Computer Aided Geometric Design 38 (2015) 1–23

Contents lists available at ScienceDirect

Computer Aided Geometric Design

www.elsevier.com/locate/cagd

Bicubic hierarchical B-splines: Dimensions, completeness, and bases ☆

Chao Zeng, Fang Deng, Jiansong Deng *

School of Mathematical Sciences, University of Science and Technology of China, Hefei, Anhui, PR China

ARTICLE INFO

Article history:
Received 18 December 2014
Received in revised form 24 July 2015
Accepted 24 July 2015
Available online 30 July 2015

Keywords:
Spline spaces over T-meshes
Completeness
Basis

ABSTRACT

In this paper, we discuss the bicubic C^2 spline spaces over hierarchical T-meshes in detail. The topological explanation of the dimension formula is further explored. We provide three necessary conditions for the completeness of the spline spaces. Based on the three conditions, the rule of the refinement of the hierarchical T-mesh is given, and a basis is constructed. The basis functions are linearly independent and complete, which provides a solution for the defects in the former literature.

© 2015 Elsevier B.V. All rights reserved.

1. Introduction

Splines, which can be adaptively locally refined, are an effective tool for both surface modeling and isogeometric analysis. Because of the special structure of T-mesh, adaptive local refinement is possible for splines defined on this type of mesh. The study of this type of splines appeared early. In 1988, hierarchical B-splines (Forsey and Bartels, 1988) were introduced by Forsey and Bartels. With the advent of isogeometric analysis, the study of this type of splines has increased.

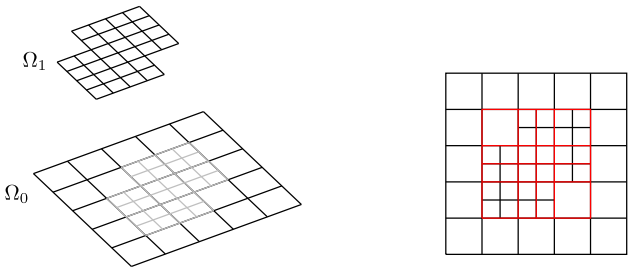
In 2003, Sederberg et al. (2003) introduced T-splines. T-splines are a useful tool in surface modeling (Sederberg et al., 2004, 2008), and they have been used in isogeometric analysis (Bazilevs et al., 2010; Dorfel et al., 2010). However, in 2010, Buffa et al. (2010) discovered that the blending functions of T-splines are linearly dependent on some special T-meshes. Li et al. (2012), Li and Scott (2014) introduced analysis-suitable T-Splines (ASTS) to solve this problem. However, because of the refinement rule of T-splines, they act badly in the locality of the refinement (Dorfel et al., 2010). Therefore, researchers have begun to pay attention to other splines defined on T-meshes for isogeometric analysis.

PHT-splines (Deng et al., 2008), which usually refer to bicubic C^1 polynomial splines defined on hierarchical T-meshes, have been used both in surface modeling (Li et al., 2007; Wang et al., 2011a) and isogeometric analysis (Nguyen-Thanh et al., 2011a, 2011b; Wang et al., 2011b). The main drawback of PHT-splines is that they are only C^1 continuous, which restricts their applications somewhere. For example, in geometric modeling, G^2 continuity is required usually. In 2013, a new type of splines, LR-splines (Dokken et al., 2013), was constructed, and they have been used in IGA (Johannessen et al., 2014). However, their linear independence cannot be guaranteed. In Dokken et al. (2013), the authors provide some strategies to solve this problem. Hierarchical B-splines are also used in isogeometric analysis (Vuong et al., 2011). For the partition of unity, THB-splines are introduced in Giannelli et al. (2012). Because hierarchical B-splines are generated directly from the T-meshes rather than the spline spaces, the completeness cannot be guaranteed generally. Therefore, several articles

☆ This paper has been recommended for acceptance by B. Juettler.

* Corresponding author.
E-mail address: dengjs@ustc.edu.cn (J. Deng).

http://dx.doi.org/10.1016/j.cagd.2015.07.002
0167-8396/© 2015 Elsevier B.V. All rights reserved.



(a) nested domains (b) The hierarchical T-mesh $\mathscr{T}$

Fig. 1. A nested sequence of domains (a) and the resulting hierarchical T-mesh (b). The spline space is $\mathbf{S}(3,3,2,2,\mathscr{T})$. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

(Berdinsky et al., 2014, 2015; Giannelli and Jüttler, 2013; Mokriš et al., 2014; Mokriš and Jüttler, 2014) were published to discuss this question.

However, these discussions are somewhat complex, and the rules that guarantee the completeness demand many additional cells to be refined beyond the ones we actually need. Actually, the choice of the basis functions of the traditional hierarchical B-splines leads to the incompleteness. In Fig. 1, we show the procedure of generating a hierarchical T-mesh $\mathscr{T}$ with the dyadic refinement in Giannelli et al. (2012), Vuong et al. (2011). There is a tensor-product B-spline defined on the tensor-product submesh consisting of the red lines. However, this B-spline is not considered in Giannelli et al. (2012), Vuong et al. (2011). Therefore, the hierarchical B-spline space is not complete in this simple hierarchical T-mesh. We believe adding this B-spline is reasonable.

In this paper, we consider this question directly from the spline spaces. As we know, cubic splines are the most considerable splines in both theory (Deng et al., 2008; Li et al., 2012; Li and Scott, 2014) and applications (Bai and Wang, 2010; Desquilbeta and Mariottib, 2010; Li, 2012; Mohanty and Gopal, 2011). We focus on the bicubic C^2 spline spaces over hierarchical T-meshes in this paper as well. This question has been discussed preliminarily in Wu et al. (2012). In that paper, the T-mesh is a special hierarchical T-mesh denoted by $\mathscr{T}_{3,3}$ and a basis function may be the sum of several tensor-product B-splines. In this paper, the hierarchical T-mesh is more general and all of the basis functions are tensor-product B-splines. Unlike the traditional hierarchical B-splines in Vuong et al. (2011), the basis functions concerned include the B-splines we discussed in the last paragraph. We explore the topological explanation of the dimension formula of bicubic C^2 spline spaces, which also cues the completeness of the traditional hierarchical B-splines. We give three rules of refinement to guarantee the completeness, and a basis is constructed.

The remainder of this paper consists of eight sections. In Section 2, we review the definitions and some results regarding T-meshes and spline spaces over T-meshes. Then, we review the dimension formula of bicubic C^2 spline spaces over hierarchical T-meshes. In Section 4, the topological explanation of the dimension formula is further explored and the rules of refinement are elaborated. The basis is constructed in Section 5. In Section 6, we compare the new splines with HB-splines and T-splines. We give the proof of the linear independence in Section 7. Section 8 provides a number of applications. We end the paper with conclusions and future work in Section 9.

2. T-meshes and spline spaces

2.1. T-meshes and hierarchical T-meshes

Definition 2.1. Given two finite strictly increasing sequences of numbers $\{x_1, x_2, \ldots, x_m\}, \{y_1, y_2, \ldots, y_n\}$, we obtain mn vertices $v_{i,j}$ $(1 \leqslant i \leqslant m, 1 \leqslant j \leqslant n)$ in $\mathbb{R}^2$, where the coordinate of $v_{i,j}$ is (x_i, y_j). Connecting $v_{i,j}$ and $v_{i+1,j}$ $(1 \leqslant i \leqslant m-1)$ with line segment, and connecting $v_{i,j}$ and $v_{i,j+1}$ $(1 \leqslant j \leqslant n-1)$ with line segment, we obtain a mesh, which is called a **tensor-product mesh**.

A T-mesh is a rectangular grid that allows T-junctions.

Definition 2.2. (See Deng et al., 2006, 2013.) Suppose $\mathscr{T}$ is a set of axis-aligned rectangles and the intersection of any two distinct rectangles in $\mathscr{T}$ either is empty or consists of points on the boundaries of the rectangles. Then, $\mathscr{T}$ is called a **T-mesh**. Furthermore, if the entire domain occupied by $\mathscr{T}$ is a rectangle, $\mathscr{T}$ is called a **regular T-mesh**. If some edges of $\mathscr{T}$ also form a T-mesh $\mathscr{T}'$, $\mathscr{T}'$ is called a **submesh** of $\mathscr{T}$.

In this paper, we only consider regular T-meshes. The definitions of vertex, edge and cell are the same as in Deng et al. (2006). A **T-junction** is a vertex of one rectangle that lies in the interior of an edge of another rectangle. A **crossing-vertex** is a vertex of four rectangles. An **l-edge** is a line segment that consists of several edges. It is the longest possible line segment, the interior edges of which are connected and the two end points are T-junctions or boundary vertices. If an l-edge consists

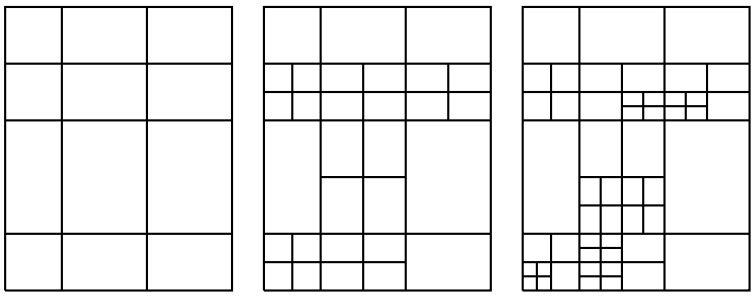


Fig. 2. The construction of a hierarchical T-mesh.

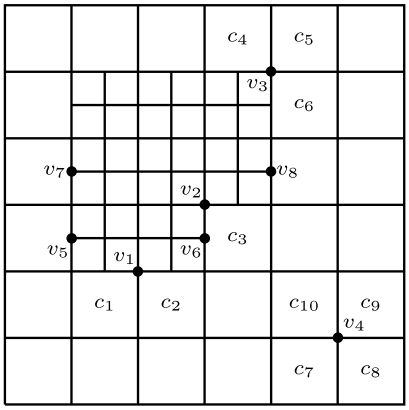


Fig. 3. A hierarchical T-mesh $\mathscr{T}$.

of some boundary edges, then it is called a **boundary l-edge**; otherwise, it is called an **interior l-edge**. Apparently, a regular T-mesh has four boundary l-edges.

A hierarchical T-mesh (Deng et al., 2008) is a special type of T-mesh that has a natural level structure. It is defined recursively. Generally, we start from a tensor-product mesh $\mathscr{T}^0$, which can be non-uniform and the elements (vertices, edges, cells and l-edges) of which are called the level 0 elements. Then, some cells at level k are each divided into 2×2 subcells equally, where the new vertices, the new edges, the new cells and the new l-edges are of level $k+1$, and the resulting T-mesh is called $\mathscr{T}^{k+1}$. Continuing this process several times, finally, we obtain a T-mesh $\mathscr{T}$, which is called **a hierarchical T-mesh**. Fig. 2 illustrates the process of generating a hierarchical T-mesh.

For a hierarchical T-mesh $\mathscr{T}$, when $\mathscr{T}^0$ is given, the maximal level number of the cells is called the **level** of the hierarchical T-mesh, which is denoted by $\mathrm{lev}(\mathscr{T})$. Conversely, if $\mathrm{lev}(\mathscr{T})$ is given, $\mathscr{T}^0$ is also determined.

