

Polynomial Splines over Hierarchical T-meshes

Jiansong Deng Falai Chen Xin Li Changqi Hu
Weihua Tong Zhouwang Yang Yuyu Feng
Department of Mathematics
University of Science and Technology of China
Hefei, Anhui, P. R. of China, 230026
dengjs|chenfl@ustc.edu.cn

October 12, 2006

Abstract

In this paper, we introduce a new type of splines — polynomial splines over hierarchical T-meshes (called *PHT-splines*) to model geometric objects. PHT-splines are a generalization of B-splines over hierarchical T-meshes. We present the detailed construction process of spline basis functions over T-meshes which satisfy some good properties with B-splines, such as nonnegativity, local support and partition of unity. As two fundamental operations, cross insertion and cross removal of PHT-splines are discussed. With the new splines, surface models can be constructed efficiently and adaptively to fit open or closed mesh models, where only linear systems of equations with a few unknowns are involved. With this approach, a NURBS surface can be efficiently simplified into a PHT-spline which dramatically reduces the superfluous control points of the NURBS surface. Furthermore, PHT-splines allow of several important types of geometric processing in a natural and efficient manner, such as segmentation of a PHT-spline into an assembly of NURBS patches, and shape simplification of PHT-splines over a coarser T-mesh. PHT-splines not only inherit many good properties of Sederberg's T-splines such as adaptivity and locality, but also exceed T-splines in several aspects except that they are only C^1 continuous. For example, PHT-splines are polynomial instead of rational; cross insertion/removal of PHT-splines is local and simple; and conversion between a NURBS surface and a PHT-spline is very effective.

Keywords T-mesh, T-splines, NURBS, local refinement, surface fitting, shape simplification

1 Introduction

A serious weakness with NURBS models is that they contain a large number of superfluous control points. T-splines [16] overcome such weakness by allowing T-junctions in the control meshes, and thus provide adaptiveness and flexibility in modeling geometric objects. In this paper, we introduce a new type of splines — polynomial splines over hierarchical T-meshes, called *PHT-splines*. A PHT-spline is a piecewise bicubic polynomial with respect to a hierarchical T-mesh. The basis functions of PHT-splines have good properties with B-splines, such as nonnegativity, local support and partition of unity. Thus PHT-splines are a generalization of B-splines over hierarchical T-meshes.

With the new splines, surface models can be constructed very efficiently and adaptively to fit open or closed meshes (with genus zero), where only linear systems of equations with a few unknowns are involved. Figure 1 illustrates an example where the female head mesh has 19,231 points and 38,388 triangular faces. Provided the mesh parameterization is given, which is not time-consuming with an up-to-date parameterization algorithm, a PHT-spline surface can be constructed to fit the mesh model in about two seconds with an ordinary personal computer. Further examples in Section 5 show that the complexity of our surface fitting algorithm is almost linear with respect to the mesh size.

The surface fitting algorithm can be easily adapted to approximate a NURBS model with a PHT-spline. Thus conversion from NURBS models to PHT-splines is very effective. As a comparison, it is reported by Zheng that

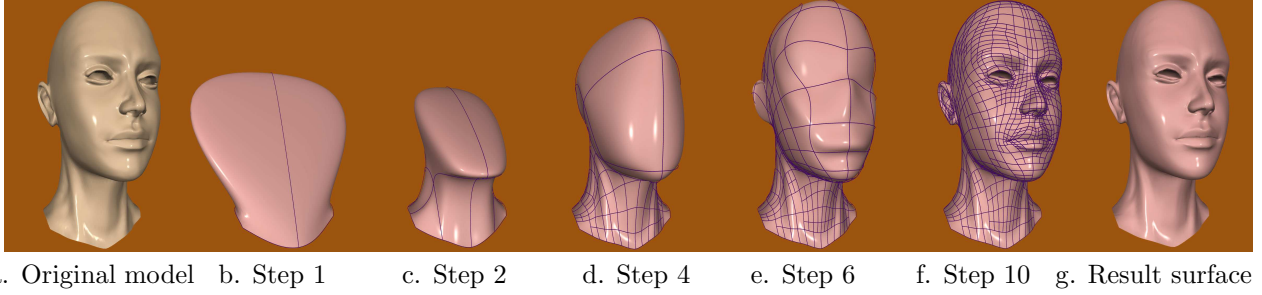


Figure 1: Fitting a mesh model (courtesy of Open3DProject) with polynomial splines over a hierarchical T-mesh. The curves on the surfaces from b to f are the images of the hierarchical T-mesh.

fitting a z-map model of about 10,000 points with T-splines takes about 30 minutes [19].

Figure 2 is another example which compares the NURBS model of an ear (from <http://www.b3dscann.com/-samples.htm>) and the fitting results of the ear model with PHT-splines. Figure 2.a shows a part of the NURBS model which has wrinkles, gaps and self-intersections, while Figure 2.b is a part of the PHT-spline model which looks much smoother and fairer than the original NURBS model.

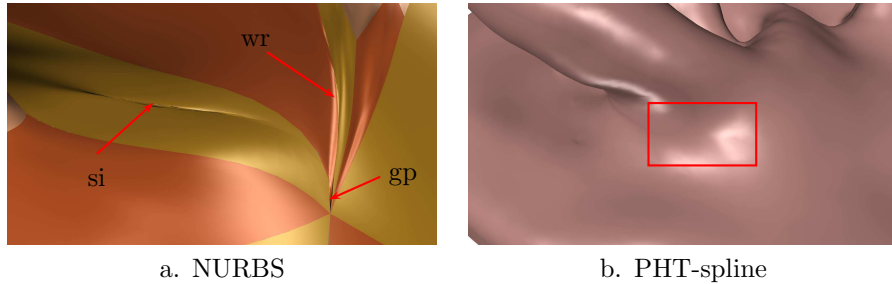


Figure 2: Comparison of fitting results (si: self-intersection, gp: gap, wr: wrinkle). (a) The original NURBS model. (b) The PHT-spline fitting model. The part in the red frame in (b) corresponds to (a).

The local refinement of PHT-splines is another important operation in PHT-spline theory. Due to the nature of hierarchical T-meshes, the local refinement of PHT-splines can be achieved by *cross insertion*, i.e., dividing a cell into four subcells with a cross. The converse process is called *cross removal*. Many important types of geometric processing such as surface fitting, conversion between NURBS and PHT-splines, segmentation of PHT-splines, shape simplification, etc. all heavily depend on cross insertion/removal operation. We will present a local and simple cross insertion/removal algorithm and apply it in several geometric processing applications such as surface fitting, shape segmentation and simplification.

