Contents lists available at ScienceDirect

Computers & Graphics

Technical Section

Surface reconstruction using bivariate simplex splines on Delaunay configurations

Juan Cao [a], Xin Li [b,*], Guozhao Wang [a], Hong Qin [c]

[a] *Institute of Computer Graphics and Image Processing, Department of Mathematics, Zhejiang University, Hangzhou 310027, PR China*
[b] *Department of Electrical and Computer Engineering and Center for Computation and Technology, Louisiana State University, Baton Rouge, LA 70803, USA*
[c] *Department of Computer Science, State University of New York at Stony Brook, Stony Brook, NY 11794-4400, USA*

ARTICLE INFO

ABSTRACT

Recently, a new bivariate simplex spline scheme based on *Delaunay configuration* has been introduced into the geometric computing community, and it defines a complete spline space that retains many attractive theoretic and computational properties. In this paper, we develop a novel shape modeling framework to reconstruct a closed surface of arbitrary topology based on this new spline scheme. Our framework takes a triangulated set of points, and by solving a linear least-square problem and iteratively refining parameter domains with newly added knots, we can finally obtain a continuous spline surface satisfying the requirement of a user-specified error tolerance. Unlike existing surface reconstruction methods based on triangular B-splines (or DMS splines), in which auxiliary knots must be explicitly added in advance to form a knot sequence for construction of each basis function, our new algorithm completely avoids this less-intuitive and labor-intensive knot generating procedure. We demonstrate the efficacy and effectiveness of our algorithm on real-world, scattered datasets for shape representation and computing.

© 2009 Elsevier Ltd. All rights reserved.

1. Introduction and motivation

With the rapid new development of modern scanning and data acquisition technologies, complicated geometric models with a huge number of point clouds are routinely collected by 3D scanners everyday at an explosive speed. Converting these acquired datasets from raw point clouds into smooth surface representations has become an important task that gives rise to a wide range of visual computing applications including reverse engineering, geometric modeling and processing, interactive 3D graphics, computational vision, etc. This reconstruction process from raw data to a smooth shape is often called surface reconstruction, or referred to as surface fitting, i.e., the interpolation or approximation of unorganized points or structured (gridded or triangulated) data using continuous spline-based representation. At present, there are three major types of smooth surfaces being employed in this problem: *algebraic surfaces*, *subdivision surfaces*, or *parametric piecewise polynomial surfaces*.

Generating *algebraic surfaces* from raw datasets is to find a smooth function whose zero level-set is in close proximity to the given point cloud with minimized deviation. Bajaj et al. [1] and Moore and Warren [2] described the methods for fitting G^1 piecewise algebraic surfaces of arbitrary topology from an unorganized set of points. Pratt [3] and Taubin [4] minimized the sum of squared Hausdorff distances from the data points to the zero level-set of a polynomial in three variables. Algebraic surface reconstruction methods are usually limited to simple and special shapes.

The basic idea of *subdivision surface* fitting is to get a smooth shape close to the given scattered data from a polygonal mesh by repeatedly adding new vertices and edges according to certain subdivision rules. Hoppe et al. [5] presented an approach for automatically fitting subdivision surfaces from dense triangle meshes, which can handle both smooth and sharp models. Later, Lee et al. [6] used subdivision surfaces with displacement maps. Suzuki et al. [7] proposed a method for subdivision surface fitting starting from an interactively defined control mesh. Ma et al. [8] presented a direct approach for subdivision surface fitting from a subset of initial vertices of the given dense triangle mesh. Subdivision-based representations of complex geometry can be manipulated and rendered very efficiently, which make subdivision surfaces a very desirable tool for animation and visualization purposes, however, they are not commonly supported with current CAD modeling and design systems.

---

* Corresponding author. Tel.: +1 225 578 0289; fax: +1 225 578 4831.
E-mail addresses: cccjjqm@gmail.com (J. Cao), xinli@lsu.edu (X. Li), wanggz@zju.edu.cn (G. Wang), qin@cs.sunysb.edu (H. Qin).

Spline-based algorithms have been widely studied and employed since they are well suited for further processing in CAD/CAM/CAE systems for FEM analysis and simulation purposes. Therefore, NURBS, as an industrial standard tool in computer-aided geometric design, becomes the most popular smooth surface representation used in surface fitting. Krishnamurthy and Levoy [9] proposed an algorithm of using B-spline surfaces to fit dense meshes, in which displacement was used to capture the fine details. Eck et al. [10] proposed a scheme to automatically fit meshes with arbitrary topological types by using B-splines. Greiner and Hormann [11] used hierarchical B-splines to interpolate and approximate scattered data by optimizing the output surface with respect to a fairness functional. Several ideas and methodologies on obtaining parameters for NURBS surface fitting toward randomly organized data were presented by Piegl and Tiller [12]. However, the inherent rectilinear shape of tensor-product B-splines severely limits their applicability in modeling general surfaces with arbitrarily complicated topological types. Specially, since the general surface is usually defined as a network of (trimmed) B-spline patches, it is very challenging to enforce the continuity across boundaries of different patches. Thus, it naturally gives rise to the urgent demand for the generalization of B-splines over triangular or polygonal domains as a more powerful modeling tool in surface fitting.

Thus far, much research work has been conducted on bivariate generalization of univariate B-splines, such as triangular Bézier patches [13], B-patches [14], box splines [15], and simplex spline [15]. Moreover, there are splines which are designed from triangular Bézier patches or simplex splines and have more nice properties than Bézier patches or simplex splines themselves. For example, triangular G^1-splines [17,18] are essentially composed of triangular Bézier patches with G^1 continuity along the boundary, which are the hierarchical version of the basic method proposed in [19,20]. Surfaces constructed by triangular G^1 splines surface can have arbitrary topology and have been used in fitting problem [21]. DMS splines [22,23] is one of the most powerful schemes among the generalization based on simplex spline, because of its numerous positive characteristics such as automatic smoothness properties, the ability to define surfaces over triangular or multi-sided domains. It has also been widely used in surface fitting [24–30]. However, a limitation of DMS spline is that, for any given set of knots, one has to explicitly add a set of auxiliary knots to each vertex within the triangulated domain in advance in order to form a knot sequence for the proper definition of each basis function and its evaluation. At present, it is still not clear how these auxiliary knots (being associated with the domain triangulation and its vertices) could affect the spline basis and the final surface in a quantitative way. Neamtu [31,32] proposed a new bivariate B-spline based on Delaunay configurations which possesses important and attractive properties in the same way as DMS splines do, such as optimal smoothness, linear independence, and polynomial reproducibility. What is more significant is that this new spline scheme could enable the knot selection procedure in a very elegant and natural way. Furthermore, this scheme can completely avoid the non-intuitive addition of auxiliary knots to each vertex within the domain triangulation.

Bivariate B-splines based on Delaunay configuration is one of the simplest and most effective natural generalizations of univariate B-splines. We therefore design practical algorithms for surface reconstruction by using cubic bivariate B-splines with Delaunay configurations in this paper. The specific contributions of our novel reconstruction framework include:

1. We present a surface reconstruction framework based on Delaunay configuration pioneered by Neamtu [31]. Our parameterization and reconstruction framework is able to define

such novel spline scheme for effective modeling of free-form surfaces. Under this scheme, no auxiliary knots outside of the original knots are necessary for basis function construction. This uniquely distinguishes our work from existing fitting based on DMS splines, which a knot sequence must be associated.
2. Without converting data into quadrilateral meshes, this framework can handle scattered or dense points from closed triangular meshes with arbitrary topology.
3. The fitting process is adaptive and capable of satisfying a user-specified error tolerance. And the final fitted surface is represented explicitly by a collection of spline basis functions and their corresponding control points.

