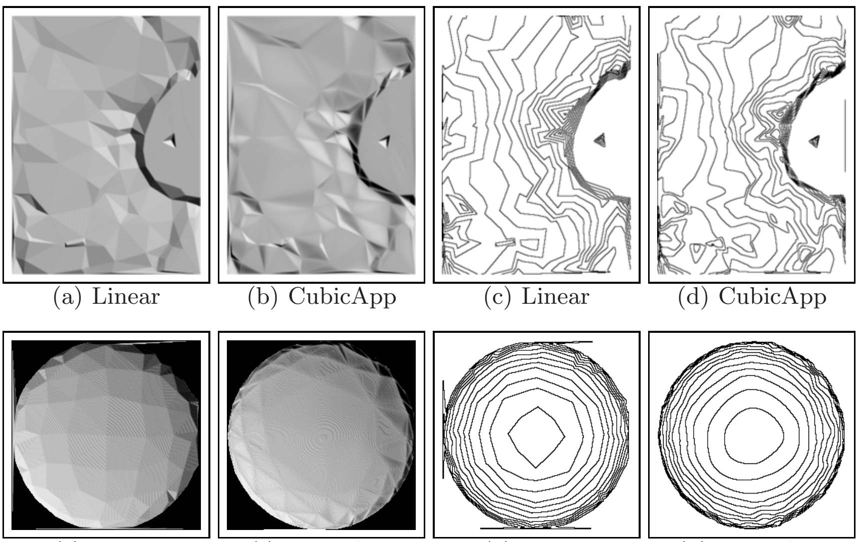
We compare four surface approximation methods. For the first three methods, a triangle mesh is firstly created according to linear interpolation, and the error of this approximation is compared when the resulting mesh is interpolated using the linear method, our previous quadratic method [1], and the proposed cubic scheme. For the fourth method (labeled *CubicApp*), the described cubic interpolant is directly used during the iterative construction of the triangle mesh, and the error for the triangles and the candidate points is computed according to this scheme.

Our method has been tested and evaluated on several different real and synthetic digital images in order to demonstrate its performance, however, due to limited space, the visual results for only three models are presented in this work. The algorithms were implemented in C programming language on a PC Athlon XP 2000 MHz with 512 Mbytes of main memory.

Table 1 reports the root mean square (RMS) error for a set of reconstructed objects. Figures 5 and 6 show three sets of images obtained by applying the linear and cubic interpolation methods, and their related contour lines. The same number of data points are retained in both methods.

Table 1. Summary of results (RMS Error) for reconstructed objects using 1% of the original points

Model	Linear	Quadratic	Cubic	CubicApp
Columbia	6.84	7.76	7.45	7.43
Crater Lake	5.52	6.93	6.63	6.40
Emory Peak	13.90	16.75	15.62	15.58
Grand Canyon	16.37	19.15	18.52	18.20
Klamath Falls	8.07	10.73	9.59	9.37
Mars	4.91	7.55	7.12	6.83
Peppers	11.22	12.68	12.64	12.53
Rice Lake	1.75	2.16	2.09	2.03

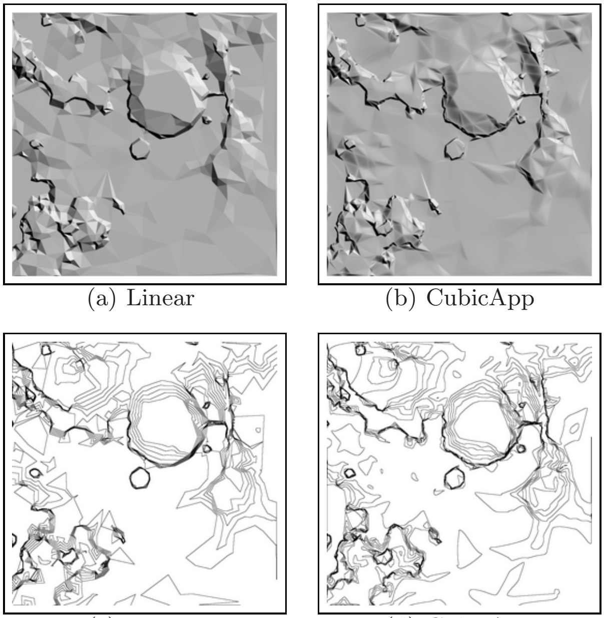


(a) Linear (b) CubicApp (c) Linear (d) CubicApp

(e) Linear (f) CubicApp (g) Linear (h) CubicApp

Fig. 5. Approximations of two data sets by the linear and cubic method, respectively. (a) and (b) Crater Lake DEM (336×459 pixels); (c) and (d) Contour lines; (e) and (f) Half-Sphere DEM (256×256 pixels); (g) and (h) Contour lines.