1.1 Related work

In geometric modeling, the representation of surfaces is a fundamental research topic. Computer graphics and computer aided design communities prefer parametric surfaces, especially tensor product (TP for short) B-spline surfaces. However, TP B-spline surfaces suffer from the weakness that the control points must lie topologically in a rectangular grid. Thus local refinement with a TP B-spline surface is difficult. To solve this problem, [7] invented hierarchical B-splines by introducing two concepts: local refinement using an efficient representation and multi-resolution editing. Other related work on hierarchical B-splines includes [10, 13, 17]. An innovation along this direction is the invention of T-splines by Sederberg [16]. A T-spline is a point-based spline defined over a T-mesh. For every vertex, a basis function is defined. Each of the basis functions comes from some TP spline space. T-splines support many valuable operations within a consistent framework, and most importantly, they can eliminate most superfluous control points in NURBS representations and they permit local refinement. However, the local refinement of T-splines depends on the structure of the T-mesh, and its complexity might be uncertain [15]. The reason leading to this problem is that the spline function over each cell of the T-mesh is not

a polynomial, but a piecewise function. On the other hand, since the basis functions do not form a partition of unity, T-splines are rational, which leads to complicated computations in subsequent geometric operations. Sederberg [15] put forward the problem on how to construct polynomial T-splines.

1.2 Overview

The main contribution of this paper is the invention of a new type of splines—polynomial splines over hierarchical T-meshes (called PHT-splines) in geometric modelling. PHT-splines not only inherit the main properties of T-splines such as adaptivity, but also exhibit several advantages over T-splines. For example, unlike T-splines, PHT-splines are polynomial instead of rational; The local refinement algorithm of PHT-splines is local and simple; And conversion between NURBS and PHT-splines is very fast. However, conversion between NURBS and T-splines is a bottleneck of T-splines in practical applications.

The remainder of the paper is organized as follows. In Section 2, we introduce polynomial spline spaces over T-meshes and the dimension formula proved in [2]. Section 3 describes in detail the construction of the basis functions of a spline space over a hierarchical T-mesh. The properties of the basis functions are discussed and PHT-spline surfaces are introduced. Section 4 presents two important operations — cross insertion and removal in PHT-spline theory. In Section 5, we propose a surface fitting scheme to fit open and closed meshes of genus-zero with PHT-splines. In Section 6, some types of geometric processing such as shape segmentation and shape simplification are discussed. Section 7 concludes the paper with a summary and some future work.

2 Polynomial Splines over T-meshes

In this section, we briefly review the definition of T-meshes, and then introduce polynomial spline spaces over T-meshes. The dimension formula of the spline space is given.

2.1 T-meshes

A *T-mesh* is basically a rectangular grid that allows T-junctions [2, 16]. It is assumed that the end points of each grid line in the T-mesh must be on two other grid lines, and each cell or facet in the grid must be a rectangle. Figure 3 shows an example of a T-mesh. A grid point in a T-mesh is also called a *vertex* of the T-mesh. If a vertex is on the boundary grid line of the T-mesh, then is called a *boundary vertex*. Otherwise, it is called an *interior vertex*. For example, $\mathbf{b}_i$, $i = 1, \dots, 10$, in Figure 3 are boundary vertices, while all the other vertices $\mathbf{v}_i$, $i = 1, \dots, 5$, are interior vertices. Interior vertices have two types. One is crossing, for example, $\mathbf{v}_2$ in Figure 3; and the other is T-junctional, for example, $\mathbf{v}_1$ in Figure 3. They are called *crossing vertices* and *T-vertices* respectively. The line segment connecting two adjacent vertices on a grid line is called an *edge* of the T-mesh.

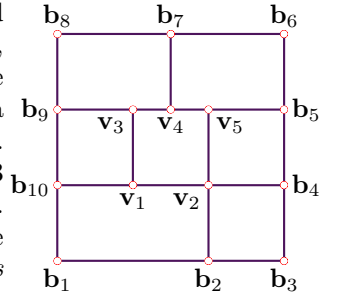


Figure 3: An example of a T-mesh

2.2 Spline spaces over T-meshes

Given a T-mesh $\mathcal{T}$, $\mathcal{F}$ denotes all the cells in $\mathcal{T}$ and Ω the region occupied by all the cells in $\mathcal{T}$. Define

$$\mathcal{S}(m, n, \alpha, \beta, \mathcal{T}) := \left\{ s(x, y) \in C^{\alpha, \beta}(\Omega) \mid s(x, y)|_{\phi} \in \mathbb{P}_{mn} \text{ for any } \phi \in \mathcal{F} \right\},$$

where $\mathbb{P}_{mn}$ is the space of all the polynomials of bi-degree (m, n) , and $C^{\alpha, \beta}(\Omega)$ is the space consisting of all the bivariate functions which are continuous in Ω with order α along x direction and with order β along y direction. It follows that $\mathcal{S}(m, n, \alpha, \beta, \mathcal{T})$ is a linear space. It is called the *spline space over the given T-mesh $\mathcal{T}$* .

In this paper, we are interested only in the spline space $\mathcal{S}(3, 3, 1, 1, \mathcal{T})$, i.e., C^1 continuous bicubic splines. According to Theorem 4.2 in [2], it follows that

$$\dim \mathcal{S}(3, 3, 1, 1, \mathcal{T}) = 4(V^b + V^+), \quad (1)$$

where V^b and V^+ represent the number of boundary vertices and interior crossing vertices, respectively. The dimension formula gives us a hint on how to construct basis functions for the spline space, i.e., each boundary vertex or interior crossing vertex associates with four basis functions. This observation will be further explored in the construction of the basis functions in the next section. Hence later on if a vertex is a boundary vertex or an interior crossing vertex, we call it a *basis vertex*. This spline space defines a function with only C^1 continuous, whereas bicubic splines are usually C^2 continuous. But this low smoothness can help us to propose an efficient surface fitting algorithm in Section 5. Furthermore, the implicit form of our new splines should be very useful in curve/surface reconstruction since a set of basis functions are available [12, 18].

3 Basis Functions of PHT-splines

Instead of considering general T-meshes, we restrict our attention to hierarchical T-meshes, since such meshes do not lose the main property — adaptivity of general T-meshes. We construct a set of basis functions over hierarchical T-meshes which exhibit many good properties with B-splines such as nonnegativity, local support and partition of unity. The basis functions directly lead to the construction of polynomial splines over hierarchical T-meshes.

3.1 Hierarchical T-meshes

A *hierarchical T-mesh* is a special type of T-mesh which has a natural level structure. It is defined in a recursive fashion. One generally starts from a TP mesh (level 0). From level k to level $k + 1$, one subdivide a cell at level k into four subcells which are cells at level $k + 1$. For simplicity, we subdivide each cell by connecting the middle points of the opposite edges with two straight lines. Figure 4 illustrates the process of generating a hierarchical T-mesh.

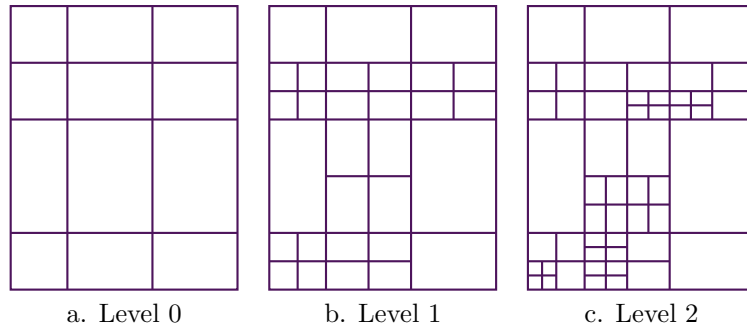


Figure 4: A hierarchical T-mesh

Hierarchical T-meshes have appeared in many research disciplines in computer science, computational mathematics, and so on. For example, adaptive finite elements [20, Chapter 15] and hierarchical B-splines [7] are defined over hierarchical T-meshes.