Our method essentially generates a spline for a surface of arbitrary topology in three steps:

1. We segment the input mesh data[1] into a set of topological disk-like patches and grow their boundaries outward to cover a few more rings. This patch expansion allows vertices near the boundary of each patch to be always covered by multiple patches.
2. We parameterize each patch and map it onto a unit disk that serves as a canonical parameter domain. Here, we only conduct local parameterization. Then we place initial knots on each parameter domain in order to compute Delaunay configurations towards the effective computation of basis functions. Fitted surfaces are obtained after solving a least-square problem.
3. We can progressively and adaptively refine the fitting result by inserting additional knots to the regions where large fitting errors occur. A set of spline surfaces can be obtained with improved fitting quality.

The rest of this paper is organized as follows: Section 2 reviews the related work on simplex splines and the bivariate B-splines generalized from simplex splines. In Section 3, we introduce Delaunay configurations and bivariate splines defined over this type of domains. Section 4 formulates the problem of surface fitting. In Section 5, we describe details of our algorithm for surface fitting by using Delaunay configuration B-splines. Experimental results are shown in Section 6. Finally, we conclude our paper and discuss future work in Section 7.

2. Background and related work

2.1. Simplex splines

The simplex spline [16], defined by de Boor in 1972, is a multivariate generalization of the well-known univariate B-splines of Scheoenberg [33]. A degree-k simplex spline $B(\cdot|X)$ is a smooth, degree-k, piecewise polynomial function defined over a set of $k + s + 1$ points $X \subset \mathbb{R}^s$ called *knots*. We focus on the bivariate case in this paper and refer readers to [34] (and references therein) for a detailed illustration and discussion of theories of simplex splines. A bivariate simplex spline of degree-zero is defined by a collection of three non-collinear knots t_1, t_2 and t_3, and is simply a normalized characteristic function of the triangle whose vertices are these knots. More specifically, if we

---

[1] If the input data are point clouds, point-cloud segmentation (e.g. [57]) should be applied, and in step 2, point-based parameterization (e.g. [55]) techniques should be used.

denote by $M(\cdot|t_1, t_2, t_3)$ the degree-0 simplex spline with knots $\{t_1, t_2, t_3\}$, then

$$M(x|t_1, t_2, t_3) = \begin{cases} 1/area[t_1, t_2, t_3], & x \in int[t_1, t_2, t_3], \\ 0 & \text{otherwise}, \end{cases}$$

where $[\ldots]$ denotes the convex hull of a set of points and $int[\ldots]$ denotes its interior.

Degree n simplex splines defined over a knot set X with size of $n+3$ can be recursively expressed in terms of lower-degree ones as

$$M(x|X) = \frac{n+2}{n} \sum_{y \in X} \lambda_y M(x|X \setminus \{y\}), \quad x \in \mathbb{R}^2, \tag{1}$$

where the numbers $\lambda_y \in \mathbb{R}, y \in X$, are chosen such that

$$\sum_{y \in X} \lambda_y y = x, \quad \sum_{y \in X} \lambda_y = 1.$$

2.2. Generalization of univariate B-splines

In order to model geometric curves, the classical univariate B-splines [33] are among the most versatile control schemes. Nowadays, they have become standard tools in computer-aided geometric design and modeling of curves and surfaces. Given the success of univariate B-splines, one naturally desires to have its multivariate analog, such as surfaces. However, extending this technique onto surface patches has proven to be a non-trivial endeavor. One straightforward way is the tensor-product B-splines, which generalizes a univariate B-spline in two parametric mutually orthogonal directions naturally. However, the main limitation of the tensor-product B-splines in shape modeling is that the underlying shape (and the parametric domain) has to be "topologically" rectangular. This largely restricts the applicability of any non-tensor-product scheme. A more powerful generalized bivariate B-spline, which can define surfaces over triangular or polygonal domains, is very desirable for CAD designers/users.

On the other hand, individual simplex splines carry many nice properties of univariate B-splines [31], therefore, it becomes another popular direction towards generalizing univariate B-splines to multivariate ones without making use of the tensor-product mechanism. The essential problem is that: for given any generic knot set K, one would like to choose appropriate collections of knot sets of size $k+3$ to construct simplex splines of degree k, such that the space spanned by these simplex splines has analogous properties to the univariate B-spline space. We refer the readers to a survey [31] by Neamtu for more generalization methods based on simplex splines.

Among existing generalization methods, the DMS spline is probably one of the most powerful and well-known ones. It has been widely studied and applied in many applications. For example, Pfeifle and Seidel [27] proposed an algorithm to evaluate quadratic triangular B-splines and used it to fit scattered functional data by using least squares and optimization techniques. Rational DMS splines were employed by Han and Medioni [28] to model and visualize sparse and noisy data, which may contain unspecified discontinuous edges and functions. Dynamic triangular NURBS, a free-form shape model based on principles of physical dynamics, was proposed by Qin and Terzopolous [29]. He et al. [30] derived the evaluation formulas for the directional derivatives of DMS splines with respect to knots and applied them to surface fitting. In most bivariate generalizations based on simplex splines, one has to add auxiliary knots to each generic knot in advance to form all the required knot sequence for each basis function. This auxiliary knot generating procedure is both labor-intensive and far less-intuitive, and it is impossible to accurately determine how these auxiliary knots influence the final shape of each basis function as well as the entire spline surface in a quantitative manner.

Recently, Neamtu [31,32] presented a new method of using simplex splines to generalize univariate B-spline onto its multivariate analogy. This idea is based on the concept of Delaunay configurations from computational geometry. Within this scheme, new basis functions share most excellent properties with DMS splines such as optimal smoothness and polynomial reproducibility. Furthermore, unlike DMS spline which has to add auxiliary knots to each generic knot before the evaluation for each basis function, this new generalization method uses circles to choose closest knots as knot sets to define each basis function (see Section 3), which is very similar to the knot selection criteria of univariate case. These elegant properties motivate us to develop practical CAD applications such as surface fitting using this new bivariate B-spline.

3. Theory of Delaunay configuration B-splines

3.1. Delaunay configuration

Given a set of knots $K \subset \mathbb{R}^2$, a Delaunay triangulation [35] of K is subsets of size-3, where no other knots are inside the circumsphere defined by each subset. The Delaunay configuration is a higher degree analog of the Delaunay triangulation, defined as follows.

Definition 1. A degree-k Delaunay configuration of a given set of knots $K \subset \mathbb{R}^2$ is a pair

$$X = (X_B, X_I), \tag{2}$$

such that

$$X_B, X_I \subset K, \quad \#X_B = 3, \quad \#X_I = k, \tag{3}$$

and such that the circumsphere of X_B contains only X_I in its interior and no other knots from K, where # means the size of the set (see Fig. 1). The family of all Delaunay configurations of degree k associated with the set K is denoted as $\tilde{\Delta}_k$. $\tilde{\Delta}_0$ is the Delaunay triangulation of K.