We use T_k to denote the set of all of the level k l-edges. The set T_k $(k\geqslant 1)$ can be divided into several subsets, $T_{k,1}, T_{k,2}, \ldots, T_{k,m_k}$, such that the l-edges of $T_{k,i}$ do not intersect with the l-edges of $T_{k,j}$, where $i,j\in\{1,2,\ldots,m_k\}$ and $i\neq j$. The subset $T_{k,r}$ is called a **connected component** of T_k, $r\in\{1,2,\ldots,m_k\}$. For any l-edge l of T_k $(k\geqslant 1)$ and any cell c of level $k-1$, if $l\cap c$ is a line segment, we say l **crosses** c. We use $N(l)$ to denote the number of cells of level $k-1$ that l crosses. For the convenience of the following discussion, we set $N(l)=+\infty$ for the l-edge l of level 0. In a hierarchical T-mesh which has more than one cell, at least one of the four vertices of a cell is a crossing-vertex. For a given crossing-vertex v, there are four cells c_1, c_2, c_3 and c_4 surrounding v. Suppose the smallest level number of the four cells is n. We call the level n cells of c_1, c_2, c_3 and c_4 the **adjacent cells** of v.

In Fig. 3, $\mathscr{T}$ is a hierarchical T-mesh and $\mathrm{lev}(\mathscr{T})=1$; T_1 has only one connected component; c_1 and c_2 are the adjacent cells of v_1, c_3 is the adjacent cell of v_2, c_4, c_5 and c_6 are the adjacent cells of v_3, and c_7, c_8, c_9 and c_{10} are the adjacent cells of v_4; $N(l_1)=2$ and $N(l_2)=3$ for the two l-edges l_1 (between v_5 and v_6) and l_2 (between v_7 and v_8).

2.2. Spline spaces over T-meshes

Given a T-mesh $\mathscr{T}$, we use Ω to denote the region occupied by all of the cells in $\mathscr{T}$, which is called the **domain** of $\mathscr{T}$. In Deng et al. (2006), the following spline space definition is proposed:

$$\mathbf{S}(m,n,\alpha,\beta,\mathscr{T}) := \{f(x,y)\in C^{\alpha,\beta}(\Omega) : f(x,y)|_{\phi}\in\mathbb{P}_{mn}, \forall\phi\in\mathscr{T}\},$$

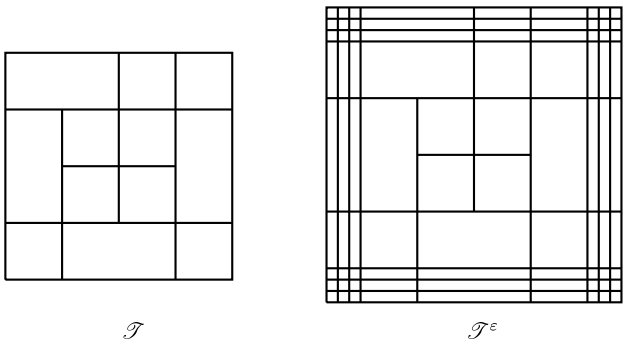


Fig. 4. A T-mesh $\mathscr{T}$ and its extended T-mesh $\mathscr{T}^{\varepsilon}$ associated with $\mathbf{S}^3(\mathscr{T})$.

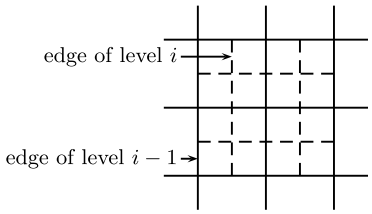


Fig. 5. A special connected component $T_{i,j}$.

where $\mathbb{P}_{mn} = \{p(x,y) : p(x,y) = \sum_{i=0}^{m}\sum_{j=0}^{n} c_{ij}x^i y^j, c_{ij} \in \mathbb{R}\}$, $C^{\alpha,\beta}$ is the space consisting of all of the bivariate functions continuous in Ω with order α along the x-direction and with order β along the y-direction, and that $\phi \in \mathscr{T}$ means ϕ is a cell of $\mathscr{T}$. It is obvious that $\mathbf{S}(m,n,\alpha,\beta,\mathscr{T})$ is a linear space. In this paper, we only discuss $\mathbf{S}(3,3,2,2,\mathscr{T})$, which is denoted as $\mathbf{S}^3(\mathscr{T})$ for convenience.

For a T-mesh $\mathscr{T}$ of $\mathbf{S}(m,n,m-1,n-1,\mathscr{T})$, we can obtain an extended T-mesh in the following fashion. Produce a tensor-product mesh $\mathscr{M}$ with $2(m+1)$ vertical lines and $2(n+1)$ horizontal lines, the central rectangle of which is identical to the region occupied by $\mathscr{T}$. Then extend the edges with one end point on the boundary of $\mathscr{T}$ to the boundary of $\mathscr{M}$. The resulting mesh, which is denoted by $\mathscr{T}^{\varepsilon}$, is called the **extended T-mesh** of $\mathscr{T}$ associated with $\mathbf{S}(m,n,m-1,n-1,\mathscr{T})$. Fig. 4 is an example of the extended T-mesh.

A spline space over a given T-mesh $\mathscr{T}$ with homogeneous boundary conditions is defined by Deng et al. (2013)

$$\bar{\mathbf{S}}(m,n,\alpha,\beta,\mathscr{T}) := \{f(x,y) \in C^{\alpha,\beta}(\mathbb{R}^2) : f(x,y)|_{\phi} \in \mathbb{P}_{mn}, \forall \phi \in \mathscr{T}, \text{ and } f|_{\mathbb{R}^2\setminus\Omega} \equiv 0\},$$

where $\mathbb{P}_{mn}$, Ω and $C^{\alpha,\beta}$ are defined as before. One important observation in Deng et al. (2013) is that the two spline spaces $\mathbf{S}(m,n,m-1,n-1,\mathscr{T})$ and $\bar{\mathbf{S}}(m,n,m-1,n-1,\mathscr{T}^{\varepsilon})$ are closely related.

Theorem 2.3. *(See Deng et al., 2013.) Given a T-mesh $\mathscr{T}$, let $\mathscr{T}^{\varepsilon}$ be the extended T-mesh associated with $\mathbf{S}(m,n,m-1,n-1,\mathscr{T})$. Then*

$$\mathbf{S}(m,n,m-1,n-1,\mathscr{T}) = \bar{\mathbf{S}}(m,n,m-1,n-1,\mathscr{T}^{\varepsilon})|_{\Omega},$$
$$\dim \mathbf{S}(m,n,m-1,n-1,\mathscr{T}) = \dim \bar{\mathbf{S}}(m,n,m-1,n-1,\mathscr{T}^{\varepsilon}).$$

With the above theorem, to consider the spline space over a T-mesh, we need only to consider the corresponding spline space with homogeneous boundary conditions over its extended T-mesh. In this paper, we only discuss the spline space $\mathbf{S}^3(\mathscr{T})$ with homogeneous boundary conditions, which is denoted by $\bar{\mathbf{S}}^3(\mathscr{T})$.

3. Dimension of $\bar{\mathbf{S}}^3(\mathscr{T})$

Definition 3.1. Given a hierarchical T-mesh $\mathscr{T}$ with $\mathrm{lev}(\mathscr{T}) \geqslant 1$, suppose $T_{i,j}(i,j \geqslant 1)$ is a connected component. If $T_{i,j}$ only divides the 2×2 neighbor cells in $\mathscr{T}^{i-1}$ (Fig. 5), we call $T_{i,j}$ a **minimal connected component**.

Theorem 3.2. *(See Zeng et al., 2015.) Suppose $\mathscr{T}$ is a hierarchical T-mesh with V^+ crossing-vertices and E interior l-edges, $\mathrm{lev}(\mathscr{T}) = n$, any l-edge t satisfies $N(t) \geqslant 2$, and there are δ_{4i} minimal connected components in T_i $(i \geqslant 1)$, where T_i is the set of all of the level i l-edges (see Section 2.1). Let $\delta_4 = \sum_{i=1}^{n}\delta_{4i}$. If $\mathscr{T}^0$ has at least 4 horizontal l-edges and 4 vertical l-edges, then*

$$\dim \bar{\mathbf{S}}^3(\mathscr{T}) = V^+ - 2E + 4 + \delta_4.$$

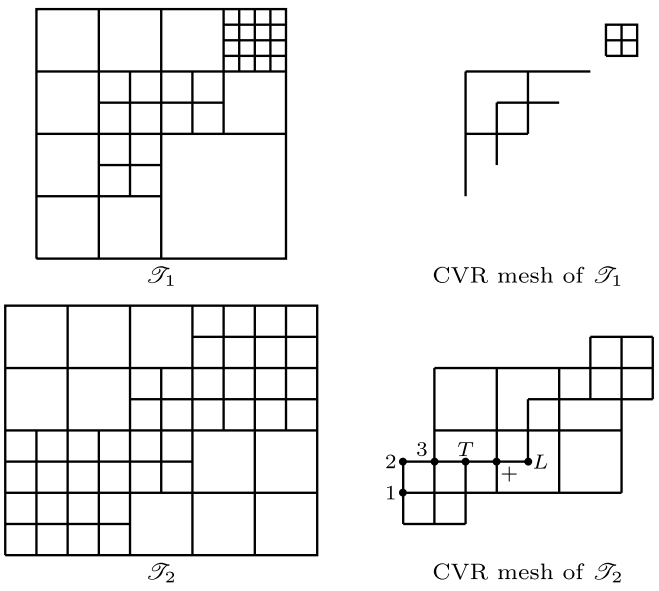


Fig. 6. Two hierarchical T-meshes and their CVR meshes.

In the following, the hierarchical T-mesh $\mathscr{T}$ we discuss satisfies that $\mathscr{T}^0$ has at least 4 horizontal l-edges and 4 vertical l-edges.

In Deng et al. (2013), the following definition is introduced to propose a topological explanation to the dimension formulae.

Definition 3.3. (See Deng et al., 2013.) Given a hierarchical T-mesh $\mathscr{T}$, retaining all of the edges that are a part of a line segment with two end points that are crossing-vertices and removing the other edges, we obtain a submesh $\mathscr{G}$. From the construction, we know that $\mathscr{G}$ consists of all of the line segments whose two end points are crossing-vertices in $\mathscr{T}$ and the vertices of $\mathscr{G}$ are the crossing-vertices of $\mathscr{T}$. The submesh $\mathscr{G}$ is called the **crossing-vertex-relationship mesh** (CVR mesh for short) of $\mathscr{T}$.

There two examples in Fig. 6.

In $\mathscr{G}$, we can define cell, edge and l-edge similarly to that in the T-mesh. The difference is that a cell of $\mathscr{G}$ may not be a rectangle.

Proposition 3.4. *For a hierarchical T-mesh $\mathscr{T}$, with the restriction of $N(t) \geqslant 2$ for any l-edge t, the CVR mesh is connected (i.e., if we treat the CVR mesh as a graph, the graph is connected).*

Proof. Suppose $\mathrm{lev}(\mathscr{T}) = n$. Consider a connected component $T_{n,1}$ of T_n. Because $N(t) \geqslant 2$, every l-edge t of $T_{n,1}$ has at least three crossing-vertices. Therefore, t corresponds to a line segment (denoted by t') whose two end points are both crossing-vertices. That is, t' will present in $\mathscr{G}$. Therefore, the number of the l-edges in $\mathscr{G}$ is the same as in $\mathscr{T}$, and the l-edges of $\mathscr{G}$ are also hierarchical. The level number of an l-edge of $\mathscr{G}$ is equal to the level number of the l-edge it corresponds to in $\mathscr{T}$.