3.2 Construction of basis functions

Before constructing the basis functions, we point out some basic facts. Let $t_0, t_0 = t_1, t_1 < t_2, t_2 < \dots < t_{p-2}, t_{p-2} < t_{p-1}, t_{p-1} = t_p, t_p$ be the knot vector of a C^1 continuous cubic spline, where every interior knot is with multiplicity two. For any interior knot t_i , only two B-spline basis functions have support $[t_{i-1}, t_{i+1}]$. These two basis functions are associated with knot vectors $(t_{i-1}, t_{i-1}, t_i, t_i, t_{i+1})$ and $(t_{i-1}, t_i, t_i, t_{i+1}, t_{i+1})$ respectively. Except for the two basis functions, all the other B-spline basis functions and their derivatives vanish at t_i . Generalize this fact to C^1 continuous bicubic TP B-splines, for any interior knot vertex (s_i, t_j) , where s_i and t_j are the knots in two directions respectively, only four B-spline basis functions have support $[s_{i-1}, s_{i+1}] \times [t_{j-1}, t_{j+1}]$.

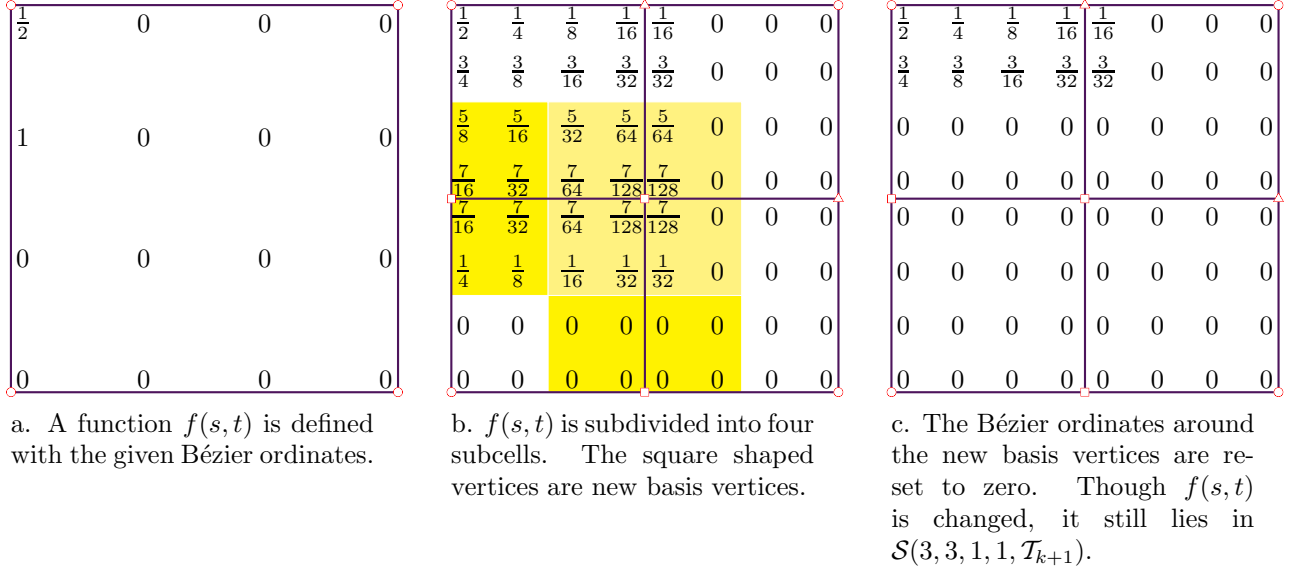


Figure 5: Modification of a basis function

We say these four basis functions are *associated with* the vertex (s_i, t_j) . The function values, the first order partial derivatives and the mixed derivatives of all the other B-spline basis functions vanish at (s_i, t_j) .

The basic strategy to construct the basis functions for the spline space $\mathcal{S}(3, 3, 1, 1, \mathcal{T})$ is level by level. Denote the T-mesh at level k by $\mathcal{T}_k$. For level 0, the standard TP B-spline basis functions are taken to be the basis functions over the initial TP mesh $\mathcal{T}_0$, where every basis vertex associates with four basis functions according to the analysis in the beginning of the subsection. Obviously, these basis functions satisfy the good properties such as nonnegativity, local support and partition of unity. Now suppose the basis functions $\{b_j^k(u, v)\}$, $j = 1, \dots, d_k$ at level k have been constructed, and these basis functions satisfy the good properties, i.e., $\sum_{j=1}^{d_k} b_j^k(u, v) \equiv 1$, $b_j^k(u, v) \geq 0$, and for each j , $b_j^k(u, v)$ has local support. Then the basis functions at level $k + 1$ consist of two parts. The first part comes from the modification of the basis functions at level k . The second part comes from the basis functions associated with the new basis vertices at level $k + 1$. The basis functions generated at level $k + 1$ keep the same good properties as the previous level.

Now we describe the details of constructing the basis functions at level $k + 1$. For brevity, a basis function is represented by specifying its 16 Bézier ordinates in every cell within the support of the basis function.

3.2.1 Modification of the basis functions at level k

Suppose, among all the cells at level k , the cells θ_i^k , $i = 1, \dots, C_k$, are subdivided. For each j , if the basis function $b_j^k(u, v)$ does not vanish in some cells of $\{\theta_i^k\}$, then subdivide $b_j^k(u, v)$ into these cells at level $k + 1$ according to formula (14.15) in [4]. See Figure 5.a and Figure 5.b on the next page for an example. It should be noted that the function $b_j^k(u, v)$ has not changed, but is now defined over the mesh $\mathcal{T}_{k+1}$.

From $\mathcal{T}_k$ to $\mathcal{T}_{k+1}$, some new basis vertices appear. Denote them as ξ_i^{k+1} , $i = 1, \dots, V_{k+1}$. In each cell at level $k + 1$, the 16 Bézier ordinates are divided into four parts. Each part is associated with a cell corner vertex as shown in Figure 6. Then all the basis functions $\{b_j^k(u, v)\}$ at level k are modified to $\{\bar{b}_j^k(u, v)\}$ in the following fashion: for each j , reset all the associated Bézier ordinates with the new basis vertices to zero. See Figure 5(c) for an example, where there are three new vertices.

One can show that $\{\bar{b}_j^k(u, v)\}$ are in $\mathcal{S}(3, 3, 1, 1, \mathcal{T}_{k+1})$.