Subscripts "B" and "I" in the definition stand for the "boundary" and "interior", respectively. Note that some Delaunay configurations might be different as pairs, yet collectively forming the same sets. In particular, there could be two different configurations (X_B^1, X_I^1) and (X_B^2, X_I^2), corresponding to the same set of points, i.e., $X_B^1 \cup X_I^1 = X_B^2 \cup X_I^2$, but $X_B^1 \neq X_B^2, X_I^1 \neq X_I^2$. (An example is illustrated
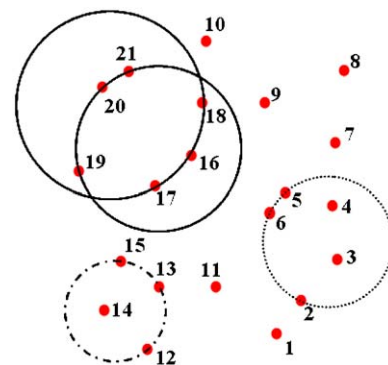

Fig. 1. Some Delaunay configurations of point set $\{1, \ldots, 21\}$: degree-1 Delaunay configuration $\{\{12, 13, 15\}, \{14\}\}$; degree-2 Delaunay configurations $\{\{2, 5, 6\}, \{3, 4\}\}$; degree-3 Delaunay configurations $\{\{19, 20, 21\}, \{16, 17, 18\}\}$ and $\{\{16, 17, 18\}, \{19, 20, 21\}\}$ with identical point set.

in Fig. 1: degree-3 Delaunay configurations $\{\{19, 20, 21\}, \{16, 17, 18\}\}$ and $\{\{16, 17, 18\}, \{19, 20, 21\}\}$.) We denote the "un-oriented" Delaunay configurations, consisting of knot sets of form $X_B \cup X_I$, i.e., without distinguishing the boundary and interior knots, as Δ_k.

The notion of a Delaunay configuration is implicit in papers concerned with the computation of higher-order Voronoi diagrams [36–38]. It is proved that the number of k-degree Delaunay configurations of given knot set with size n is bounded by $(2k + 1)n$ (refer to [36] for more details).

3.2. Delaunay configuration B-spline surface

The set of simplex splines from Delaunay configurations has many attractive properties (proved in [31]) such as partition-of-unity, polynomial reproduction, and degenerate case being univariate B-splines. Moreover, if we remove the multiple entries in the collections of simplex splines from Delaunay configurations, we can get linear independent splines.

Definition 2. Let $k \geqslant 0$, $X_T = \{(X_B, X_I) | X_B \cup X_I = T, T \in \Delta_k\}$. We define simplex splines from un-oriented Delaunay configurations as

$$B_T = \sum_{(X_B, X_I) \in X_T} |area[X_B]| M(\cdot | T), \qquad (4)$$

where $area[X_B]$ is the area of the triangle determined by X_B. We call it Delaunay Configuration B-spline (DCB), which are the sum of normalized simplex splines defined from Delaunay configurations with the identical knot set. The surface constructed by Delaunay configuration B-splines (abbreviated as DCB surface) is defined as

$$S = \sum_{T \in \Delta_k} B_T c_T, \qquad (5)$$

where $c_T \in \mathbb{R}^3$ is the control point corresponding to the basis function B_T.

4. Surface reconstruction from scattered points with DCB

The goal of surface reconstruction from scattered points is to determine a compact representation surface M, such that M approximates an unknown surface partially determined by a set of sample points $X = \{x_1, \ldots, x_n\}, x_i \in \mathbb{R}^3$. In this paper, we take vertices of a polygonal (triangular) mesh as X, and use parametric cubic DCB to fit surface M.

If the data points $\{x_1, \ldots, x_n\}$ stay on a *topological disk-like* patch and correspond to known parameters $\{u_1, \ldots, u_n\}$ on a parameter domain D, the case is much simpler: the parametric surface fitting is then formulated as how we can determine all control points to minimize the error vector:

$$x_i - S(u_i),$$

where for $(t \in D)$, $S(t)$ is the parametric surface defined over D. Therefore, globally we will minimize these quantities in a least-square sense as follows:

$$minimize \sum_{i=1}^{n} \| x_i - \sum_s B_s(u_i) c_s \|^2, \qquad (6)$$

where $\sum_s c_s B_s(u_i)$ is the DCB surface with control point c_s.

Our framework aims to find a DCB surface to approximate given sample points from *closed surfaces with arbitrary topology*. This case is much more complicated and technically challenging, and it needs a proper parameterization in advance. Globally parameterizing a surface cannot avoid singularity points (unless the underlying surface is of genus-1). In addition, technical issues

such as controlling the area-stretching distribution globally and retaining enough sampling rates near sharp geometric features also arise in global approaches without satisfactory solutions. Therefore, we resort to local conformal parameterizations to generate the covering parametric domains. Local methods give rise to better flexibility, user control, and improved time/space performance, but will need to specifically address the issue of segmentation and patch blending. We explain our idea with details and illustrate our implementation algorithm in the next section.

5. Algorithm description

5.1. Overview

In the literature of surface fitting, a surface with arbitrary topological types is usually defined as a network of patches. And among adjacent patches, certain degree of continuity at each patch's boundary is guaranteed by enforcing some constraints on the control points. In contrast, in this work, our obtained surface can be viewed as a "one-piece" representation. The input of the fitting algorithm is a dense triangular mesh $M_n = (X = \{x_1, x_2, \ldots, x_n\}, V)$, where n is the vertex number, and V are the connectivity of the vertices. M_n is closed and of arbitrary topology. Our output is a one-piece cubic DCB parametric surface.

For the purpose of convenience, we shall first introduce notations used in this paper. The subset of X is denoted as $X_k, k = 1, 2, \ldots, m$, where m is the total number of subset X_k. We denote the size of X_k by $\#X_k$. Suppose Γ is a positive integer set, then we use x_i^Γ to represent a vertex $x_i \in X$ is involved in subset $X_k, k \in \Gamma$. For example, $x_1^{\{1,2,3\}}$ means vertex x_1 belongs to subset X_1, X_2 and X_3 at the same time. Vertex x_i^Γ with $\#\Gamma > 1$ and $\#\Gamma = 1$ is called *overlapped vertex* and *non-overlapped vertex*, respectively. Finally, we call a parametric domain related to X_k a *chart*, denote it as D_k, and use u_i^k to represent the parameter coordinate of vertex x_i on domain D_k.

A pipeline overview of our algorithm is illustrated in Fig. 2, our DCB surface fitting algorithm comprises three stages:

1. *Generating charts.* It includes a sequential pipeline of *mesh segmentation*, *extension* and *parameterization*. Mesh segmentation partitions scattered data X into several topological disk
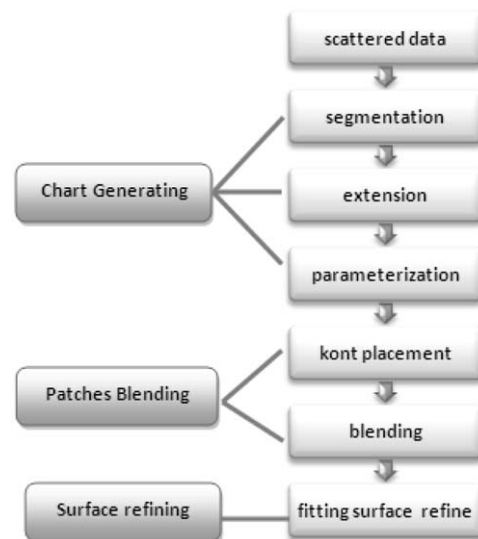

Fig. 2. The pipeline of our surface fitting algorithm.

patches $\widetilde{X_k}$; and then we *grow* each patch outward for several rings and get the patch X_k, so that vertices near boundaries of each patch become overlapped vertices. This extension will be used later to glue patches into a "one-piece" closed surface. Finally, we conformally *parameterize* each patches X_k to a unit disks, chart D_k (so that overlapped vertices have different parameters on different charts). In other words, charts of adjacent patches share overlapped regions.