Moreover, for two l-edge t_1 and t_2 in $\mathscr{T}$, if t_1 intersects with t_2 at a crossing-vertex, then t_1' intersects with t_2', where t_1' and t_2' are the l-edges that t_1 and t_2 corresponds to in $\mathscr{G}$, respectively. Since $\mathscr{T}^0$ has at least 4 horizontal l-edges and 4 vertical l-edges, the CVR mesh $\mathscr{G}^0$ of $\mathscr{T}^0$ is a tensor-product mesh which has at least 2 horizontal l-edges and 2 vertical l-edges. That is, $\mathscr{G}^0$ is a connected mesh. For any l-edge l at level 1, because $N(l) \geqslant 2$, l intersects with at least one l-edge l_0 of level 0 at a crossing-vertex. Therefore, l' intersects with l_0', where l' and l_0' are the l-edges that l and l_0 correspond to in $\mathscr{G}$, respectively. That is, the CVR mesh of the mesh obtained by adding l to $\mathscr{T}^0$ is a connected mesh. Similarly analyzing for other l-edges at level 1, we obtain that $\mathscr{T}^1$ is connected.

By induction on the level of the T-mesh, we know the conclusion is right. □

In the following, the hierarchical T-mesh $\mathscr{T}$ we discussed is with the restriction of $N(t) \geqslant 2$ for any l-edge t.

We use $V_{\mathscr{G}}$ to denote the number of the vertices of the CVR mesh. Therefore, $V_{\mathscr{G}} = V^+$. In $\mathscr{G}$, a vertex is the intersection of two l-edges. We distinguish vertices of $\mathscr{G}$ into six types as follows:

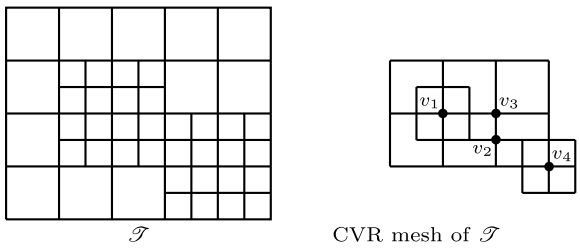


Fig. 7. A hierarchical T-mesh and its CVR mesh.

1. Type 1: a boundary vertex of $\mathscr{G}$, which is an end point of an l-edge and an interior vertex of another l-edge,
2. Type 2: a boundary vertex of $\mathscr{G}$, which is the end points of two l-edges,
3. Type 3: a boundary vertex of $\mathscr{G}$, which is the interior vertices of two l-edges,
4. Type T: an interior vertex of $\mathscr{G}$, which is a T-junction of $\mathscr{G}$,
5. Type +: an interior vertex of $\mathscr{G}$, which is a crossing-vertex of $\mathscr{G}$,
6. Type L: an interior vertex of $\mathscr{G}$, which is the end points of two l-edges.

The six types of vertices have been marked in Fig. 6. We use $V^1_{\mathscr{G}}, V^2_{\mathscr{G}}, V^3_{\mathscr{G}}, V^{\mathrm{T}}_{\mathscr{G}}, V^+_{\mathscr{G}}$ and $V^{\mathrm{L}}_{\mathscr{G}}$ to denote the numbers of the six types of vertices, respectively. For the T-mesh $\mathscr{T}_2$ in Fig. 6, we have $V^1_{\mathscr{G}_2} = 10$, $V^2_{\mathscr{G}_2} = 8$, $V^3_{\mathscr{G}_2} = 4$, $V^{\mathrm{T}}_{\mathscr{G}_2} = 2$, $V^+_{\mathscr{G}_2} = 7$, $V^{\mathrm{L}}_{\mathscr{G}_2} = 2$, where $\mathscr{G}_2$ is the CVR mesh of $\mathscr{T}_2$. Every crossing-vertex v of $\mathscr{T}$ corresponds to a vertex in the CVR mesh. If the corresponding vertex is type x, $x \in \{1, 2, 3, T, +, L\}$, we say v is a type x point in the CVR mesh for convenience.

Theorem 3.5. *(See Zeng et al., 2015.) Suppose $\mathscr{T}$ is a hierarchical T-mesh, $\mathrm{lev}(\mathscr{T}) = n$, $N(t) \geqslant 2$ for any l-edge t, its CVR mesh is $\mathscr{G}$, and there are δ_{4i} minimal connected components in T_i $(i \geqslant 1)$. Let $\delta_4 = \sum_{i=1}^{n} \delta_{4i}$. If $\mathscr{T}^0$ has at least 4 horizontal l-edges and 4 vertical l-edges, then*

$$\dim \bar{\mathbf{S}}^3(\mathscr{T}) = V^+_{\mathscr{G}} - V^L_{\mathscr{G}} + \delta_4. \tag{1}$$

4. Rules of the refinement and algorithm

There are several ways to construct a basis, such as the smoothing co-factor method (Wang, 2001), B-net method (Schumaker and Wang, 2012), and finding B-splines over local tensor-product meshes (Deng et al., 2008; Dokken et al., 2013; Sederberg et al., 2003). From the viewpoint of application, the last method is the best choice. Therefore, the first task is to determine the configuration of the hierarchical T-mesh to guarantee that all the tensor-product B-splines can span the whole spline space.

4.1. Three rules

A tensor-product B-spline in $\bar{\mathbf{S}}^3(\mathscr{T})$ is defined on a 5×5 tensor-product submesh whose CVR mesh is a 3×3 tensor-product mesh. We want to construct the basis with the help of the CVR mesh of $\mathscr{T}$. We call the 3×3 tensor-product submesh of $\mathscr{G}$ a **basic mesh**, and call the center vertex of the basic mesh an **anchor**. We use $\mathcal{A}$ to denote the set of all the anchors of $\mathscr{G}$. A vertex may be the anchor of several basic meshes.

In Fig. 7, there are 5 basic meshes; v_1, v_2, v_3 and v_4 are all of the anchors; v_1 is the anchor of 2 basic meshes.

Every 5×5 tensor-product submesh of $\mathscr{T}$ corresponds to a basic mesh of $\mathscr{G}$, whereas a basic mesh of $\mathscr{G}$ does not necessarily correspond to a 5×5 tensor-product submesh of $\mathscr{T}$. See Fig. 8 for an example. For the basic mesh in red in $\mathscr{G}$, we cannot find a 5×5 tensor-product submesh corresponding to it in $\mathscr{T}$.

The reason is that the level-difference of some adjacent cells is bigger than 1. In Fig. 8, the cell in light-gray is of level 0, while one of its adjacent cells in blue is of level 2. The level-difference of the two adjacent cells is 2. We have the following lemma.

Lemma 4.1. *For a hierarchical T-mesh $\mathscr{T}$, if the level-difference of the adjacent cells is at most one, a basic mesh of $\mathscr{G}$ corresponds to a 5×5 tensor-product submesh of $\mathscr{T}$.*

Proof. To clarify this fact, we should classify the basic meshes.

We consider the basic mesh in the refinement process. Suppose the old T-mesh is $\mathscr{T}$; we refine some cells of $\mathscr{T}$ and obtain a new mesh $\mathscr{T}'$. The anchor of the new basic mesh could be the center of a cell of $\mathscr{T}$, the center of an edge of $\mathscr{T}$ or a crossing-vertex of $\mathscr{T}$. With the restriction of the level-difference less than 2, all types of basic meshes and their corresponding 5×5 tensor-product meshes are discussed in the following. In Figs. 9, 10 and 11, the basic meshes are bounded by the red lines and the corresponding 5×5 tensor-product meshes are bounded by the dashed lines.

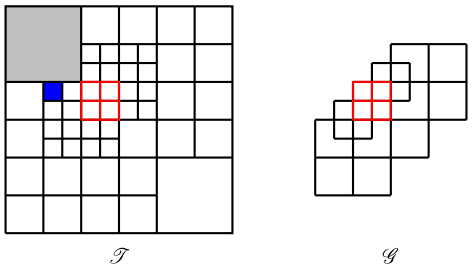


Fig. 8. A hierarchical T-mesh and its CVR mesh. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

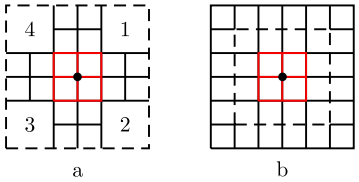


Fig. 9. The anchor is the center of a cell of $\mathscr{T}$. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

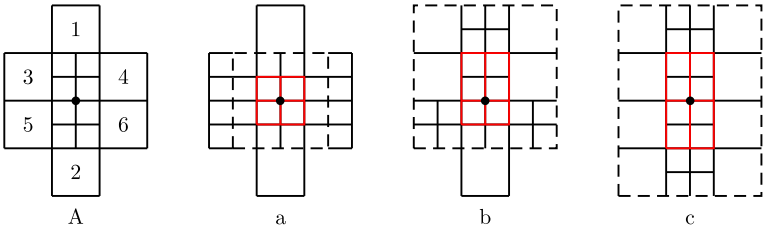


Fig. 10. The anchor is the center of an edge of $\mathscr{T}$. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

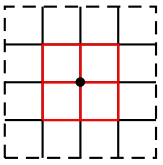


Fig. 11. The anchor is a crossing vertex of $\mathscr{T}$. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

1. The anchor is the center of a cell of $\mathscr{T}$. See Fig. 9.
   (a) If cells 1, 2, 3 and 4 are not all refined, the situation is as in Fig. 9.a.
   (b) If cells 1, 2, 3 and 4 are all refined, the situation is as in Fig. 9.b.
2. The anchor is the center of an edge of $\mathscr{T}$. See Fig. 10.A. Here we only consider the horizontal edge; the vertical edge can be discussed similarly.
   (a) If cells 3, 4, 5 and 6 are all refined, the situation is as in Fig. 10.a.
   (b) If cells 5 and 6 are both refined, cells 3 and 4 are not both refined, and cell 1 is refined, then the situation is as in Fig. 10.b.
   If cells 5 and 6 are not both refined, cells 3 and 4 are both refined, and cell 2 is refined, the situation is similar.
   (c) If cells 5 and 6 are not both refined, cells 3 and 4 are not both refined, and cells 1 and 2 are both refined, then the situation is as in Fig. 10.c.
3. The anchor is a crossing vertex of $\mathscr{T}$, the situation is as in Fig. 11. □

A basic mesh is a 3×3 tensor-product mesh, and its edges belong to 6 l-edges in $\mathscr{G}$. The maximal level number of the 6 l-edges is called the **level** of the basic mesh, and so is its corresponding 5×5 tensor-product submesh in $\mathscr{T}$. For example, in Fig. 9, if cells 1, 2, 3 and 4 are of level k, then the new basic meshes are of level $k+1$.

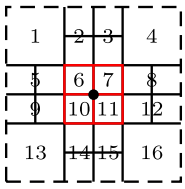


Fig. 12. The submesh $\mathscr{T}_0$ for case 1(a).