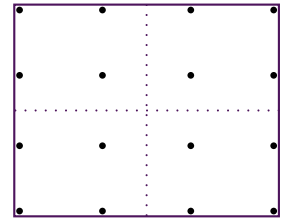


Figure 6: The Bézier ordinates are associated with the four corner vertices

3.2.2 Adding new basis functions at level $k + 1$

For each new basis vertex (s_i, t_i) , its neighboring cells must lie in a position as shown in Figure 7. Then the four basis functions associated with (s_i, t_i) are defined to be $M_{ik}^3(s)N_{il}^3(t)$, $k, l = 1, 2$, where $M_{i1}^3(s)$, $M_{i2}^3(s)$, $N_{i1}^3(t)$, and $N_{i2}^3(t)$ are the cubic B-spline basis functions associated with the knot vectors $(s_{i-1}, s_{i-1}, s_i, s_i, s_{i+1})$, $(s_{i-1}, s_i, s_i, s_{i+1}, s_{i+1})$, $(t_{i-1}, t_{i-1}, t_i, t_i, t_{i+1})$ and $(t_{i-1}, t_i, t_i, t_{i+1}, t_{i+1})$, respectively. If (s_i, t_i) is a boundary vertex, then either $s_{i-1} = s_i$, or $s_{i+1} = s_i$, or $t_{i-1} = t_i$ or $t_{i+1} = t_i$.

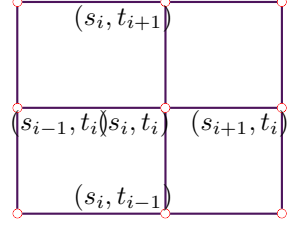


Figure 7: A new basis vertex and its neighboring cells

It is easy to show that these four basis functions are in $\mathcal{S}(3, 3, 1, 1, \mathcal{T}_{k+1})$ and they have the same support $[s_{i-1}, s_{i+1}] \times [t_{i-1}, t_{i+1}]$. Denote all the basis functions associated with the new basis vertices as $\{\tilde{b}_j^{k+1}(u, v)\}_{j=1}^{4V_{k+1}}$.

From the previous construction process, it is easy to see that $\{\tilde{b}_j^{k+1}(u, v)\}_{j=1}^{4V_{k+1}}$ and $\{\bar{b}_j^k(u, v)\}_{j=1}^{d_k}$ are linearly independent. Since $d_{k+1} = d_k + 4V_{k+1}$ according to the dimension formula (1), these functions form a basis for the spline space $\mathcal{S}(3, 3, 1, 1, \mathcal{T}_{k+1})$, and are denoted as $\{b_j^{k+1}(u, v)\}_{j=1}^{d_{k+1}}$.

3.3 Properties of the basis functions

The basis functions constructed in the last subsection satisfy the following properties:

- **Nonnegativity.** This follows from the fact that each basis function has nonnegative Bézier ordinates.
- **Partition of unity.** Define

$$s_1(u, v) = \sum_{j=1}^{4V_{k+1}} \tilde{b}_j^{k+1}(u, v), \quad s_2(u, v) = \sum_{j=1}^{d_k} \bar{b}_j^k(u, v).$$

Check the Bézier ordinates of $s_1(u, v)$ and $s_2(u, v)$ in every cell. For $s_1(u, v)$, the Bézier ordinates associated with the new basis vertices are one, and all the others are zero; for $s_2(u, v)$, the Bézier ordinates associated with the new basis vertices are zero, and all the others are one. Hence it follows that $s_1(u, v) + s_2(u, v) \equiv 1$.

- **Local support.** From the construction of the basis functions, the support for each basis function is inside a rectangle, and its size decreases level by level. In fact, the rectangular support for each basis function has the following properties: 1. It is bounded by edges in the mesh. 2. Suppose the basis vertex associating with the basis function is $\mathbf{v}$. If $\mathbf{v}$ is an interior vertex, then it lies in the interior of the rectangle. Otherwise it lies on the boundary of the rectangle. 3. The rectangle has the minimal size among all the rectangles satisfying the previous two properties.

Next we study another important property of the basis functions. For any basis function $b(u, v)$, we define a *geometric information* (the function value, the first order partial derivatives and the mixed partial derivative) operator by

$$\mathcal{L}b(u, v) = (b(u, v), b_u(u, v), b_v(u, v), b_{uv}(u, v)). \quad (2)$$

It follows that $\mathcal{L}(\cdot)$ is a linear operator and satisfies the following properties:

- For any basis function $b(u, v)$ and any basis vertex (u_0, v_0) in $\mathcal{T}$, $\mathcal{L}b(u_0, v_0) = \mathbf{0}$ holds for all the basis functions, except for the four basis functions associated with the basis vertex (u_0, v_0) .
- For a basis vertex (u_0, v_0) which appears in the hierarchical T-mesh since level k_0 , $\mathcal{L}b_j^k(u_0, v_0)$ remains unchanged for any $k \geq k_0$. Here $b_j^k(u, v)$ is a basis function constructed at level k .

This is true since the Bézier ordinate modification happens only around the new basis vertices. These observations are important for the surface fitting algorithm developed in Section 5.

3.4 PHT-spline surfaces

Let $\mathcal{T}$ be a hierarchical T-mesh, and $\{b_j(u, v)\}$, $j = 1, \dots, d$ be the basis functions constructed in Subsection 3.2. Then the polynomial spline surface over $\mathcal{T}$ (called *PHT-spline surface*) is defined by

$$\mathbf{S}(u, v) = \sum_{j=1}^d \mathbf{C}_j b_j(u, v), \quad (u, v) \in [0, 1] \times [0, 1], \quad (3)$$

where $\mathbf{C}_j$ are control points in a column vector form. PHT-spline surfaces have similar properties with B-spline surfaces such as convex-hull property, affine invariant, local support, etc..

To efficiently manipulate and evaluate a PHT-spline surface, one should maintain the Bézier representation of the PHT-spline surface in every cell, since the basis functions are represented in Bézier forms in every cell. The geometric processing algorithms in later sections rely on such representation.

Since the operator $\mathcal{L}(\cdot)$ in Equation (2) is linear, for any fixed basis vertex (u_0, v_0) with which four basis functions with indices j_1, j_2, j_3, j_4 are associated, one has

$$\mathcal{L}\mathbf{S}(u_0, v_0) = \sum_{j=1}^d \mathbf{C}_j \mathcal{L}b_j(u_0, v_0) = \sum_{i=1}^4 \mathbf{C}_{j_i} \mathcal{L}b_{j_i}(u_0, v_0) = \mathbf{C} \cdot \mathbf{b},$$

where $\mathbf{C} = (\mathbf{C}_{j_1}, \mathbf{C}_{j_2}, \mathbf{C}_{j_3}, \mathbf{C}_{j_4})$ is a 3×4 matrix, and $\mathbf{b} = (\mathcal{L}b_{j_1}(u_0, v_0), \mathcal{L}b_{j_2}(u_0, v_0), \mathcal{L}b_{j_3}(u_0, v_0), \mathcal{L}b_{j_4}(u_0, v_0))$ is a 4×4 matrix.