2. *Patch blending.* First, on each chart D_k, we set preliminary knots according to the distribution of u_i^k on the parameter domain. Then we blend all patches together by using rational DCB basis functions to guarantee the "partition of unity" of the involved basis functions.

3. *Surface refinement.* We iteratively refine the spline surface by accurately estimating fitting errors. Knots are inserted locally in those regions whose approximation is not satisfied by the given threshold.

Fig. 3. Segmentation and expansion: (left) before expansion, (right) after expansion.

5.2. Generating charts

The first step in surface fitting is to find proper parametric domains and generate a good parametrization. A general surface can either be globally parameterized onto a single domain or be partitioned and parameterized separately onto a set of sub-domains. Globally parameterizing a complicated closed mesh oftentimes introduces larger angle and area distortion between the model and its corresponding domain, and when the surface is not a topological disk or torus, the mapping inevitably has singularities which are extremely difficult to handle. No bijectivity of the mapping is guaranteed on singularities, and their nearby regions usually has huge angle and area distortion. In contrast, when a surface is with complicated topology and geometry, if we segment it into geometrically flat sub-patches with compact boundaries, we can effectively obtain low-area-distorted and angle-distortion-free local parameterization, which is very important and attractive for our surface fitting purpose. Based on these observations, in our work we adopt the strategy of partitioning the mesh into sets of sub-patches and then individually compute their conformal parameterization onto unit disks.

5.2.1. Segmentation
A patch with flat geometry shape and compact boundary usually yields conformal parameterization with small area distortion. Therefore, segmentation techniques that can slice ridges and ravines and produce flat local subregions are much more desirable. Many recent surface segmentation techniques currently available in literatures (a detail discussion is beyond the scope of this work and we refer the readers to the survey [49]) can be used. The idea presented by [39] is simple and it fits our goal very well, therefore we employ this idea in generating our surface segmentation. The segmentation starts from a set of randomly selected seed faces on the original mesh, then these seeds grow to roughly flat and compact charts whose boundaries align with the model's crease. A necessary post-process is interactively con-ducted to merge or slice resultant sub-regions when they are too large or small to assure that each patch has a suitable size. As shown in Fig. 3 (left), the model of gargoyle are partitioned into total 12 patches, rendered in different colors.

5.2.2. Expansion
In order to continuously glue sub-patches to the closed surface, we want to assure regions near segmentation boundaries are covered by more than one chart. Therefore, we grow each sub-patch $\tilde{X}_i$, and let its extended region X_i covers larger part of the
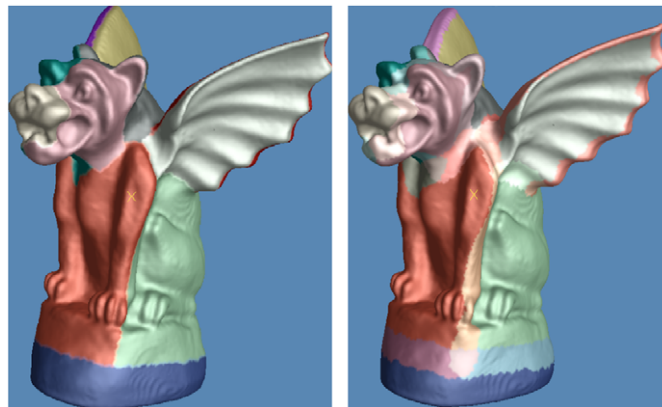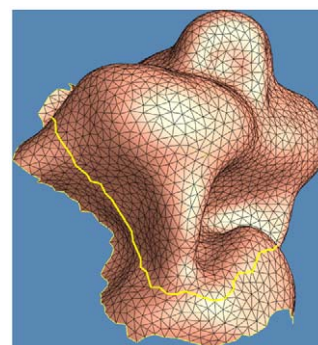

Fig. 4. Extension of patch of gargoyle's nose. The patch of gargoyle's nose is extended outward for 10 rings, and the boundary before extension is highlighted using yellow curve.

original surface. An efficient implementation of this over the triangular mesh is to iteratively expand each sub-patch (by adding in new triangles the growing boundary meets) for several times. We assure the disk-topology of the growing chart $\tilde{X}_i$ during this process. After the expansion, vertices nearby the sub-patch boundaries will all become "blending vertices", since they are covered by several sub-patches at the same time. In our experiments, we apply the expansion of each patch for 10-rings by default. Then we conduct a sequential post-expansion smoothing to remedy the potential "zigzag" boundary of the new patch X_i due to the combinatorial broadcasting. Intuitively, during the smoothing process, if a triangle face is not within a patch but two of its edges are on the patch boundary, then this face will be included into this patch; if a triangle face is in a patch, while neither of its neighboring faces is, then this face will be removed from the patch. Fig. 3 (right) shows the result of our expansion process: each shared sub-region is illustrated with the blended color of those patch that cover it. Fig. 4 illustrates another example where the patch of gargoyle's nose is expanded. We show the expanded chart, with its original segmentation boundary highlighted using the yellow curve.

5.2.3. Parametrization
Conformal maps [40,41] have been widely used in texture mapping, geometry meshing and visualization (see surveys [48,47] for more applications). It has many good properties in serving for the domain of spline functions. Specifically, it is

intrinsic to the geometry of the surface, independent of the mesh resolution, and free of angle distortion.

Boundary-fixed quasi-conformal mappings such as harmonic maps [51] and mean-value coordinators [50] usually have large angle distortion near the boundary regions, in contrast, boundary-free conformal mapping using circle packing [52] and curvature flow [53] are angle-distortion free but usually have non-compact sub-patch boundaries, therefore we compute the conformal mapping to the unit disk. The idea is based on the fact that harmonic mapping from a closed surface to the unit sphere is conformal. We *double-cover* the topological-disk surface P (i.e., duplicate P by gluing P to its inversely oriented copy P', along its boundary ∂P) to a genus-zero surface $\tilde{P}$, then compute harmonic mapping from $\tilde{P}$ onto the unit sphere $\mathbb{S}^2$. Because of the symmetry of the "double-covering" operation, the topological disk P is conformally mapped onto a hemisphere of $\mathbb{S}^2$. By composing a stereo-projection, we can obtain P's conformal parameterization onto the unit disk D_i (see [54] for more details of this algorithm).

Each patch X_i is parameterized onto a unit disk D_i. As shown in Fig. 5, the patch of the Gargoyle's nose is mapped onto a unit disk. Non-overlapped vertices, colored in gray, locate in the interior region of the chart. Vertices near the boundary are covered by several charts and rendered in different colors. As mentioned previously, overlapped points will be mapped onto different charts (with different parameter coordinates), and used for patches blending in the next subsection.

5.3. Patch blending and fitting

In this section, we construct a globally continuous DCB surface based on the blending of sub-patches and their corresponding parameterizations computed in Section 5.2.