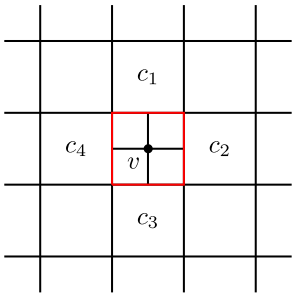


Fig. 13. Type L point. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

Every 5×5 tensor-product submesh $\mathscr{T}_0$ is divided into 16 parts by its edges, which are called the **cells** of $\mathscr{T}_0$. Fig. 12 is the mesh of case 1(a), and the 16 cells have been marked $1, 2, \ldots, 16$. Every basic mesh $\mathscr{G}_0$ is divided into 4 parts, which are called the **cells** of $\mathscr{G}_0$. In Fig. 12, the four cells are the cells 6, 7, 10 and 11.

If $\mathscr{T}_0$ is of level $k+1$, we can see that a cell of $\mathscr{T}_0$ is a cell of $\mathscr{T}$ at level k, consists of two cells of $\mathscr{T}$ at level $k+1$, or is a cell of $\mathscr{T}$ at level $k+1$. We call the three types of cells of $\mathscr{T}_0$ are of level k, level $k+1/2$, level $k+1$, respectively. In Fig. 12, the cells 1, 4, 13, 16 are of level k, and all the cells 2, 3, 5, 8, 9, 12, 14, 15 are of level $k+1/2$. For the case 1(b), the 16 cells of $\mathscr{T}_0$ are all of level $k+1$. A cell of $\mathscr{G}_0$ consists of two cells of $\mathscr{T}$ at level $k+1$, or is a cell of $\mathscr{T}$ at level $k+1$. We call the two types of cells of $\mathscr{G}_0$ are of level $k+1/2$, level $k+1$, respectively. In Fig. 12, the cells 6, 7, 10, 11 are of level $k+1$. For the case 2(c), all of the 4 cells of $\mathscr{G}_0$ are of level $k+1/2$. We call the 5×5 tensor-product mesh that consists of 16 cells at the same level a **standard tensor-product mesh**. For example, the 5×5 tensor-product meshes of case 1(b), case 2(b), and case 3 are all standard tensor-product meshes. The tensor-product B-spline defined on the standard tensor-product mesh is called a **standard B-spline**, and the CVR mesh of a standard tensor-product mesh is called a **standard basic mesh**.

Remark 4.2. For the hierarchical B-splines in Vuong et al. (2011), all of the basis functions are standard B-splines.

With the restriction of the level-difference less than 2, we discuss when the type L points appear. In Fig. 13, dividing the cell c_0 (bounded by red lines), we want the vertex v to be a type L point in the CVR mesh. Cell c_0 must be an interior cell, whose adjacent cells are c_1, c_2, c_3 and c_4. When c_1 and c_2 are divided and c_3 and c_4 are not divided, or c_2 and c_3 are divided and c_4 and c_1 are not divided, or c_3 and c_4 are divided and c_1 and c_2 are divided, or c_4 and c_1 are divided and c_2 and c_3 are not divided, v is a type L point in the CVR mesh.

Our **first rule** is: the level-difference of the adjacent cells is at most one. In the following, the hierarchical T-mesh we discuss satisfies this rule.

We have the following two facts about the CVR mesh.

1. The adjacent points of a type L point are type $+$ or type 3 points of $\mathscr{G}$;
2. A type $+$ point is either an anchor or the adjacent point of some type L points.

Definition 4.3. L **chain**: the longest chain consisting of $v_0, v_1, \ldots, v_{2n}$, where $v_0, v_2, \ldots, v_{2n}$ are type $+$ or type 3 points, $v_1, v_3, \ldots, v_{2n-1}$ are type L points, and v_{i+1} is adjacent to v_i. Because the level-difference is less than 2, the vertices in an L chain are of the same level.

Because an L chain is the longest chain, a type $+$ or type 3 point could only be in one L chain.

If v_0 or v_{2n} is type 3, the chain is called a **boundary** L **chain**; otherwise it is called an **interior** L **chain**.

For a boundary L chain, if v_0 and v_{2n} are both type 3, the chain is called a **type 1 chain**; otherwise, it is called a **type 2 chain**. For an interior L chain, if $v_0 = v_{2n}$, it is called a **type 3 chain**; otherwise, it is called a **type 4 chain**.

If two L chains have the same vertices, we regard the two chains as the same chain. For example, $v_0, v_1, \ldots, v_{2n}$ and $v_{2n}, v_{2n-1}, \ldots, v_0$ are the same chain.

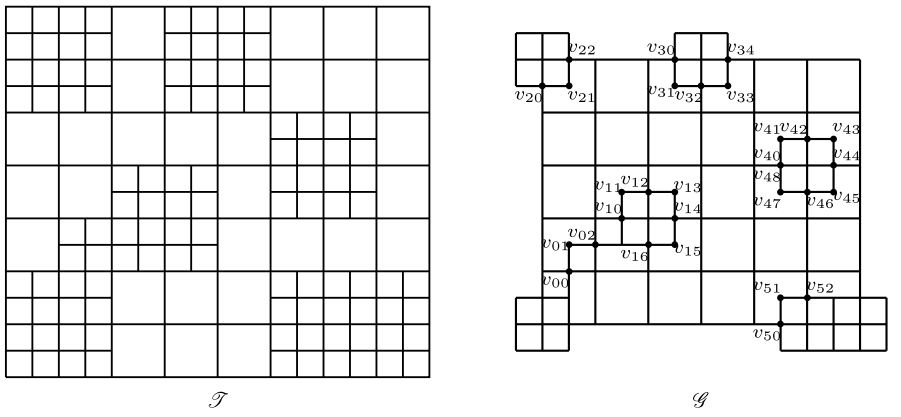


Fig. 14. L chain.

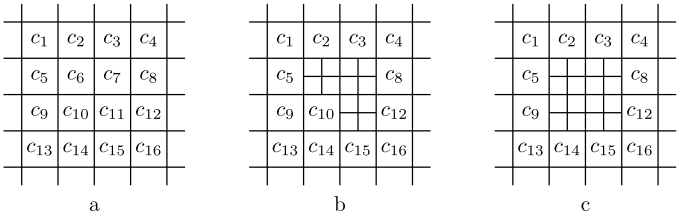


Fig. 15. Figure for the proof of Lemma 4.4.

In Fig. 14, there are 6 L chains: v_{20}, v_{21}, v_{22} and $v_{30}, v_{31}, v_{32}, v_{33}, v_{34}$ are type 1 chains; v_{50}, v_{51}, v_{52} is a type 2 chain; $v_{40}, v_{41}, \ldots, v_{48}$ is a type 3 chain; v_{00}, v_{01}, v_{02} and $v_{10}, v_{11}, \ldots, v_{16}$ are type 4 chains.

Lemma 4.4. *Type 1 chains and type 3 chains appear if and only if they are in the basic meshes corresponding to the minimal connected components (Fig. 5).*

Proof. This fact could be proved by analyzing the mesh. We only prove the conclusion for type 3 chains, and the case of type 1 chains could be proved similarly.

The sufficiency is obvious. For the necessity, consider a part of a hierarchical T-mesh in Fig. 15.a. Suppose all of the cells $c_1, c_2, \ldots, c_{16}$ are of level k. We divide some of them to construct a type 3 chain of level $k+1$. From the analysis above, the type L points in the CVR mesh appear at the center of the cells that will be divided. Suppose the top-right type L point v_L of the type 3 chain appears at the center of c_7. Here "top-right" means that in this type 3 chain, there is not a type L point above v_L, and among all of the type L points which are in the same height as v_L is, v_L is the rightmost one. Then c_6, c_7 and c_{11} will be divided, and c_3 and c_8 will not be divided. See Fig. 15.b. Because we will construct a type 3 chain, the center of c_6 will also be a type L point in the CVR mesh, otherwise, this chain will terminate. Then c_{10} will be divided, and c_2 and c_5 will not be divided. See Fig. 15.c. Similarly, the center of c_{10} and c_{11} will also be type L points in the CVR mesh, then c_9, c_{14}, c_{15} and c_{12} will not be divided.

Therefore, the connected component in Fig. 15.c is just the connected component that the type 3 chain corresponds to, which is a minimal connected component. □

Theorem 4.5. *Suppose $\mathscr{T}$ is a hierarchical T-mesh, $N(t) \geqslant 2$ for any l-edge t, the level-difference of the adjacent cells is at most one, and the numbers of type 3 chains and type 4 chains in $\mathscr{G}$ are C_3 and C_4, respectively, where $\mathscr{G}$ is the CVR mesh of $\mathscr{T}$. Then,*

$$\dim \tilde{\mathbf{S}}^3(\mathscr{T}) = \#\mathcal{A} + C_3 + C_4, \tag{2}$$

where $\#\mathcal{A}$ is the number of the elements of $\mathcal{A}$.

Proof. Suppose the numbers of type 1 chains and type 2 chains in $\mathscr{G}$ are C_1 and C_2, respectively. By Lemma 4.4, we have

$$\delta_4 = C_1 + C_3, \tag{3}$$

where δ_4 has the same meaning as in Theorem 3.5.

For a type 2 chain or a type 3 chain, the numbers of type $+$ and type L points are the same. For a type 1 chain, there is one more type L point than type $+$ points. For a type 4 chain, there is one less type L point than type $+$ points. Therefore,

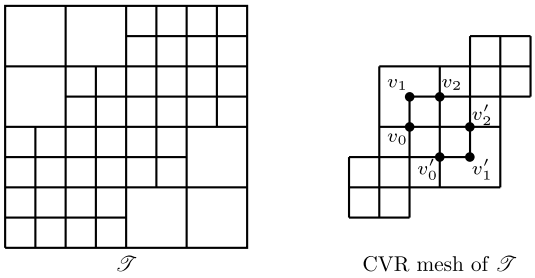


Fig. 16. A hierarchical T-mesh and its CVR mesh.

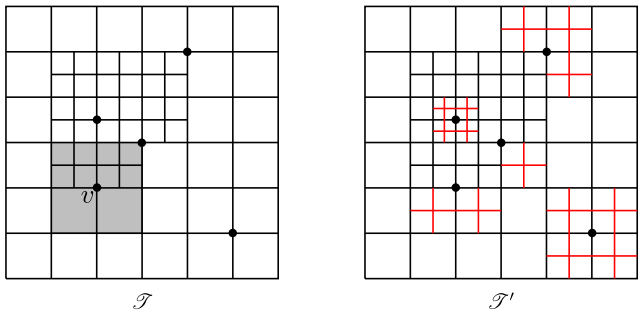


Fig. 17. The refinement style.

$$V_{\mathscr{G}}^{+} - V_{\mathscr{G}}^{L} = \#\mathcal{A} + C_4 - C_1. \tag{4}$$

Combining Equations (1), (3) and (4), we obtain Equation (2). □

We want all of the basis functions to be tensor-product B-splines. From Equation (2), we know that every type 4 chain must correspond to a basic mesh, that is, every type L point should be in a basic mesh. To explain this phenomenon simply, consider the hierarchical T-mesh $\mathscr{T}_2$ in Fig. 6. By Equation (1) or Equation (2), we have $\dim \bar{\mathbf{S}}^3(\mathscr{T}_2) = 5$. However, there are only 4 basic meshes in the CVR mesh of $\mathscr{T}_2$. That is, all of the tensor-product B-splines cannot span the space of $\bar{\mathbf{S}}^3(\mathscr{T}_2)$. Therefore, our **second rule** is: every type 4 chain must correspond to a basic mesh.