Suppose the measurement of the T-mesh $\mathcal{T}$ around the basis vertex (u_0, v_0) , when (u_0, v_0) just appears in $\mathcal{T}$, is as shown in Figure 8. Here the black dots illustrate the locations of Bézier ordinates. Then it follows that

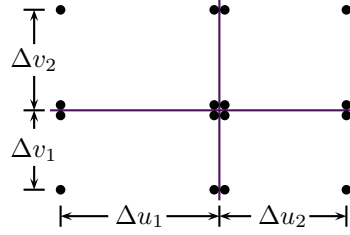


Figure 8: The size of the T-mesh $\mathcal{T}$ around a new basis vertex

$$\mathbf{b} = \begin{pmatrix} (1-\lambda)(1-\mu) & -\alpha(1-\mu) & -\beta(1-\lambda) & \alpha\beta \\ \lambda(1-\mu) & \alpha(1-\mu) & -\beta\lambda & -\alpha\beta \\ (1-\lambda)\mu & -\alpha\mu & \beta(1-\lambda) & -\alpha\beta \\ \lambda\mu & \alpha\mu & \beta\lambda & \alpha\beta \end{pmatrix},$$

where

$$\alpha = \frac{1}{\Delta u_1 + \Delta u_2}, \beta = \frac{1}{\Delta v_1 + \Delta v_2}, \lambda = \alpha \Delta u_1, \mu = \beta \Delta v_1.$$

The matrix $\mathbf{b}$ is invertible and one has

$$\mathbf{C} = \mathcal{L}\mathbf{S}(u_0, v_0) \cdot \mathbf{b}^{-1}. \quad (4)$$

The above formula reflects the relationship between the control points and the geometric information at the basis vertices of a spline surface over a (hierarchical) T-mesh. An efficient surface fitting scheme will be proposed in Section 5 based on Equation (4).

4 Cross Insertion and Removal

Knot insertion and removal operations play an important role in curve and surface modeling with standard B-splines. For PHT-splines, cross insertion and removal operations play the same role as knot insertion and removal with B-splines. Many geometric processing applications such as surface fitting, NURBS simplification (Section 5), segmentation to TP surfaces, and shape simplification (Section 6) all heavily depend on these two operations.

4.1 Cross insertion

Suppose we are given a PHT-spline surface $\mathbf{S}(u, v)$ over a hierarchical T-mesh $\mathcal{T}$ as defined in (3), and a cell θ from $\mathcal{T}$ is subdivided into four subcells by inserting a cross. Denote the new T-mesh after subdivision by $\tilde{\mathcal{T}}$. We want to find the representation of $\mathbf{S}(u, v)$ over the new T-mesh $\tilde{\mathcal{T}}$.

Suppose the level of the cell θ in $\mathcal{T}$ is k_1 , and the maximal level of the adjacent cells of θ is k_2 . If $k_1 \geq k_2 - 1$, then the old control points $\mathbf{C}_j$ are unchanged, and for each new basis vertex in $\tilde{\mathcal{T}}$, the associated basis functions can be constructed as described in Subsection 3.2, and the corresponding control points are determined according to Equation (4).

If $k_1 < k_2 - 1$, the process is a bit more complicated. We need to remove temporarily the cells adjacent to θ whose levels are larger than $k_1 + 1$, and then perform cross insertion for θ . After this process, we perform cross insertion algorithm to recover those removed cells level by level.

It should be noted that all the operations take place in θ and its adjacent cells, hence the operation is local. Furthermore, $k_1 \geq k_2 - 1$ is the most common case in geometric processing applications.

4.2 Cross removal

Cross removal is the converse operation of cross insertion. For simplicity, we require that the mesh after cross removal is still a hierarchical T-mesh. Like knot removal, cross removal can be achieved only approximately in general.

Let $\mathbf{S}(u, v)$ be a PHT-spline surface over a hierarchical T-mesh $\mathcal{T}$, and $\tilde{\mathcal{T}}$ be the mesh after removing a cross from $\mathcal{T}$. Let θ be the cell in $\tilde{\mathcal{T}}$ from which the cross is removed. Our idea is to find a PHT-spline surface $\tilde{\mathbf{S}}(u, v)$ defined over $\tilde{\mathcal{T}}$ such that $\mathbf{S}(u, v)$ and $\tilde{\mathbf{S}}(u, v)$ share the same geometric information at the basis vertices of $\tilde{\mathcal{T}}$. This requirement guarantees that $\tilde{\mathbf{S}}(u, v) \equiv \mathbf{S}(u, v)$ whenever $\mathbf{S}(u, v)$ is obtained from $\tilde{\mathbf{S}}(u, v)$ by inserting the cross in θ .

The basic approach is as follows. First, eliminate those terms in $\mathbf{S}(u, v)$ whose corresponding basis functions are associated with the basis vertices in $\mathcal{T}$ but not in $\tilde{\mathcal{T}}$ to obtain $\tilde{\mathbf{S}}(u, v)$. Second, modify the basis function $b_i(u, v)$ in each term of $\tilde{\mathbf{S}}(u, v)$ to $\tilde{b}_i(u, v)$ while keeping the corresponding control points unchanged. The requirement is that $\{\tilde{b}_i(u, v)\}$ form a basis for the spline space $\mathcal{S}(3, 3, 1, 1, \tilde{\mathcal{T}})$, and that $b_i(u, v)$ and $\tilde{b}_i(u, v)$ share the same geometric information at the basis vertices of $\tilde{\mathcal{T}}$. Over the cell θ as shown in Figure 9, the Bézier representation of $\tilde{b}_i(u, v)$ can be easily obtained from the 16 black Bézier ordinates which come from $b_i(u, v)$. In the adjacent cells of θ , the modification is similar. Note that only those basis functions need to be modified which do not vanish over θ or its adjacent cells.

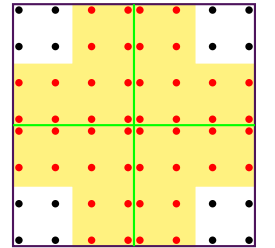


Figure 9: Cross removal

5 Surface Fitting

Surface fitting is a widely investigated problem and there is a wealth of literature focusing on it. The reader is referred to [1, 8] for surveys. In this paper, we are going to present a very efficient scheme to fit a mesh model with a PHT-spline. The scheme directly leads to an efficient method to convert a NURBS surface into a

PHT-spline, and a NURBS model into a few PHT-spline patches which are C^1 continuous globally. It should be noted that if one tries to solve the fitting problem by the least-squares method, a system of linear equations of huge size would be involved.

5.1 Fitting open meshes

Suppose we are given an open mesh model with vertices $\mathbf{P}_i$, $i = 1, 2, \dots, N$ in 3D space, and their corresponding parameter values (s_i, t_i) , $i = 1, 2, \dots, N$ obtained from some parameterization of the mesh (The reader is referred to [5, 6] for a survey about mesh parameterization). The parameter domain is assumed to be $[0, 1] \times [0, 1]$.