5.3.1. Initial knot placement

In order to construct Delaunay configuration and DCB basis functions, we have to place initial knots on each chart. According to the classical B-spline theory, different choices of knots have considerable and drastic influence on the final shape of spline
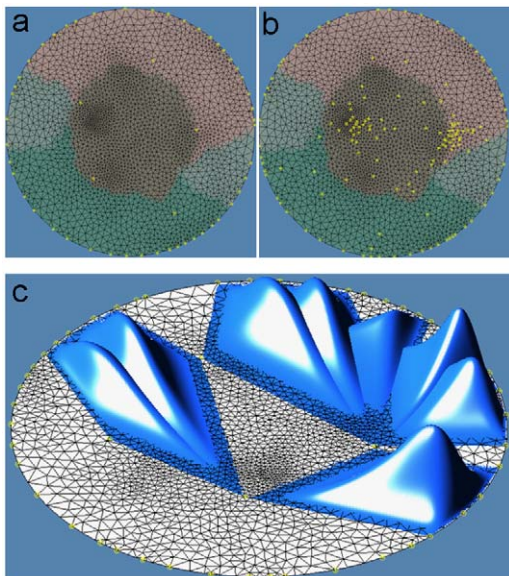

Fig. 5. Parameterizaiton, knot placement, and Delaunay configuration basis functions. (a) Parameterization of the patch of gargoyle's nose and initial knot placement; 53 initial knots are placed at the beginning of the algorithm. (b) Ninety more knots are placed during the refinement. (c) Some Delaunay configuration B-splines defined over the initial knot set in (a).

curves or surfaces. However, the automatic knot placement is a multivariate and multimodal nonlinear optimization problem. Therefore, many papers only deal with this problem by heuristic methods [42–44]. Intuitively, within the domain regions that have subtle geometric details and more parameter points, more knots are necessary. Based on this observation, we use K-mean cluster method [45,46] to place preliminary knots within the interior region of each chart. Specifically, for each chart with parameter points $P = \{u_1, u_2, \dots, u_n\}$ on it, the first k knots $t_1, t_2, \dots, t_k$ are determined by minimizing the following function:

$$\sum_{i=1}^{k} \sum_{u_j \in S_i} \|u_j - t_i\|^2,$$

where there are k clusters $S_i, i = 1, 2, \dots, k$ of parameters P and $\|u_j - t_i\|$ is the Euclidean distance between u_j and t_i. In addition to these initially added knots, we further add more knots by sampling the parameters points which are on the boundary of the disk to guarantee that all the overlapping vertices are covered by at least one chart. In our experiments, k is empirically set to be a ratio of the total vertex number $\lfloor \frac{\#X_i}{200} \rfloor$, and we also place 40 more knots at the boundary of the chart. In Fig. 5, total 53 preliminary knots are placed on the chart covering gargoyle's nose.

5.3.2. Basis function construction

Once we have knots on charts, we can compute Delaunay configurations and corresponding DCB basis functions on each individual chart (as described in Section 3). Suppose we generate total m_j DCB basis functions on chart D_j, denote these functions as $b_s^j(t), t \in D_j, s = 1, \dots, m_j$, and denote the set of b_s^j on chart D_j as $\theta_j = \{b_s^j\}|_{s=1}^{m_j}$.

So far, basis functions defined on one chart do not interact with functions on other charts. To obtain a closed DCB surface, we further use transition functions α_j^i to extend the basis functions defined over chart D_i to an adjacent chart D_j. In particular, the extended basis functions are defined as follows:

$$\hat{B}_s^j(t) = \begin{cases} b_s^j(\alpha_j^i(t)) & \text{if } t \in \Delta_{i_1, i_2, i_3} \subset D_i, \\ 0 & \text{otherwise,} \end{cases} \tag{7}$$

where $x_{i_1}, x_{i_2}, x_{i_3} \in X_i \bigcap X_j$, Δ_{i_1, i_2, i_3} is the triangle formed by their parameter points $u_{i_1}^i, u_{i_2}^i, u_{i_3}^i \in D_i$ and

$$\alpha_j^i(t) = v_1 u_{i_1}^j + v_2 u_{i_2}^j + v_3 u_{i_3}^j,$$

with (v_1, v_2, v_3) being the barycenter coordinate with respect to Δ_{i_1, i_2, i_3}, i.e., $t = v_1 u_{i_1}^i + v_2 u_{i_2}^i + v_3 u_{i_3}^i$. To ensure the basis functions sum up to one, we normalize basis functions obtained in Eq. (7) as follows:

$$B_s^j(t) = \frac{\hat{B}_s^j(t)}{\sum_{j=1}^{m} \sum_{s=1}^{m_j} \hat{B}_s^j(t)}, \tag{8}$$

where $t \in D_j, j = 1, 2, \dots, m$. The patches constructed by basis functions defined in Eq. (8) are smooth in the non-overlap region but with C^0 continuity in the blending region. In our future work, we plan to apply the global parameterization, so that the atlas has all transition functions among overlapped local charts being affine [56], and a globally more smooth result could be obtained along the regions covered by multiple charts.

5.3.3. Surface fitting

Given basis functions B_s^j and their associated control points c_s^j, the rational parametric surface is represented as

$$S(t) = \sum_{j=1}^{m} \sum_{s=1}^{m_j} B_s^j(t) c_s^j.$$

Therefore, the linear least-square problem in Eq. (6) becomes

$$minimize \sum_{i=1}^{n} \|x_i - \sum_{j=1}^{m} \sum_{s=1}^{m_j} B_s^j(u_i^j)c_s^j\|^2. \tag{9}$$

We simplify the notation by concatenating the data points and control points into vectors

$$X = [x_1, x_2, \ldots, x_n]' \quad \text{and} \quad C = [c_1^1, c_2^1, \ldots, c_s^j, \ldots, c_m^{m_m}]'$$

and rewrite Eq. (9) as

$$minimize \|X - AC\|^2, \tag{10}$$

where A is a $3n \times M$ sparse matrix with $M = \sum_{j=1}^{m} m_j$ being the total number of normalized DCB basis functions. We only treat control points as variables in Eq. (9). Therefore, this leads to a sparse linear least-square problem, which can be solved by using many existing methods. In our experiment, we use the singular value decomposition (SVD) method to get numerically stable results.

5.4. Surface refinement

Before this section, the surface reconstruction algorithm attempts to minimize the total squared distance of the scattered data X to a DCB surface. It is oftentimes desirable to specify an error tolerance ε, such that each item $\|x_i - \sum_{j=1}^{m} \sum_{s=1}^{m_j} B_s^j(u_i^j)c_s^j\|$ in Eq. (9) (denoted as e_j^i, representing fitting error of vertex x_i) is smaller than this threshold. Towards this goal, we must repeatedly refine the DCB surface by adding more knots on parametric domains. This refinement is performed as follows:

1. For each patch $X_i = \{x_{i_1}^i, x_{i_2}^i, \ldots, x_{i\#X_i}^i\}$, measure fitting error e_j^i at each vertex x_j^i and find the largest one $e_{j_0}^i$. If the maximal fitting error on every patches is smaller than the threshold ε, STOP.
2. For each patches X_i, if $e_{j_0}^i > \varepsilon$, triangulate the knot set of chart D_i and add one more knot at the barycenter of the triangle where $u_{j_0}^i$ is located. Then we locally update the Delaunay configurations and corresponding DCB basis functions, and update the corresponding coefficients matrix A in Eq. (10).
3. Solve the least-square problem Eq. (10), GOTO Step 1.

Note that at Step 2, Delaunay configurations can be updated locally by the high-order Delaunay tree algorithm in $O(n \lg n + 27n)$ expected time [36]. In addition, all the coefficient elements in matrix A can be dynamically updated with high efficiency when new knots are added.

This surface refinement algorithm is an iterative procedure, where the linear least-square problem (Eq. (9) or Eq. (10)) is solved repeatedly. If the size of the coefficient matrix A is large, this procedure takes longer time to satisfy the error threshold requirement. An effective and practical solution to overcome this deficiency is to first apply the procedure individually over non-overlapped vertices on each chart before we globally solve Eq. (9). Such a strategy essentially divides the problem of Eq. (9) into several linear least-square problems with coefficient matrices of much smaller size, therefore, it effectively reduces the total execution time and improves the adaptivity of our framework to large raw datasets.