From Equation (2), we know that different interior L chains should correspond to different basic meshes. In Fig. 16, the L chain v_0, v_1, v_2 and the L chain v_0', v_1', v_2' are in the same basic mesh. By Theorem 3.5, we have $\dim \bar{\mathbf{S}}^3(\mathscr{T}) = 7$, but we can only find 6 basic meshes.

To prevent the occurrence of such a situation, we set the **third rule**: two different L chains cannot be in the same (standard) basic mesh.

Now we collect all the restrictions of the hierarchical T-mesh here:

1. the hierarchical T-mesh $\mathscr{T}$ is regular;
2. the initial tensor-product mesh $\mathscr{T}^0$ has at least 4 horizontal l-edges and 4 vertical l-edges;
3. $N(t) \geqslant 2$ for any l-edge t;
4. the level-difference of the adjacent cells is at most one;
5. every type 4 chain corresponds to a basic mesh;
6. two different L chains are not in the same (standard) basic mesh.

Then we can define the **multiplicity** of the anchor under these restrictions. For an anchor v, if there are m type 3 and type 4 chains, the anchors of which are all v, then the multiplicity of v is $m+1$, denoted by $\mathrm{mul}(v) = m+1$. Combining with Equation (2), we have

$$\dim \bar{\mathbf{S}}^3(\mathscr{T}) = \sum_{v \in \mathcal{A}} \mathrm{mul}(v). \tag{5}$$

For example, in Fig. 7, $\mathrm{mul}(v_1) = 2, \mathrm{mul}(v_2) = \mathrm{mul}(v_3) = \mathrm{mul}(v_4) = 1$.

4.2. Algorithm for the refinement

In this section, we introduce the algorithm for the refinement briefly. Usually, we choose a 5×5 tensor-product mesh as $\mathscr{T}^0$, no matter whether it is uniform or not.

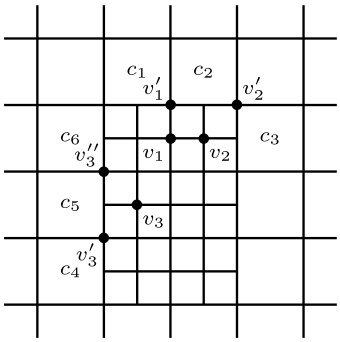


Fig. 18. The first rule.

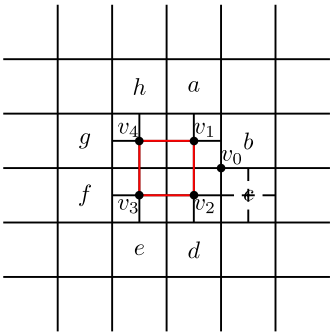


Fig. 19. The third rule. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

4.2.1. Refinement style for the second rule

To ensure the second rule, we set the **refinement style**: we refine the mesh around the crossing vertices. If we want to refine around a crossing vertex v_0, we divide the adjacent cells of v_0. In Fig. 17, the crossing vertices of $\mathscr{T}$ that will be refined around have been marked and $\mathscr{T}'$ is the resulting T-mesh. Under this refinement style, every type 3 or type 4 chain is in a standard basic mesh. We call the anchor of the basic mesh the **anchor** of this L chain. With this refinement style, we have $N(t) \geqslant 2$ for any l-edge t as well.

Different from other types of splines, we refine the mesh based on crossing-vertices instead of cells. Now we give the algorithm for finding the crossing-vertices that will be refined around.

We compute the error around every crossing vertex. For a given crossing vertex v_0, suppose the adjacent cells of v_0 are of level k. There are four cells of level k in $\mathscr{T}^k$ that are around v_0. For example, in Fig. 17, the four cells are in light-gray for the crossing vertex v. We compute the error on the area occupied by the four cells. If the error is greater than a given tolerance ε, this crossing-vertex should be refined around.

4.2.2. Algorithm for the first rule

To guarantee that the level-difference is at most one, we should refine around more crossing-vertices.

See Fig. 18. There are three situations that we should refine around more crossing-vertices. If we want to refine around v_1, because either the cell c_1 or c_2 is not divided, we should refine around v_1' first. If we want to refine around v_2, because neither of the cell c_2 and c_3 is divided, we should refine around v_2' first. If we want to refine around v_3, because the cell c_5 is not divided, we should refine around v_3' or v_3'' first. Our principle is to divide as few cells as possible. Therefore, if the cell c_4 has been divided already, we choose v_3' to refine around, which means we should only divide c_5. Similarly, if the cell c_4 has not been divided and the cell c_6 has been divided already, we choose v_3'' to refine around, which means we should only divide c_5 as well. If neither of c_4 and c_6 has been divided, we can choose either of v_3' and v_3'' to refine around.

4.2.3. Algorithm for the third rule

A standard basic mesh can have one, two, three or four type L points. The situation where two different L chains appear in the same (standard) basic mesh occurs only when the standard basic mesh has two type L points that are not in two different L chains.

We use Fig. 19 to illustrate when this situation occurs. The red lines correspond to the boundary lines of a standard basic mesh in the CVR mesh. We pay attention to the four vertices v_1, v_2, v_3, v_4 that can be type L points. It is easy to check that this situation occurs when v_1 and v_3 are both type L points and neither of v_2 and v_4 is a type L point, or v_2 and v_4 are both type L points and neither of v_1 and v_3 is a type L point.

Here we suppose v_1 and v_3 are both type L points in the CVR mesh and neither of v_2 and v_4 is a type L point in the CVR mesh. Therefore, none of the cells a, b, e and f is divided, c or d is divided, and g or h is divided. To avoid this situation, we divide more cells to change v_1 or v_3 to a type $+$ point or a type T point. The refinement style tells us that we should refine around the crossing vertex. We choose an appropriate crossing vertex to avoid dividing more cells. For example, in Fig. 19, c is divided. Therefore, we choose v_0 to refine around. Then, we should only divide b.

4.2.4. Algorithm

If we want to refine around a vertex v, Algorithm 1 is a simple process for finding a sequence of cross-vertices that should be refined around to ensure the first rule and the third rule. Here we use the recursion of functions. This algorithm is always guaranteed to terminate, because the worst case is that the mesh will become a tensor-product mesh after refining around all the crossing-vertices in RefineListAt(v).

In the actual operation, we should refine around more than one cross-vertex. Algorithm 2 is a simple process of the refinement of a hierarchical T-mesh.

Algorithm 1 Finding a sequence of cross-vertices for refining around a crossing-vertex.

Input: A cross-vertex v_0 that will be refined around
Output: A sequence of cross-vertices S that should be refined around to ensure the first rule and the third rule

```
function RefineListAt(v_0)
    push v_0 to a vector S
    if v_0 does not satisfies the third rule then
        find out a crossing-vertex v' should be refined around
        push v' to S
        RefineListAt(v')
    end if
    if v_0 does not satisfies the first rule then
        find out a crossing-vertex v'' should be refined around
        push v'' to S
        RefineListAt(v'')
    end if
    erase the repetitive elements in S
    return S
end function
```

Algorithm 2 Refinement of a hierarchical T-mesh.

Input: A hierarchical T-mesh $\mathscr{T}$ and a tolerance ε
Output: A refined hierarchical T-mesh $\mathscr{T}'$

```
for each crossing-vertex v of T do
    compute the error e[v] around v
    if e[v] > ε then
        push v to a vector R
    end if
end for
for each crossing-vertex v ∈ R do
    if v has not been refined around in the process of this algorithm then
        S[v] ← RefineListAt(v)
        for each crossing-vertex v' ∈ S[v] do
            refine around v
        end for
    end if
end for
return T'
```

5. Construction of the basis

In this section, we provide a detailed construction process for the bicubic hierarchical basis. Suppose $\mathrm{lev}(\mathscr{T}) = N$. Now that we refined around a crossing vertex v, we divide some cells at level $k+1$. The chosen cells are the adjacent cells of v.

5.1. The basis functions at level $k+1$

In Fig. 20, we divide the cell bounded by red lines. The crossing vertices (if they exist) upon this cell that will be influenced have been marked $v_1, v_2, \ldots, v_7$. We only consider these crossing vertices, and the crossing vertices in other directions can be dealt with in the same manner.

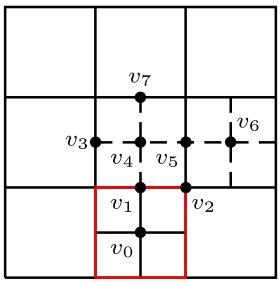


Fig. 20. Figure for the construction of the basis. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

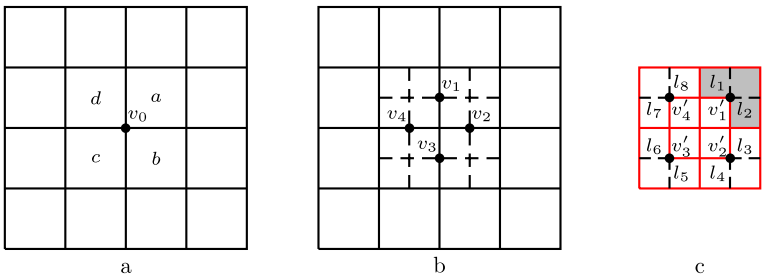


Fig. 21. Delete a basis function. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

We can analyze every vertex of $v_1, v_2, \ldots, v_7$ as the classification of anchors in Section 4. If the vertex satisfies one of the cases, we add the corresponding tensor-product B-spline to the original basis. After treating all of the vertices surrounding the divided cell, we deal with the crossing vertex v_0.

We use $\mathrm{lev}(f)$ to represent the level of the basis function f and $\mathcal{F}_i^0$ to represent the set of all of the basis functions at level i, and $\mathcal{F}^0 = \bigcup_{i=0}^{N} \mathcal{F}_i^0$, where $\mathcal{F}_0^0$ is the tensor-product basis functions of $\bar{\mathbf{S}}^3(\mathscr{T}^0)$ as in Vuong et al. (2011).

5.2. Deleting some of the basis functions at level k

An anchor can be the center of several basic meshes at different levels.

We delete the basis function at level k if the domain of its basic mesh is covered by the domains of the basic meshes of some basis functions at level $k+1$. Because the level difference is less than 2, this only occurs to the standard basis functions (i.e., standard B-splines). Now we discuss when this situation appears.

In Fig. 21.a, the 5×5 tensor-product mesh is the domain of a standard B-spline g at level k. First, the cells a, b, c and d should all be divided. This is because the cells of the basic mesh at level $k+1$ are at least at level $k+1/2$. In Fig. 21.b, the four cells have been divided. In Fig. 21.c, the red lines compose the CVR mesh of the mesh in Fig. 21.b.