According to Equation (4), in order to construct a PHT-spline to fit the given mesh, one only has to estimate $\mathcal{LS}(u, v)$ at every basis vertex. This can be done in the following two steps. First, compute $\mathcal{LS}(s_i, t_i)$. To do so, we find a set of points in the neighborhood $\mathcal{N}(\mathbf{P}_i)$ of $\mathbf{P}_i$, and then fit the set of points with a quadratic surface patch. The required information can be obtained by evaluating the surface patch at (s_i, t_i) . Next, the geometric information $\mathcal{LS}(u, v)$ at every basis vertex can be calculated by a simple linear interpolation to $\mathcal{LS}(s_i, t_i)$, $i = 1, 2, \dots, N$.

The surface fitting scheme repeats the following two steps until the fitting error in each cell is less than some tolerance ε :

1. Keep unchanged the control points associating with the old basis functions, and compute the control points for the new basis functions (in the beginning, every basis function is new) according to Equation (4);
2. Determine the cells whose fitting errors are greater than ε , and then subdivide these cells into subcells to form a new mesh, and construct basis functions for the new mesh.

Here the fitting error over a cell θ is defined to be

$$\max_{(u,v) \in \theta} \|\mathbf{P}(u, v) - \mathbf{S}(u, v)\|,$$

where $\mathbf{P}(u, v)$ is the (piecewise linear) parametric equation of the mesh. In practice, the fitting error is calculated as the maximum of $\|\mathbf{P}(u_i, v_i) - \mathbf{S}(u_i, v_i)\|$ for some sample points (u_i, v_i) in θ .

Three examples are provided to illustrate the above surface fitting scheme in Figures 1 and 10. In all these examples, the initial T-meshes are the square $[0, 1] \times [0, 1]$, and the parameterizations are obtained with the discrete harmonic mappings proposed by [3]. The tolerance of the fitting error is $\varepsilon = 0.1\%$. Table 1 shows the computational time for the three examples, where CP stands for control points. The computation is performed on a PC with Pentium 4 CPU, 3.20GHz and 1.0GB RAM. From the examples, we can conclude that the computational complexity is almost linear in the mesh size.

Table 1: Computation time for fitting open meshes

Mesh	#points	#levels	# CP	Time(sec.)
Female head	19,231	10	4,432	2.31
Igea Artifact	44,992	9	17,744	10.73
Gargoyle	91,279	17	35,764	33.23

5.2 Fitting closed meshes

Many geometric models are described by closed meshes of genus-zero. For such models, the unit sphere is the most natural parameterization domain. In this section, we propose a method to fit a closed mesh model with a family of PHT-splines which are pieced together with C^1 continuity. The method relies on the spherical parameterization of closed meshes. The reader is referred to [9, 11, 14] for some references on this topic.

The basic idea is as follows. Find an inscribed cube for the sphere of the parameterization domain, and project the six faces of the cube onto the sphere by the central projection from the center of the sphere. The sphere is divided into six parts corresponding to the six faces, and correspondingly the mesh model is divided into six

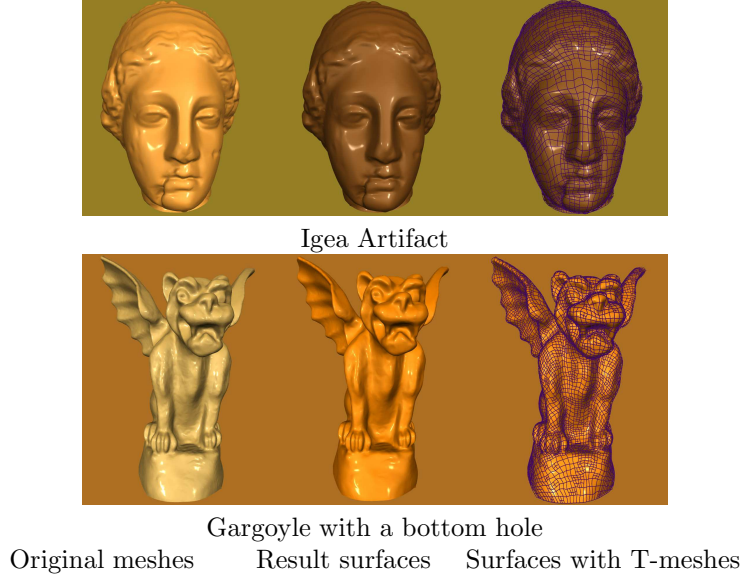


Figure 10: Two examples of fitting open meshes

parts by the spherical parameterization. This induces a planar parameterization for each part of the mesh model. Fitting each part of the mesh by a PHT-spline results in a surface model for the closed mesh.

To guarantee the smoothness between two adjacent PHT-splines, one should ensure that the two corresponding T-mesh domains share the same boundary vertices along their common edges when subdividing cells in the fitting process, and that the geometric information at boundary basis vertices is the average information of the two parts along the common edges. In this fashion, the final PHT-spline surface is C^1 continuity except at the corners, where G^1 smoothness is obtained.

Figure 11 illustrates two examples for fitting closed meshes with PHT-splines, and Table 2 shows the execution time. Again the computational complexity is linear in the mesh size.

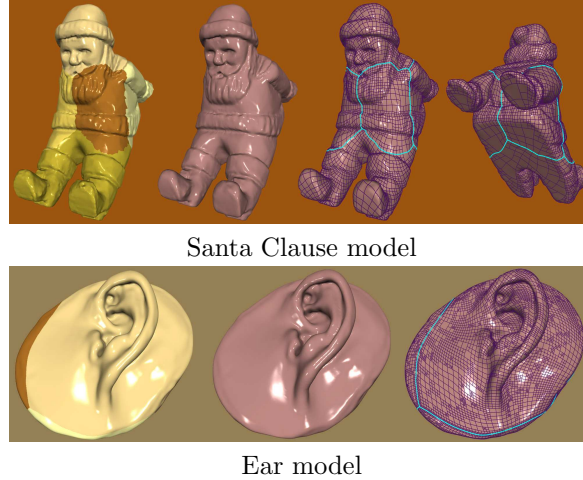


Figure 11: Two examples of fitting closed meshes (Courtesy of Berding 3D Scanning (<http://www.b3dscann.com/-samples.htm>))

When modeling geometric objects with NURBS, it is a hard problem to smoothly piece the NURBS patches together. PHT-splines demonstrate a promising approach to solve the problem. Furthermore, the number of PHT-spline patches is much fewer (compare 914 NURBS patches with 6 PHT-spline patches for the ear model!).

Table 2: Computation time for fitting closed meshes

Mesh	#points	#levels	#CP	Time(sec.)
Santa Claus	46,048	11	23,552	8.52
Gargoyle	100,002	11	50,652	23.22
Ear	1,407,202	9	58,876	304.25

5.3 NURBS approximation

The surface fitting scheme provided in the previous subsections can be easily adapted to converting a NURBS surface into a PHT-spline surface. Since the local geometric information can be directly obtained from the NURBS surface, the algorithm is much faster. Figure 12 demonstrates an example. The conversion process takes 0.875 seconds from a to b/c, and 0.125 seconds from a to d/e.