Fig. 5 shows an example, 53 initial knots are placed as shown in (a), and after refinement, 90 more (yellow) knots are added.

6. Experimental results

We apply our surface fitting algorithm over several real scattered/unorganized data models. In our experiments, all the scattered data are uniformly scaled to fit within a unit cube to compare the fitting error across different models.

Fig. 6 illustrates the result of our surface reconstruction method on the gargoyle model. The model with 59k vertices is illustrated in (a); (b) shows our final reconstructed spline surface. Computed control points are rendered in red points in (c), and the fitting error is evaluated and color-encoded in (d).

Figs. 7, 8 and 9 show our experiments on the Rocker-Arm, Igea, and Kitten models, respectively. In each of these figures, the given model is shown in (a); its segmentation and extended covering charts are shown in (b) and (c); and the final spline surface and control points are illustrated in (d) and (e), with the fitting error color-encoded in (f).
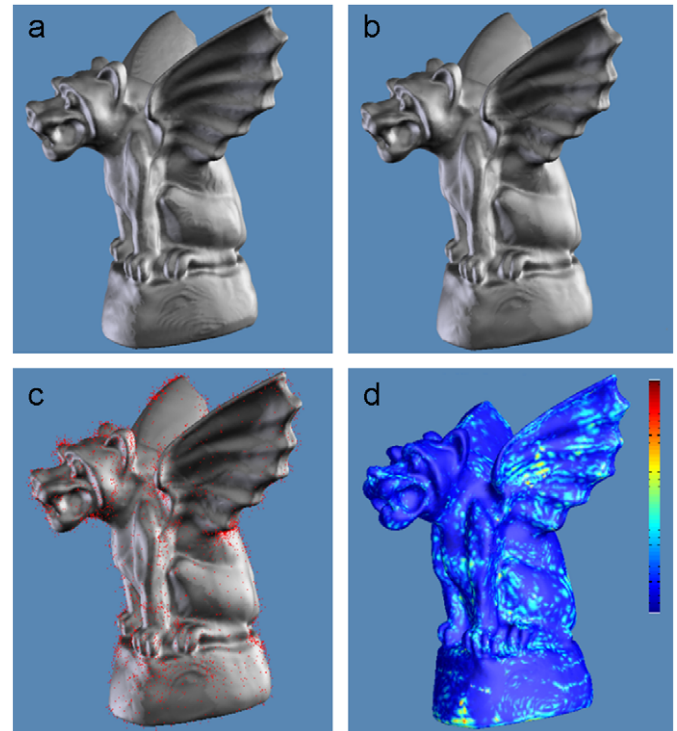

Fig. 6. Fitting gargoyle surface: (a) initial model (59k vertices and 119k faces); (b) final fitted surface; (c) fitted surface with 13k control points; (d) fitting error (maximum error 0.34% and root mean error 0.049%).
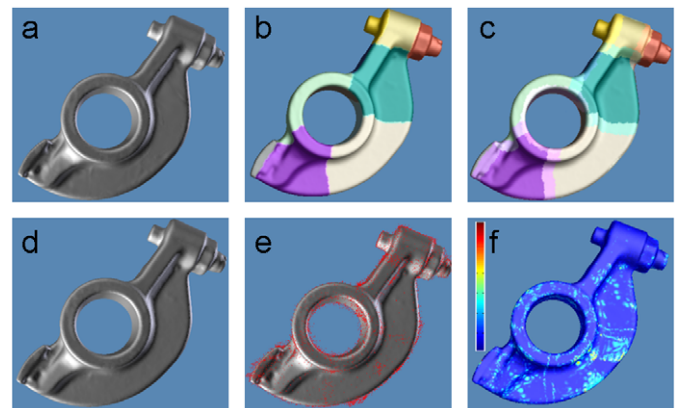

Fig. 7. Rocker-arm: (a) initial model (50k vertices and 100k faces); (b) model segmentation (15 patches); (c) patch expansion; (d) final fitted surface; (e) fitted surface with 13k control points; (f) fitting error (maximum error 0.11% and root mean error 0.012%).
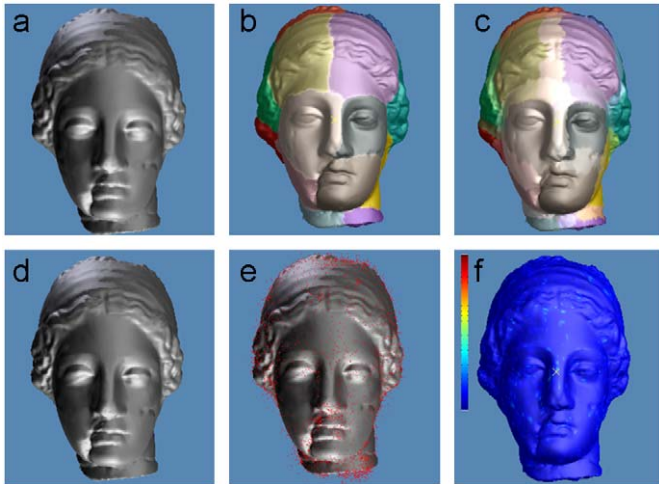
Fig. 8. Igea: (a) initial model (50k vertices and 100k faces); (b) model segmentation (20 patches); (c) patch expansion; (d) final fitted surface; (e) fitted surface with 12k control points; (f) fitting error (maximum error 0.44% and root mean error 0.024%).
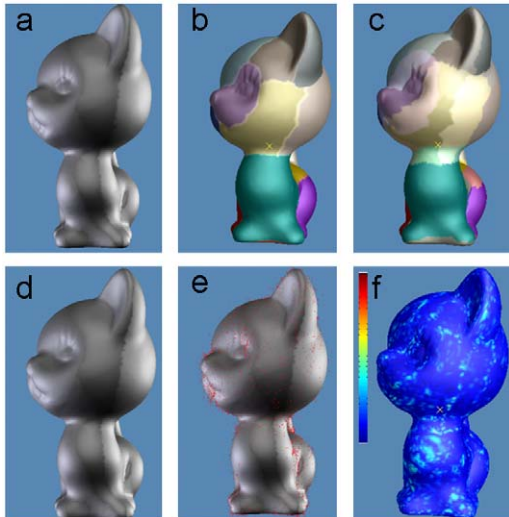

Fig. 9. Kitten: (a) initial model (33k vertices and 70k faces); (b) model segmentation (17 patches); (c) patch expansion; (d) final fitted surface; (e) fitted surface with 7k control points; (f) fitting error (maximum error 0.15% and root mean error 0.027%).

As we can find out in Figs. 6(c), 7(e), 8(e), and 9(e), more control points locate around protuberant features, because under conformal parameterization these regions have larger area stretching factors. This demonstrates from another aspect that local segmentation and separate parameterization fits our surface reconstruction framework very well because global surface parameterization inevitably leads to larger area and angle distortion. We will also explore other novel parameterization techniques that can further reduce area and angle distortion in our future work.

The fitting error of each vertex is normalized (by dividing maximum error) as

$$\frac{e_{j_0}^i}{max\{e_{j_0}^1, e_{j_0}^2, \ldots, e_{j_0}^m\}}.$$

We color-code fitting errors using the scale 0 (blue) to 1 (red) as shown in the color-bar of Figs. 6(d), 7(f), 8(f), and 9(f),

Table 1
The statistics of scattered datasets and surface fitting configurations.