Second, the four crossing vertices v_1, v_2, v_3 and v_4 should all be anchors. It is easy to verify that when this is true, the domain of the basic mesh of g is covered. To prove the necessity, look at the CVR mesh in Fig. 21.c. The edge l_i or l_{i+1}, where $i = 1, 3, 5, 7$, should exist in the CVR mesh. Otherwise, suppose neither of l_1 and l_2 exists. Then, there is a cell (the light-gray cell) that is neither at level $k+1/2$ nor at level $k+1$, thus it cannot be contained in any basic mesh of level $k+1$, a contradiction. When l_1 or l_2 exists, and l_7 or l_8 exists, v_1 is an anchor. Similarly, v_2, v_3 and v_4 are all anchors. This conclusion is equivalent to none of v_1', v_2', v_3' or v_4' is a type L point.

After deleting, we use $\mathcal{F}_i$ to represent the basis functions remaining at level i. The final basis we obtain is denoted by $\mathcal{F} = \bigcup_{i=0}^{N} \mathcal{F}_i$.

Lemma 5.1. *After the process above, we have*

$$\dim \bar{\mathbf{S}}^3(\mathscr{T}) = \#\mathcal{F}. \tag{6}$$

Proof. After this process, for $\forall$ anchor v, we claim that there are $\mathrm{mul}(v)$ basis functions whose anchor is v.

We use Fig. 21 to prove this claim. The crossing-vertex v_0 is an anchor. Without losing generality, we can assume v_0 is of level 0. There is one basis function f_0 whose anchor is v_0 in $\mathcal{F}^0$. After dividing the cells a, b, c and d, v is the anchor of a basic mesh of level 1. If f_0 is not deleted, then at least one of v_1', v_2', v_3', v_4' is a type L point. That is, this basic mesh corresponds to a type 3 or type 4 chain of level 1. Therefore, $\mathrm{mul}(v_0) = 2$ in $\mathscr{T}^1$. If f_0 is deleted, none of v_1', v_2', v_3', v_4' is a type L point. That is, this basic mesh does not correspond to a type 3 or type 4 chain of level 1. Therefore, $\mathrm{mul}(v_0) = 1$ in $\mathscr{T}^1$. By induction on the level, we know this claim is right.

Combing with Equation (5), we complete the proof. □

5.3. An example

In this section, we give an example to illustrate the process above. We use $N[x_0, x_1, x_2, x_3, x_4](x)$ to denote the cubic B-spline basis function associated with the knot vector $[x_0, x_1, x_2, x_3, x_4]$ in x-direction, $N[y_0, y_1, y_2, y_3, y_4](y)$ to denote the cubic B-spline basis function associated with the knot vector $[y_0, y_1, y_2, y_3, y_4]$ in y-direction, and f_v^i to denote the basis function at level i whose anchor is v.

In Fig. 22, $\mathscr{T}$ is the initial T-mesh and $\mathscr{G}$ is its CVR mesh. The vertices $v_1^0, v_2^0, v_3^0, v_4^0$ are all the anchors. Therefore, $\#\mathcal{F}_0^0 = 4$. The basis function whose anchor is v_1^0 is

$$f_{v_1^0}^0 = N[0,1,2,3,4](x)N[1,2,3,4,5](y).$$

Now we refine around v_4^0, v_5^0, v_6^0 and obtain a new mesh $\mathscr{T}'$ whose CVR mesh is $\mathscr{G}'$. By Lemma 4.1, three basis functions whose anchors are v_4^0, v_5^0, v_6^0 are added for case 3, one basis function whose anchor is v_3^1 is added for case 2(a), one basis function whose anchor is v_2^1 is added for case 2(b), and one basis function whose anchor is v_1^1 is added for case 2(c). Because the adjacent cells of v_1^0 are not all divided in the refinement process, $f_{v_1^0}^0$ will not change. So do $f_{v_2^0}^0$ and $f_{v_3^0}^0$. For v_4^0, because the vertex v' is a type L point, $\mathrm{mul}(v_0) = 2$ and $f_{v_4^0}^0$ will not change as well. We have

$$f_{v_4^0}^0 = N[1,2,3,4,5](x)N[0,1,2,3,4](y),$$
$$f_{v_4^0}^1 = N[2,2.5,3,3.5,4](x)N[1,1.5,2,2.5,3](y).$$

Now we refine around v_7^0 and obtain a new mesh $\mathscr{T}''$ whose CVR mesh is $\mathscr{G}''$. By Lemma 4.1, one basis function whose anchor is v_7^0 is added for case 3, one basis function whose anchor is v' is added for case 1(a), two basis functions whose anchors are v_4^1, v_5^1 are added for case 2(b). Because v' is no longer a type L point in $\mathscr{G}''$, $\mathrm{mul}(v_0) = 1$ and $f_{v_4^0}^0$ will be deleted.

Finally, we have $\mathcal{F}_0 = \{f_{v_1^0}^0, f_{v_2^0}^0, f_{v_3^0}^0\}$, $\mathcal{F}_1 = \{f_{v_5^0}^1, f_{v_6^0}^1, f_{v_7^0}^1, f_{v_4^0}^1, f_{v_1^1}^1, f_{v_2^1}^1, f_{v_3^1}^1, f_{v_4^1}^1, f_{v_5^1}^1, f_{v'}^1\}$.

6. Properties

In this section, we discuss some properties of the basis obtained in Section 5. The support set of any function f is

$$\mathrm{supp}(f) = \{(x, y) \in \Omega : f(x, y) \neq 0\},$$

where Ω is the region occupied by the cells in $\mathscr{T}$.

6.1. Linear independence

First, we give three lemmas about the basis.

Lemma 6.1. *For two basis functions f_1 and f_2, suppose $\mathscr{T}_1$ and $\mathscr{T}_2$ are the tensor-product meshes which f_1 and f_2 are defined on, respectively. In Fig. 23, $\mathscr{T}_1$ and $\mathscr{T}_2$ are bounded by solid lines and dashed lines, respectively. The points A_1, A_2, A_3 and A_4 are the four corner points of $\mathscr{T}_1$. If $\mathrm{supp}(f_1) \cap \mathrm{supp}(f_2) \neq \emptyset$, $\mathrm{supp}(f_1) \not\subseteq \mathrm{supp}(f_2)$ and $\mathrm{supp}(f_2) \not\subseteq \mathrm{supp}(f_1)$, then A_i cannot be the corner point of $\mathscr{T}_2$, where $i = 1, 2, 3, 4$.*

Proof. This lemma can be proved by checking all of the types of the basis functions. If $\mathrm{lev}(f_2) > \mathrm{lev}(f_1)$, suppose A_1 is also the corner point of $\mathscr{T}_2$. Then, $\mathrm{supp}(f_2) \subseteq \mathrm{supp}(f_1)$, a contradiction. Conversely, if $\mathrm{lev}(f_2) < \mathrm{lev}(f_1)$, then $\mathrm{supp}(f_1) \subseteq \mathrm{supp}(f_2)$, also a contradiction.

If $\mathrm{lev}(f_1) = \mathrm{lev}(f_2)$, this conclusion is also correct. We omit the proof here. □

Lemma 6.2. *Suppose $\mathcal{F}_k^0 = \{f_1, f_2, \ldots, f_m\}$. For any basis function f at level $k+1$, $\exists\, i, i \in \{1, 2, \ldots, m\}$, such that $\mathrm{supp}(f) \subseteq \mathrm{supp}(f_i)$.*

Proof. We should check this conclusion for all of the types of the basis functions. Here, we only check this lemma for the first type (in Fig. 9.a) of the basis functions. The others can be verified similarly.

The set $\mathrm{supp}(f)$ consists of 9 cells at level k, which have been marked in Fig. 24.a. The cells at level k are obtained by dividing the cells at level $k-1$. Therefore, the cells 1, 2, 4 and 5, or 2, 3, 5 and 6, or 4, 5, 7 and 8, or 5, 6, 8 and 9 are from the same cell at level $k-1$. Here, we suppose the cells 4, 5, 7 and 8 are from the same cell at level $k-1$. In Fig. 24.b, the cells at level $k-1$ have been bounded by red lines. The standard B-spline whose support is occupied by the four cells at level $k-1$ is the one we are looking for. □

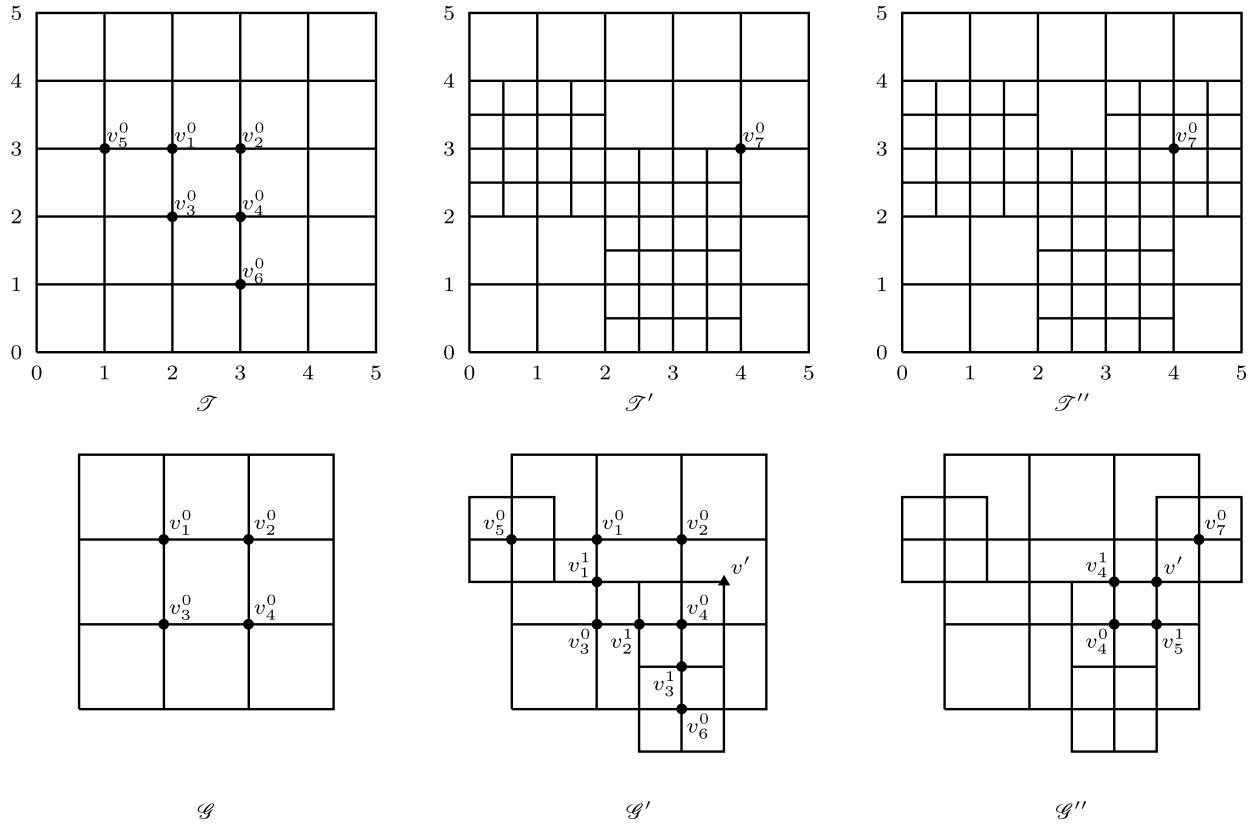


Fig. 22. An example for constructing basis functions.