Although the linear method presented lower errors than the smooth interpolation schemes, the cubic interpolation used during the iterative construction of the triangulation resulted in high quality meshes, when compared to the methods that only fit a smooth surface over an already created linear mesh with either the quadratic or cubic scheme. Thus, it appears as a better alternative when lower error approximations with properties such as C^1 continuity and smooth contour lines are required.



(a) Linear (b) CubicApp

(c) Linear (d) CubicApp

Fig. 6. Approximations of one data set by the linear and cubic method, respectively. (a) and (b) Mars DEM (948×948 pixels); (c) and (d) Contour lines.

5 Conclusions

A method for approximating digital images by smooth surfaces is presented in this paper. The method generates a set of piecewise cubic polynomial patches possessing C^1 continuity. The approximation error that guides the point selection process during the iterative construction of the surface is evaluated according to the interpolation of the cubic polynomials. Therefore, the method selects at a given iteration the point that reduces the maximum error for the current smooth surface.

The results show that the method constructs surfaces of high quality, which are comparable in terms of RMS Error to surfaces constructed by the linear method, but that are more suitable for applications where smooth approximations for digital images or other types of datasets are necessary.

References

1. van Kaick, O.M., da Silva, M.V.G., Schwartz, W.R., Pedrini, H.: Fitting Smooth Surfaces to Scattered 3D Data Using Piecewise Quadratic Approximation. In: IEEE International Conference on Image Processing, Rochester, New York, USA, pp. 493–496. IEEE Computer Society Press, Los Alamitos (2002)

2. Catmull, E., Clark, J.: Recursively generated B-spline surfaces on arbitrary topological meshes. Computer-Aided Design 10(6), 350–355 (1978)
3. Dyn, N., Levin, D., Gregory, J.A.: A Butterfly subdivision scheme for surface interpolation with tension control. ACM Transactions on Graphics 9(2), 160–169 (1990)
4. Loop, C.: Smooth subdivision surfaces based on triangles. Master's thesis, University of Utah, Department of Mathematics (1987)
5. Stam, J.: On subdivision schemes generalizing uniform B-spline surfaces of arbitrary degree. Computer Aided Geometric Design 18, 383–396 (2001)
6. Forsey, D.R., Bartels, R.H.: Surface fitting with hierarchical splines. ACM Transactions on Graphics 14(2), 134–161 (1995)
7. Farin, G.: Curves and Surfaces for Computer-Aided Geometric Design - A Practical Guide. Academic Press, London (1992)
8. Hahmann, S., Bonneau, G.P.: Polynomial surfaces interpolating arbitrary triangulations. IEEE Transactions on Visualization and Computer Graphics 9(1), 99–109 (2003)
9. Chaikin, G.M.: An algorithm for high-speed curve generation. Computer Graphics and Image Processing 3(4), 346–349 (1974)
10. Doo, D., Sabin, M.: Behaviour of recursive division surfaces near extraordinary points. Computer-Aided Design 10(6), 356–360 (1978)
11. Zorin, D., Schröder, P., Sweldens, W.: Interpolating subdivision for meshes with arbitrary topology. In: SIGGRAPH 1996 Conference Proceedings, New Orleans, Louisiana, USA, pp. 189–192 (August 1996)
12. Eck, M., Hoppe, H.: Automatic reconstruction of B-spline surfaces of arbitray topological type. In: ACM SIGGRAPH 1996, pp. 325–334. ACM Press, New York (1996)
13. Zheng, J.J., Zhang, J.J., Zhou, H.J., Shen, L.G.: Smooth spline surface generation over meshes of irregular topology. The Visual Computer 21(8–10), 858–864 (2005)
14. Krishnamurthy, V., Levoy, M.: Fitting smooth surfaces to dense polygonal meshes. In: Computer Graphics Proceedings, Annual Conference Series, pp. 313–324 (1996)
15. Loop, C.: Smooth splines surfaces over irregular meshes. In: Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH, pp. 303–310. ACM Press, New York (1994)
16. Peters, J.: C^1 surface splines. SIAM Journal of Numerical Analysis 32(2), 645–666 (1995)
17. Clough, R., Tocher, J.: Finite element stiffness matrices for analysis of plates in bending. In: Proceedings of Conference on Matrix Methods in Structural Analysis (1965)
18. Quak, E., Schumaker, L.L.: Cubic spline fitting using data dependent triangulations. Computer-Aided Geometric Design 7(1–4), 293–301 (1990)