Model	N_v	N_c	N_d	m.e. (%)	rms (%)	it-time (g)
Gargoyle	59k	9k	12	0.34	0.049	51.9
Rocker-arm	50k	13k	15	0.11	0.012	74.8
Igea	50k	12k	20	0.44	0.024	70.3
Kitten	33k	7k	17	0.15	0.027	30.0

N_v, # of vertex; N_f, # of faces; N_c, # of control points, N_d, # of domain; m.e., maximum error; rms, root-mean-square error. it-time, the running time for one global fitting iteration.

respectively. Experimental results demonstrate that our DCB surfaces approximate given data very well.

Table 1 shows the statistical performance of our surface fitting procedure on our experimental models. N_v denotes the vertex number of the scattered model. N_f is its triangular face number. N_c is the control point number and N_d is the chart number. The maximum fitting error is denoted as m.e. while the root-mean-square error is denoted as rms. For most of our test models, the process of segmentation takes less than 5 min. The subsequent chart generating and expanding are linear and each topological-disk chart is parameterized by the standard discrete harmonic map, which leads to two sparse linear systems and are solved in a few seconds. As mentioned in Section 5.3.3, we fit each patches, respectively, before the global fitting. One iteration of local fitting for a patch with about 70 knots and 9000 vertices usually takes about 0.75 s in our Intel 2.5G Duo CPU. Necessary local fitting step varies among different shapes and different rms threshold, and in our experiments it usually takes 40–120 iterations to reach 0.049-rms. After the local fitting, about 2–15 iterations of global fitting are necessary to reach the rms of the entire surface.

7. Conclusion and future work

In this paper, we have developed a new bivariate B-spline scheme to reconstruct closed and arbitrary topological surfaces, based on the novel Delaunay configuration of knots.

Compared to other surface fitting scheme, using DCB provides an elegant and simple knot selection rule. In computer graphics, some uses subdivision surface for surface fitting. However, in the CAD/CAM industry (for instance, scientific computations and physical simulations/analysis), explicit parametric representations (e.g. splines) are desirable because they provide more effective and accurate differential operator evaluations, higher surface continuity and so on. For splines whose fitting methods are based on the tensor-product B-spline or NURBS surface. The result surfaces are usually defined as a network of tensor-product B-spline patches. Due to the inherent rectilinear shape of tensor-product surface, surfaces with non-trivial topology usually require a non-trivial stitching process with continuity restrictions. Fundamentally, bivariate splines defined over triangular or polygonal domains are more powerful modeling tools. For example, both DMS and DCB have the property of automatic-smoothness, therefore they can define a surface over arbitrary triangulation.

The biggest advantage of using DCB over DMS is the natural knot selection rule. Specially, when constructing DMS splines, generating the knot sets for basis construction needs the non-intuitive insertion of auxiliary knots to be associated with each generic knot in order to form a knot sequence. In contrast, our DCB splines use circles to choose knots that are closer to each other as a natural set in order to construct basis functions. The spline space spanned from this set of basis functions naturally behaves like univariate B-splines. Experimental examples demonstrate that

one can achieve good fitting results when using DCB splines for surface reconstruction through adaptive refinement and error threshold control.

Despite that DCB is mathematically elegant and conceptually simple, limitations of our current scheme include that first, we only focus on closed surfaces. The basis functions being defined are always zero at the boundary of its corresponding parameter domain in our current settings. Therefore, they cannot be used to construct a function with non-zero value on the boundary, i.e., they cannot be trivially used to model surfaces with curved boundary. One possible solution is that we shall place knots outside the domain chart, and then trim the corresponding region of the fitted surface.

Second, like many other spline schemes, modeling sharp feature is another challenging issue. A natural way is to enforce more special constraints on knot placement, i.e., to place multiple and collinear knots along the shape feature. At present, all of our basis functions are constructed isometrically (locally no three knots are collinear and no four knots are co-circular), therefore our spline surface would appear too smooth in the vicinity of feature-rich regions. In the future, robust feature detection and flexible knot placement strategies for knots generation will be studied in this framework.

Third, our current blending region has C^0 continuity due to the local parameterization scheme. Global atlas generation with affine transition functions among overlapped local charts will be used in our framework to obtain more smooth result along patch boundaries.

This paper shows its advantage over multivariate splines and demonstrate its theoretical elegance. Our ongoing work is on further improving the efficiency of the algorithm and fitted surface quality. The trade-offs in computation, memory usage, or surface quality will be the discussed with more details.

The application we have focused on in this paper is surface fitting for reverse engineering and shape modeling. Many good properties of this new spline scheme are inspiring us to foresee and explore its broader application scopes in trivariate splines for solid modeling, FEM-based dynamic analysis, and interactive deformation.

Acknowledgments

References

[1] Bajaj CL, Bernardini F, Xu G. Automatic reconstruction of surfaces and scalar fields from 3d scans. In: Proceedings of the 22nd annual conference on computer graphics and interactive techniques. New York: ACM Press; 1995. p. 109–18.

[2] Moore D, Warren J. Approximation of dense scattered data using algebraic surfaces. In: Proceedings of the 24th annual Hawaii international conference on system science, Kauai, HI, USA. Silver Spring, MD: IEEE Computer Society Press; 1991. p. 681–90.

[3] Pratt V. Direct least-squares fitting of algebraic surfaces. In: Computer graphics (SIGGRAPH 87 proceedings), vol. 21(4), July 1987. p. 145–52.

[4] Taubin G. Estimation of planar curves, surfaces and nonplanar space curves defined by implicit equations, with applications to edge and range image segmentation. Technical Report LEMS-66, Division of Engineering, Brown University; 1990.

[5] Hoppe H, DeRose T, Duchamp T, Halstead M, Jin H, McDonald J, et al. Piecewise smooth surface reconstruction. In: SIGGRAPH computer graphics proceedings. New York: ACM; 1994. p. 295–302.

[6] Lee A, Moreton H, Hoppe H. Displaced subdivision surfaces. In: Akeley K, editor. Siggraph 2000, computer graphics proceedings. New York: ACM Press, ACM SIGGRAPH, Addison-Wesley Longman; 2000. p. 85–94.

[7] Suzuki H, Takeuchi S, Kanai T. Subdivision surface fitting to a range of points. In: Proceedings of seventh pacific conference on graphics and applications (pacific graphics'99). Silver Spring, MD: IEEE Computer Society; 1999. p. 158–67.

[8] Ma W, Ma X, Tso S, Pan Z. A direct approach for subdivision surface fitting from a dense triangle mesh. Computer Aided Design 2004;36(6):525–36.

[9] Krishnamurthy V, Levoy M. Fitting smooth surfaces to dense polygon meshes. In: Computer graphics (SIGGRAPH'96 proceedings), 1996. p. 313–24.

[10] Eck M, Hoppe H. Automatic reconstruction of b-spline surfaces of arbitrary topological type. In: Proceedings of the 23rd annual conference on computer graphics and interactive techniques. New York: ACM Press; 1996. p. 325–34.

[11] Greiner G, Hormann K. Interpolating and approximating scattered 3d data with hierarchical tensor product bsplines. In: Mhaut A, Schumaker LL, editors. Curves and surfaces with applications in CAGD. New York: Vanderbilt University Press; 1997.

[12] Piegl L, Tiller W. Parameterization for surface fitting in reverse engineering. Computer Aided Design 2001;33(8):593–603.

[13] Farin G. Curves and surfaces for CAGD. 5th ed. New York: Morgan Kaufmann; 2002.