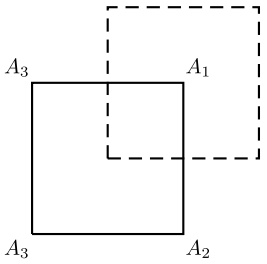


Fig. 23. Figure for Lemma 6.1.

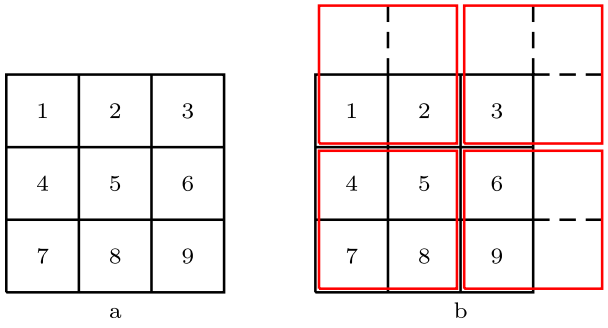


Fig. 24. Figure for Lemma 6.2. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

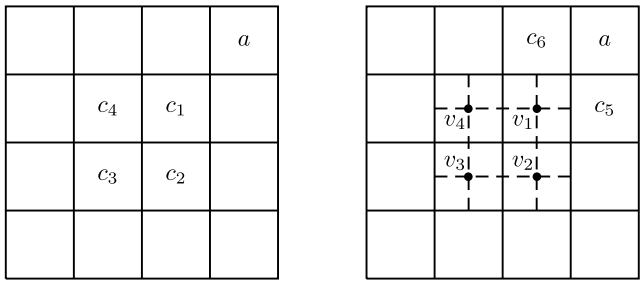


Fig. 25. Figure for Lemma 6.3.

Lemma 6.3. *Suppose $f \in \mathcal{F}_k$, $f_1, f_2, \ldots, f_m$ are all of the basis functions at level $k+1$ whose support is a subset of* $\mathrm{supp}(f)$. *Then,*

$$\mathrm{supp}(f) \backslash \bigcup_{i=1}^{m} \mathrm{supp}(f_i) \neq \emptyset$$

Proof. If f is not a standard basis function, $\mathrm{supp}(f)$ consists of at least one cell at level $k-1$. However, the support set of the basis functions at level $k+1$ cannot contain the cells at level $k-1$. Therefore, the conclusion is correct for this situation.

If f is a standard basis function, see Fig. 25.a. If not all of the cells c_1, c_2, c_3 and c_4 are divided, we can suppose c_1 is not divided. Then we can check that the cell $a \nsubseteq \bigcup_{i=1}^{m} \mathrm{supp}(f_i)$. If all of c_1, c_2, c_3 and c_4 are divided, see Fig. 25.b. Because f is not deleted in the refinement, at least one of v_1, v_2, v_3 or v_4 is a type L point in the CVR mesh. Suppose v_1 is a type L point in the CVR mesh. Then, both c_5 and c_6 are not divided. We can also check that the cell $a \nsubseteq \bigcup_{i=1}^{m} \mathrm{supp}(f_i)$. □

Theorem 6.4. *The functions in $\mathcal{F}$ are linearly independent.*

Proof. Suppose $\mathcal{F}_0^0 = \{f_1, f_2, \ldots, f_m, f_{m+1}, \ldots, f_n\}$, $\mathcal{F}_0 = \{f_1, f_2, \ldots, f_m\}$, and $\mathrm{lev}(\mathscr{T}) = N$.

For any function f at level 1, from Lemma 6.2, we know that $\mathrm{supp}(f)$ is a subset of the support sets of some functions at level 0. We only choose one of them and call it the mother function of f, which is denoted by $\mathrm{mot}_0(f)$. Similarly, we can define $\mathrm{mot}_{i-1}(f')$ for a function f' at level i, where $1 \leqslant i \leqslant N$. For a function g at level k, we denote $\mathrm{mot}_0(\mathrm{mot}_1 \cdots (\mathrm{mot}_{k-1}(g)) \cdots)$ as $\mathrm{mot}_0(g)$.

Suppose

$$\sum_{f \in \mathcal{F}} c(f) f = 0, \qquad c(f) \in \mathbb{R} \tag{7}$$

We should only prove $c(f) = 0$ for all the $f \in \mathcal{F}$.

Let

$$g_i = \begin{cases} c(f_i) f_i + \sum_{f \in \mathcal{F}_1, \mathrm{mot}_0(f) = f_i} c(f) f + \cdots + \sum_{f \in \mathcal{F}_n, \mathrm{mot}_0(f) = f_i} c(f) f, & 1 \leqslant i \leqslant m, \\ \sum_{f \in \mathcal{F}_1, \mathrm{mot}_0(f) = f_i} c(f) f + \cdots + \sum_{f \in \mathcal{F}_n, \mathrm{mot}_0(f) = f_i} c(f) f, & m+1 \leqslant i \leqslant n. \end{cases}$$

Equation (7) becomes

$$\sum_{i=1}^{n} g_i = 0. \tag{8}$$

From the definition, we know $\mathrm{supp}(g_i) = \mathrm{supp}(f_i)$. Consider the bottom-left cell α of the mesh $\mathscr{T}$. Here "bottom-left" means that there is not a cell whose bottom edge is below α, and among the cells whose bottom edges are in the same horizontal line as α is, α is the leftmost cell. From Lemma 6.1, there is only one function $f_j \in \mathcal{F}_0^0$ satisfying $\alpha \in \mathrm{supp}(f_j)$, namely $\alpha \in \mathrm{supp}(g_j)$. Therefore, g_j is the only non-zero function acting on α. By Equation (8), we obtain $g_j = 0$.

Excluding g_j from Equation (8), we consider the bottom-left cell of the region given by $\mathscr{T} \backslash \alpha$. Also from Lemma 6.1, there is only one function of $g_k, k \neq j$ that can be non-zero on this cell, and it is zero. Continuing this argument, we obtain $g_l = 0$ for all $1 \leqslant l \leqslant n$.

From the definition, we know that $\bigcup_{f \in \mathcal{F}_{k+1}^0, \mathrm{mot}_0(f) = f_i} \mathrm{supp}(f) \subseteq \bigcup_{f \in \mathcal{F}_k^0, \mathrm{mot}_0(f) = f_i} \mathrm{supp}(f)$. Therefore, by Lemma 6.3, we have

$$\mathrm{supp}(f_i) \backslash \bigcup_{k=1}^{N} \bigcup_{f \in \mathcal{F}_k, \mathrm{mot}_0(f) = f_i} \mathrm{supp}(f)$$

$$\supseteq \operatorname{supp}(f_i)\setminus \bigcup_{k=1}^{N} \bigcup_{f\in\mathcal{F}_k^0,\operatorname{mot}_0(f)=f_i} \operatorname{supp}(f)$$
$$= \operatorname{supp}(f_i)\setminus \bigcup_{f\in\mathcal{F}_1^0,\operatorname{mot}_0(f)=f_i} \operatorname{supp}(f)$$
$$\neq \emptyset,$$

where $1 \leqslant i \leqslant m$. Because $g_i = 0$, we obtain $c(f_i) = 0, 1 \leqslant i \leqslant m$.

Now every g_i has become:

$$g_i = \sum_{f\in\mathcal{F}_1,\operatorname{mot}_0(f)=f_i} c(f)f + \cdots + \sum_{f\in\mathcal{F}_n,\operatorname{mot}_0(f)=f_i} c(f)f.$$

We can repeat the discussion on every g_i, and obtain $c(f) = 0$ when $f \in \mathcal{F}_1, \operatorname{mot}_0(f) = f_i$.

Continuing this process, we prove this theorem. □

6.2. Other properties

The basis functions are all tensor-product B-splines, so they inherit some properties, such as nonnegativity and local support. From Equation (6) and the linear independence, we know they are complete. Therefore, $\mathcal{F}$ is a basis of $\bar{\mathbf{S}}^3(\mathscr{T})$.

7. Comparing with HB-splines and T-splines

Our new splines are somewhat similar to HB-splines in Vuong et al. (2011), but the two types of splines are different in essence:

1. The starting points are different. The starting point of HB-splines is that we first construct the basis, and then add some restrictions to guarantee the completeness. The starting point of our new splines is that we first analyze the spline spaces and the topological explanation of the dimension formula, and then construct the basis.
2. The basis is different. We have discussed this difference in Section 4. The tensor-product mesh that a basis function is defined on in HB-splines consists of cells from the same level, while the tensor-product mesh that a basis function is defined on in our new splines may consist of cells from different levels.

The starting point of T-splines is the same as HB-splines. The basis (blending functions for more accurate) were constructed as early as in 2003 (Sederberg et al., 2003), while the analysis of the basis begun from 2010 (Buffa et al., 2010). The biggest difference between T-splines and our new splines is that the method of the construction is different. For the same (hierarchical) T-mesh, every vertex is an anchor for T-splines, while only a small part of the vertices are anchors for our new splines.

8. Applications

8.1. Finite element analysis

In this section, we attempt to solve a two-dimensional elliptic boundary value problem (BVP) $-\Delta u = f$ defined on $[0,1]\times[0,1]$ with homogeneous conditions. Here $f = -\{40\,000[(2x-1)^2(y^2-y)^2 + (x^2-x)^2(2y-1)^2] + 400(y^2-y+x^2-x)\}\mathrm{e}^{200(x^2-x)(y^2-y)}$. The exact solution is $u = \mathrm{e}^{200(x^2-x)(y^2-y)} - 1$.

The exact solution is depicted in Fig. 26. Fig. 27 shows the adaptive hierarchical T-meshes after 1, 4, 9 and 13 refinements. Fig. 28 gives the convergence results for $S^3(\mathscr{T})$ over hierarchical T-meshes and for $S^3(\mathscr{T})$ over tensor-product meshes. In Table 1 we show the iteration number (n), dimensions of the spline spaces, the CPU time, and the error $\|u_{\text{exact}} - u^{\text{h}}\|_{\text{L}^2}/\|u_{\text{exact}}\|_{\text{L}^2}$ for $S^3(\mathscr{T})$ over hierarchical T-meshes and for $S^3(\mathscr{T})$ over tensor-product meshes.

8.2. Isogeometric analysis

In this section, we give two examples of the application on IGA.

8.2.1. Linear elasticity: plate with a circular hole

This problem is a benchmark problem: an infinite plate with a circular hole under constant in-plane tension in the x-direction in IGA and can be described as follows. The infinite plate is modeled by a finite quarter plate. The setup is illustrated in Fig. 29. The exact solution evaluated at the boundary of the finite quarter plate is applied as a Neumann boundary condition. Here, T_x is the magnitude of the applied stress for the infinite plate case (we let $T_x = 10$ here), R is the radius of the hole, L is the length of the finite quarter plate, E is Young's modulus, and ν is Poisson's ratio.