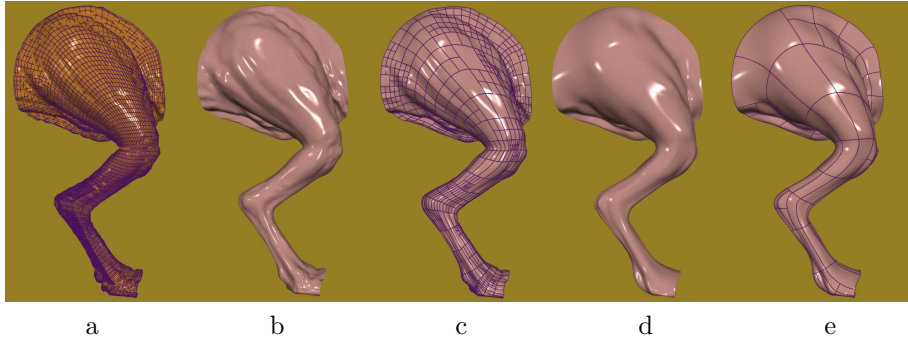


Figure 12: NURBS approximation. a. The original NURBS surface with 8835 control points. b and c: A PHT-spline approximation with 4404 control points. $\varepsilon = 0.35\%$. e and f: A PHT-spline approximation with 468 control points. $\varepsilon = 1.1\%$.

6 Geometric Processing

PHT-splines allow of many types of geometric processing. For example, in the same approach as in [16], two or more PHT-splines can be stitched together. The surface fitting scheme for closed meshes in Subsection 5.2 provides another approach for merging two PHT-spline surfaces. On the other hand, similar to [7], the local details can be added onto the shape of the PHT-spline surfaces by mesh refinements.

In this section, more types of geometric processing on PHT-splines will be explored briefly. They include segmentation of a PHT-spline into an assembly of TP B-splines and shape simplification.

6.1 Segmentation of PHT-splines

Given a PHT-spline defined over a hierarchical T-mesh, the PHT-spline can be easily converted into a family of TP B-spline patches. This process is called *segmentation*. Segmentation makes it easy for PHT-splines to be conveniently imported into the current surface modeling system.

Given a nonnegative integer σ and a hierarchical T-mesh $\mathcal{T}$, one can organize the cells in $\mathcal{T}$ into an assembly of rectangles such that the level differences of the cells in each rectangle are not greater than σ . This organization is usually not unique, but any solution leads to a segmentation of the PHT-spline surface in the following fashion: in each rectangle, perform some cross insertions such that the T-mesh in the rectangle becomes a TP mesh, and then form a TP B-spline surface over the rectangle from the Bézier representation of the PHT-spline surface in each cell. See Figure 13 for an example.

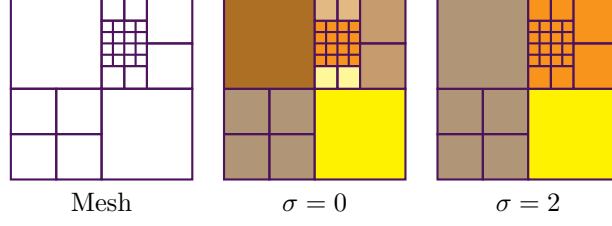


Figure 13: Examples of hierarchical T-mesh segmentation

Figure 14 shows the segmentation results of the PHT-spline surface from Figure 1.g. Here the control nets of the TP B-spline patches are displayed in brown, and the patch boundaries are in light blue. It takes 1.44, 0.33 and 0.14 seconds to generate the results in Figure 14.a, b and c, respectively.

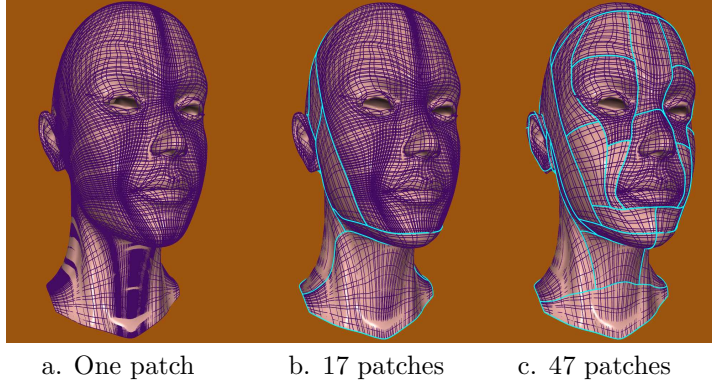


Figure 14: Segmentation of PHT-spline surfaces

6.2 Shape simplification

Shape simplification of a PHT-spline surface over a T-mesh $\mathcal{T}$ is to find a coarser mesh $\tilde{\mathcal{T}}$ (obtained from $\mathcal{T}$ by removing some crosses) of $\mathcal{T}$ such that the difference between the original PHT-spline surface and the new PHT-spline surface defined over $\tilde{\mathcal{T}}$ is smaller than some given tolerance, and the two surfaces share the same geometric information at the same basis vertices. The result of shape simplification can be applied in level-of-detail operation in computer graphics.

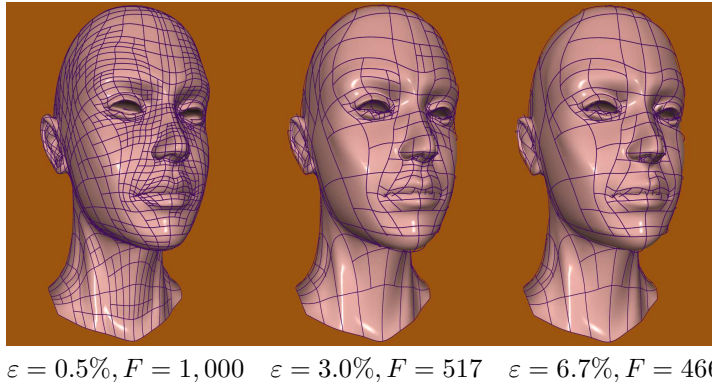


Figure 15: Shape simplification

Given a PHT-spline surface $\mathbf{S}(u, v)$ over a hierarchical T-mesh $\mathcal{T}$ with level k_0 and a tolerance ϵ , the procedure of shape simplification can be described as follows:

- Set $k = k_0 - 1$ and $\tilde{\mathbf{S}}(u, v) = \mathbf{S}(u, v)$.
- For each cell θ in $\mathcal{T}$ at level k which has a cross inside θ (and each subcell of θ does not have a cross inside itself), perform cross removal operation for $\tilde{\mathbf{S}}(u, v)$. Denote the PHT-spline surface after cross removal by $\tilde{\mathbf{S}}(u, v)$, and the mesh after cross removal by $\tilde{\mathcal{T}}$. If $\|\mathbf{S}(u, v) - \tilde{\mathbf{S}}(u, v)\| < \epsilon$ over θ and its adjacent cells, then replace $\mathbf{S}(u, v)$ by $\tilde{\mathbf{S}}(u, v)$, and $\mathcal{T}$ by $\tilde{\mathcal{T}}$.
- Set $k = k - 1$. If $k \geq 0$, go to the previous step.