[14] Seidel HP. Symmetric recursive algorithms for surfaces: b-patches and the de Boor algorithm for polynomials over triangles. Constructive Approximation 1991;7:257–79.

[15] de Boor C, Hollig K, Riemenschneider SD. Box splines. Berlin: Springer; 1993.

[16] de Boor C. Splines as linear combinations of B-splines. A survey. In: Lorentz GG, Chui CK, Schumaker LL, editors. Approximation theory II. New York: Academic Press; 1976. p. 1–47.

[17] Hahmann S, Bonneau G-P. Triangular G1 interpolation by 4-splitting domain triangles. Computer Aided Geometric Design 2000;17(8):731–57.

[18] Yvart A, Hahmann S, Bonneau G-P. Hierarchical triangular splines. ACM Transactions on Graphics 2005;24(4):1374–91.

[19] Peters J. C1-surface splines. SIAM Journal on Numerical Analysis 1995;32(2):645–66.

[20] Gonzalez-Ochoa C, Peters J. Localized-hierarchy surface splines (LeSS). In: Proceedings of the 1999 symposium on interactive 3D graphics. New York: ACM Press; 1999. p. 7–15.

[21] Yvart A, Hahmann S, Bonneau G-P. Smooth adaptive fitting of 3D models using hierarchical triangular splines. In: International conference on shape modeling and applications, SMI'05. Cambridge, MA, Boston: MIT, IEEE Computer Society Press; 2005. p. 13–22.

[22] Greiner G, Seidel HP. Modeling with triangular B-splines. IEEE Computer Graphics and Applications 1994;14:56–60.

[23] Franssen MGJ. Evaluation of DMS-splines. Master's thesis, Eindhoven University of Technology; 1995.

[24] Fong P, Seidel H-P. Control points for multivariate B-spline surfaces over arbitrary triangulations. Computer Graphics Forum 1991;10(4):309–17.

[25] Fong P, Seidel H-P. An implementation of multivariate B-spline surfaces over arbitrary triangulations. In: Proceedings of graphics interface'92, 1992. p. 1–10.

[26] Greiner G, Seidel H-P. Modeling with triangular B-splines. IEEE Computer Graphics and Applications 1994;14(2):56–60.

[27] Pfeifle R, Seidel H-P. Fitting triangular B-splines to functional scattered data. Computer Graphics Forum 1996;15(1):15–23.

[28] Han S, Medioni G. Triangular NURBS surface modeling of scattered data. In: Proceedings of the 7th conference on visualization'96, 1996. p. 295–302.

[29] Qin H, Terzopoulos D. Triangular NURBS and their dynamic generalizations. Computer Aided Geometric Design 1997;14(4):325–47.

[30] He Y, Qin H. Surface reconstruction with triangular B-splines. In: Proceedings of GMP'04, 2004. p. 279–90.

[31] Neamtu M. What is the natural generalization of univaritate splines to higher dimensions? In: Mathematical methods for curves and surfaces: Oslo 2000, 2001. p. 355–92.

[32] Neamtu M. Bivariate simplex B-splines: a new paradigm. In: Proceedings of the 17th spring conference on computer graphics, 2001. p. 71–8.

[33] Curry HB, Schoenberg IJ. On Pólya frequency functions IV: the fundamental spline functions and their limits. Journal d'Analyse Mathematique 1966;17:71–107.

[34] Micchelli CA. Mathematical aspects of geometric modeling. In: CBMS-NSF regional conference series in applied mathematics, vol. 65. Philadelphia, PA: SIAM; 1995.

[35] Delaunay B. Sur La sphere vide, lzvestia Skademii Nauk SSSR. Otdelenie Matematicheskikh i Estestvennykh Nauk 1934;7:793–800.

[36] Boissonnat JD, Devillers O, Teillaud M. A semidynamic constructions of higher-order Voronoi diagrams and its randomized analysis. Algorithmica 1993;9:329–56.

[37] Okabe A, Boots B, Sugihara K. Spatial tessellations: concepts and applications of Voronoi diagrams. 2nd ed. Chichester, England: Wiley; 2000.

[38] Preparata FP, Shamos MI. Computational geometry. Berlin: Springer; 1985.

[39] Sander P, Wood Z, Gortler S, Snyder J, Hoppe H. Multi-chart geometry images. In: Eurographics symposium on geometry processing'03, 2003. p. 146–55.

[40] Levy B, Petitjean S, Ray N, Maillot J. Least squares conformal maps for automatic texture atlas generation. In: SIGGRAPH 02, 2002. p. 362–71.

[41] Gu X, Yao S-T. Global conformal surface parameterization. In: SGP'03: proceedings of the 2003 eurographics/ACM SIGGRAPH symposium on geometry processing. Aire-la-Ville, Switzerland: Eurographics Association; 2003. p. 127–37.

[42] Yoshimoto F, Harada T, Yoshimoto Y. Data fitting with a spline using a real-coded genetic algorithm. Computer Aided Design 2003;35:751–60.

[43] Li WS, Xu SH, Zhao G, Goh LP. Adaptive knot placement in B-spline curve approximation. Computer Aided Design 2005;37:791–7.

[44] Razdan A. Knot placement for B-spline curve approximation. Tempe, AZ: Arizona State University; 1999 ⟨http://3dk.asu.edu/archives/publication/publication.html⟩.

[45] Steinhaus H. Sur la division des corp materiels en parties. Bulletin de l Academie Polonaise des Sciences C1. III 1956;IV:801–4.

[46] Lloyd S. Least square quantization in PCMs. Bell Telephone Laboratories Paper, 1957. Published in journal much later: Lloyd SP. Least squares quantization in PCM. IEEE Transactions on Information Theory 1982;28:129–37 [special issue on Quantization].

[47] Hormann K, Lévy B, Sheffer A. Mesh parameterization: theory and practice. In: Proceedings of the ACM Siggraph 2007 course, 2007.

[48] Sheffer A, Praun E, Rose K. Mesh parameterization methods and their applications. Foundations and Trends in Computer Graphics and Vision 2006;2(2):105–71.

[49] Shamir A. A survey on mesh segmentation techniques. Computer Graphics Forum 2008;27(6):1539–56.

[50] Floater MS. Mean value coordinates. Computer Aided Geometric Design 2003;20(1):19–27.

[51] Eck M, DeRose T, Duchamp T, Hoppe H, Lounsbery M, Stuetzle W. Multiresolution analysis of arbitrary meshes. In: Proceedings of the SIGGRAPH, 1995. p. 173–82.

[52] Stephenson K. Introduction to circle packing. Cambridge: Cambridge University Press; 2005.

[53] Yang Y, Kim J, Luo F, Hu S, Gu X. Optimal surface parameterization using inverse curvature map. IEEE Transactions on Visualization and Computer Graphics 2008;14(5):1054–66.

[54] Li X, He Y, Gu X, Qin H. Curves-on-surface: a general shape comparison framework. In: Proceedings of the IEEE international conference of shape modeling, 2006. p. 352–7.

[55] Guo X, Li X, Bao Y, Gu X, Qin H. Meshless thin-shell simulation based on global conformal parameterization. IEEE Transactions on Visualization and Computer Graphics 2006;12(3):375–85.

[56] Gu X, He Y, Qin H. Manifold splines. Graph Models 2006;68(3):237–54.

[57] Yamazaki I, Natarajan V, Bai Z, Hamann B. Segmenting point sets. In: Proceedings of the IEEE international conference on shape modeling and applications, 2006. p. 4–13.