To speed up computation, we use the Bézier representations for the PHT-spline surfaces. The Bézier representation for $\tilde{\mathbf{S}}(u, v)$ over $\tilde{\mathcal{T}}$ can be easily obtained from the Bézier representation of $\mathbf{S}(u, v)$ by the smoothness conditions. The difference between $\mathbf{S}(u, v)$ and $\tilde{\mathbf{S}}(u, v)$ can be measured by the control points of the two surfaces. Figure 15 shows the surface simplification result of the female head model, where the original mesh has 1,528 cells. The simplification operation can be accomplished in real-time.

7 Conclusions and Future Work

The paper presents the theoretical foundation for PHT-splines — polynomial splines over hierarchical T-meshes. The construction of basis functions is simple and straightforward to implement. The cross insertion and removal operations are discussed as well. With PHT-splines, an efficient and adaptive surface fitting scheme are presented. Several types of geometric processing are studied, such as segmentation of a PHT-spline surface into a family of NURBS patches, and shape simplification of PHT-splines.

PHT-splines can model a geometric object with a single PHT-spline patch or several PHT-spline patches which are C^1 continuous globally. On the other hand, it is a very difficult problem to construct geometric models with NURBS patches to guarantee global smoothness and fairness.

PHT-splines inherit many good properties of T-splines such as adaptivity and locality. They also exhibit several advantages over T-splines. For example, unlike T-splines, PHT-splines are polynomial instead of rational. Thus geometric computation with PHT-splines is simpler and less costly. The local refinement algorithm of PHT-splines is local and simple while the complexity of knot insertion with T-splines may be uncertain. Finally, conversion between NURBS and PHT-splines is very fast, while conversion between NURBS and T-splines is a bottleneck of T-splines in practical applications.

One main drawback of our new splines is that they are only C^1 continuous. However, having a set of basis functions should be helpful for theoretic analysis and many other applications. For example, in fitting scattered data points with implicit splines [12, 18], a set of basis functions are a necessity. Thus the implicit form of our new splines (i.e., the zero level-set of PHT-splines) can be applied in adaptive curve/surface fitting of scattered data. We will explore this possibility in future papers.

There are still many interesting research problems about PHT-splines for further research. For example, how to construct the basis functions for spline surfaces over general T-meshes? How to generalize the PHT-splines to higher degree surfaces with arbitrary order of smoothness? Especially how to construct C^2 continuous bicubic spline surfaces over T-meshes? Is there a more efficient cross insertion algorithm whenever $k_1 < k_2 - 1$? How to fit a model with complicated topology with PHT-splines? There are problems worthy of further study.

References

- [1] P. Chinvate and A. Jablolkow. Review of surface representation and fitting for reverse engineering. *Computer Integrated Manufacturing Systems*, 8(3):193–204, 1995.
- [2] Jiansong Deng, Falai Chen, and Yuyu Feng. Dimensions of spline spaces over T-meshes. *Journal of Computational and Applied Mathematics*, 194(2):267–283, 2006.

- [3] Matthias Eck, Tony DeRose, Tom Duchamp, Hugues Hoppe, Michael Lounsbery, and Werner Stuetzle. Multiresolution analysis of arbitrary meshes. In *SIGGRAPH'95: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 173–182, New York, NY, USA, 1995. ACM Press.
- [4] Gerald Farin. *Curves and Surfaces for CAGD — A Practical Guide*. Morgan Kaufmann Publishers, 5th edition, 2002.
- [5] Michael S. Floater and Kai Hormann. Parameterization of triangulations and unorganized points. In Armin Iske, Ewald Quak, and Michael S. Floater, editors, *Tutorials on Multiresolution in Geometric Modelling*, pages 287–316. Springer-Verlag, 2002.
- [6] Michael S. Floater and Kai Hormann. Surface parameterization: A tutorial and survey. In N. A. Dodgson, M. S. Floater, and M. A. Sabin, editors, *Advances in Multiresolution for Geometric Modelling*, pages 157–186. Springer-Verlag, Heidelberg, 2004.
- [7] David Forsey and Richard H. Bartels. Hierarchical B-spline refinement. *Computer Graphics*, 22(4):205–212, 1988.
- [8] R. Franke and G. M. Nielson. Scattered data interpolation and applications: A tutorial and survey. In H. Hagen and D. Roller, editors, *Geometric Modeling*, pages 131–161. Springer, Berlin, 1991.
- [9] Craig Gotsman, Xianfeng Gu, and Alla Sheffer. Fundamentals of spherical parameterization for 3D meshes. *ACM Transactions on Graphics*, 22(3):358–363, 2003.
- [10] E. Grinspun, P. Krysl, and P. Schröder. Charms: A simple framework for adaptive simulation. *ACM Transactions on Graphics*, 21(3):281–290, 2002.
- [11] X. Gu, Y. Wang, T. F. Chan, P. M. Thompson, and S-T Yau. Genus zero surface conformal mapping and its application to brain surface mapping. *IEEE Transaction on Medical Imaging*, (7):949–958, 2004.
- [12] Bert Jüttler. Least-squares fitting of algebraic spline curves via normal vector estimation. In *Proceedings of the 9th IMA Conference on the Mathematics of Surfaces*, pages 263–280. Springer-Verlag, 2000.
- [13] R. Kraft. Adaptive and linearly independent multilevel B-splines. In A. L. Méhauté, C. Rabut, and L. L. Schumaker, editors, *Surface Fitting and Multiresolution Methods*, pages 209–218. Vanderbilt University Press, Nashville, 1997.
- [14] Emil Praun and Hugues Hoppe. Spherical parametrization and remeshing. *ACM Transaction on Graphics*, 22(3):340–349, 2003.
- [15] Thomas W. Sederberg, David L. Cardon, G. Thomas Finnigan, Nicholas S. North, Jianmin Zheng, and Tom Lyche. T-spline simplification and local refinement. *ACM Transactions on Graphics*, 23(3):276–283, 2004.
- [16] Thomas W. Sederberg, Jianmin Zheng, Almaz Bakenov, and Ahmad Nasri. T-splines and T-NURCCs. *ACM Transactions on Graphics*, 22(3):161–172, 2003.
- [17] F. Weller and H. Hagen. Tensor product spline spaces with knot segments. In M. Daehlen, T. Lyche, and L. L. Schumaker, editors, *Mathematical Methods for Curves and Surfaces*, pages 563–572. Vanderbilt University Press, Nashville, 1995.
- [18] Zhouwang Yang, Jiansong Deng, and Falai Chen. Fitting point clouds with active implicit B-spline curves. *The Visual Computer (Special Issue for Pacific Graphics 2005)*, 21(8–10):831–839, 2005.
- [19] Jianmin Zheng, Yimin Wang, and Hock Soon Seah. Adaptive T-spline surface fitting to z-map models. In *GRAPHITE '05: Proceedings of the 3rd international conference on computer graphics and interactive techniques in Australasia and South East Asia*, pages 405–411, New York, NY, USA, 2005. ACM Press.
- [20] O. C. Zienkiewicz and R. L. Taylor. *The Finite Element Method, Volume 1: The Basis*. Elsevier (Singapore) Pte Ltd., 5th edition, 2004.