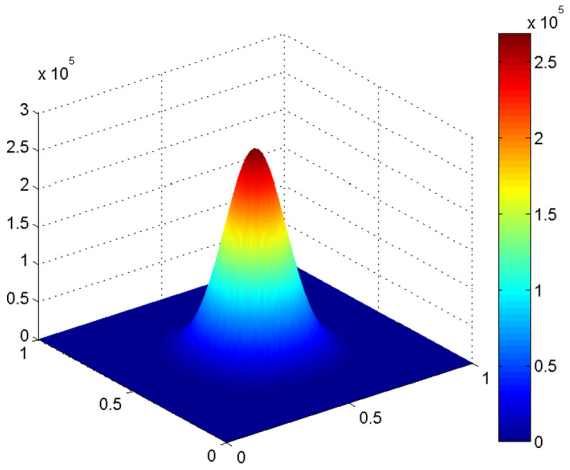


Fig. 26. The exact solution.

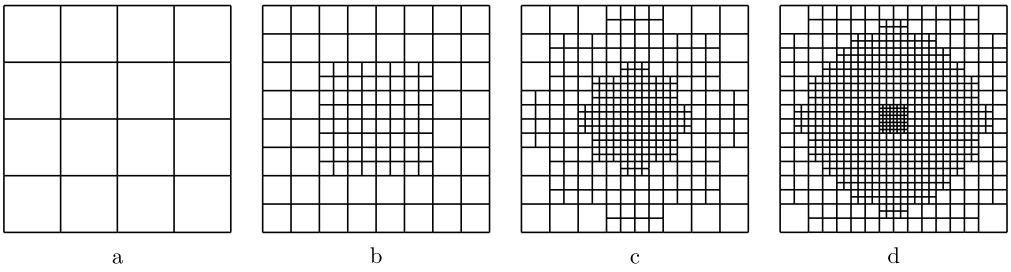


Fig. 27. Resulting meshes after 1, 4, 9, 13 refinements.

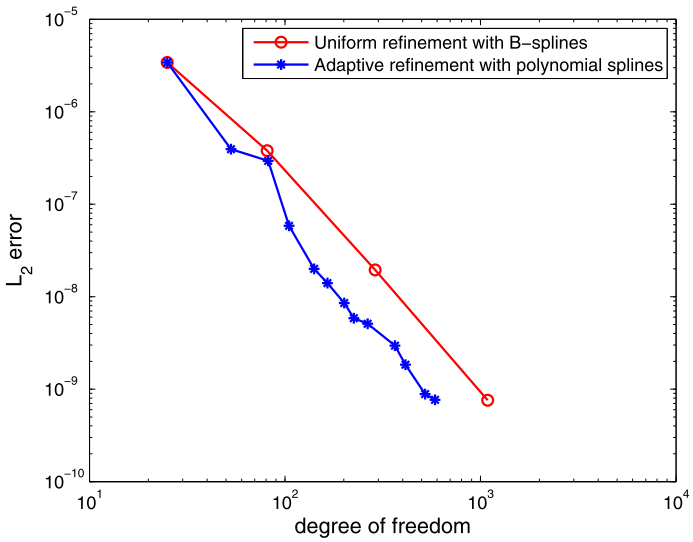


Fig. 28. Comparison of the convergence results for $S^3(\mathscr{T})$ over T-meshes and $S^3(\mathscr{T})$ over tensor-product meshes.

At the initial tensor-product mesh, the geometry is represented exactly with two patches by rational bicubic B-splines. The geometrical representation remains the same as divisions are being performed. Fig. 30 shows the adaptive hierarchical T-meshes after 1, 5, 10, 14 and 18 refinements. Fig. 31 is the xy-direction stress error $\|\sigma_{\text{exact}} - \sigma^{\text{h}}\|_{\text{L}^2}$ after 1, 5 and 10 refinements. Fig. 32 gives the convergence results for $S^3(\mathscr{T})$ over hierarchical T-meshes and for $S^3(\mathscr{T})$ over tensor-product meshes. In Table 2, we show the iteration number (n), the dimensions of the spline spaces, the CPU time and the xy-direction stress error $\|\sigma_{\text{exact}} - \sigma^{\text{h}}\|_{\text{L}^2}$ for $S^3(\mathscr{T})$ over T-meshes and $S^3(\mathscr{T})$ over tensor-product meshes.

Table 1
Experiment data.

Spline over hierarchical T-meshes				B-splines over tensor-product meshes			
n	dim	$t(s)$	error	n	dim	$t(s)$	error
1	25	0.501	3.41164×10^{-6}	1	25	0.501	3.41164×10^{-6}
2	53	0.877	3.94374×10^{-7}	2	81	1.751	3.79702×10^{-7}
5	141	2.547	2.00426×10^{-8}	3	289	5.299	1.94915×10^{-8}
13	585	15.214	7.66092×10^{-10}	4	1089	31.569	7.59877×10^{-10}

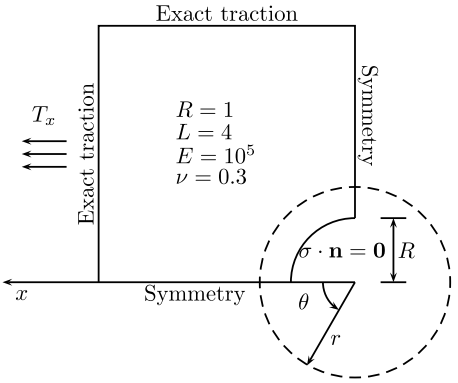


Fig. 29. Elastic plate with a circular hole: problem definition.

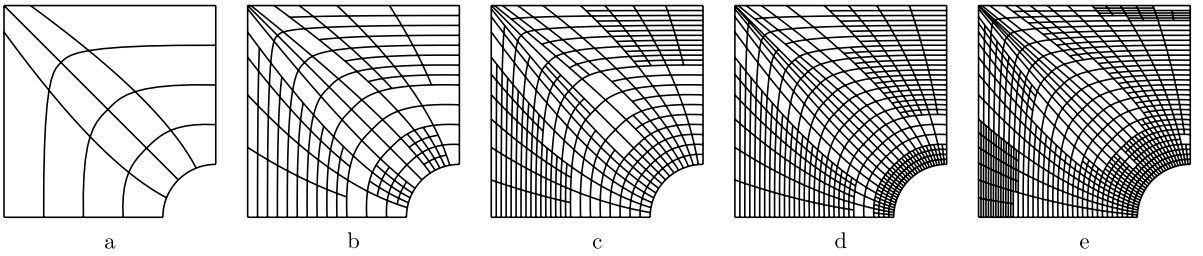


Fig. 30. Resulting meshes after 1, 5, 10, 14, 18 refinements.

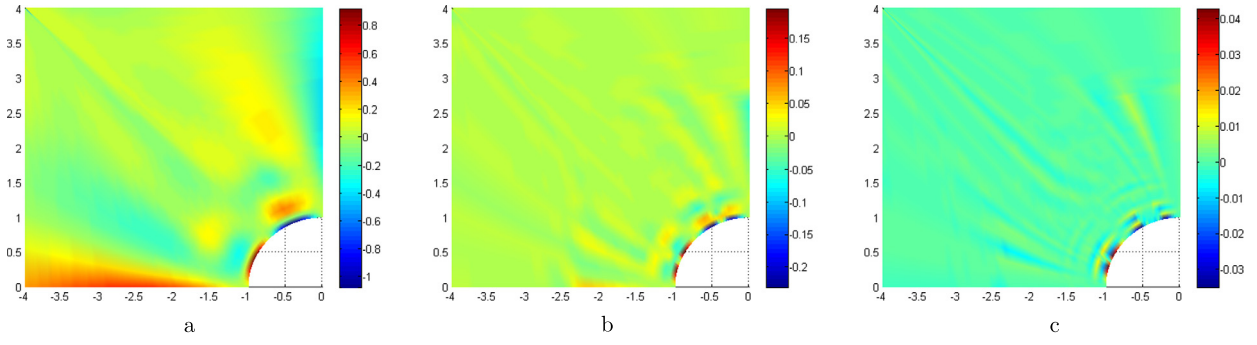


Fig. 31. The xy-direction stress error after 1, 5, 10 refinements.

Fig. 32 and Table 2 both show that the adaptive refinement reaches a given tolerance with slightly fewer degrees of freedom than the uniform refinement with NURBS. This phenomenon has be shown in Dorfel et al. (2010), Vuong et al. (2011), Wang et al. (2011b).

8.2.2. Stationary heat conduction: L-domain

We consider a conduction equation in this part. The geometry of the domain is illustrated in Fig. 33, where $\Omega = [-1, 1] \times [-1, 1] \backslash [0, 1] \times [0, 1]$. The heat conduction equation is $-\Delta u = 0$ in Ω with a homogeneous condition on Γ_D, and Neumann

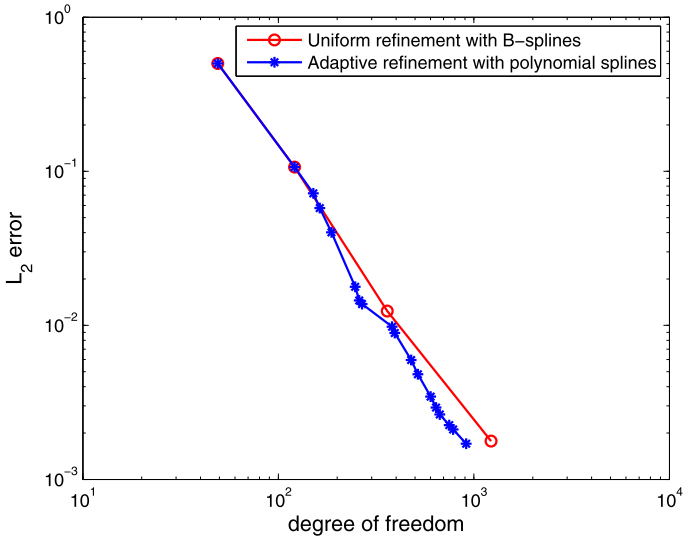


Fig. 32. Comparison of the L^2 error.

Table 2
Experiment data.

Spline over hierarchical T-meshes				B-splines over tensor-product meshes			
n	dim	$t(s)$	error	n	dim	$t(s)$	error
1	49	0.359	0.50037	1	49	0.359	0.50037
2	121	0.639	0.10634	2	121	0.639	0.10634
8	268	2.857	0.013747	3	361	1.543	0.01238
18	913	17.194	0.0017056	4	1225	20.816	0.00177236

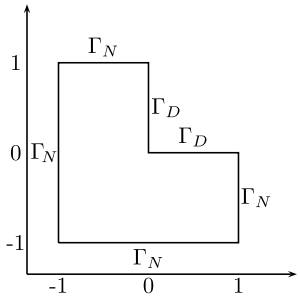


Fig. 33. Figure for the stationary heat conduction.

boundary condition $\frac{\partial u}{\partial \mathbf{n}} = \frac{\partial f}{\partial \mathbf{n}}$ on Γ_N, where $\mathbf{n}$ is outer normal direction, f is an exact solution of the problem and is given here for reference.

$$f = \begin{cases} (x^2+y^2)^{\frac{1}{3}} \sin((2\theta-\pi)/3), & (x,y) \neq (0,0), \\ 0, & (x,y) = (0,0), \end{cases}$$

where

$$\theta = \begin{cases} \arccos(x/r), & x \leq 0 \quad \text{and} \quad y \geq 0, \\ \pi - \arcsin(y/r), & x \leq 0 \quad \text{and} \quad y \leq 0, \\ 2\pi + \arcsin(y/r), & x \geq 0 \quad \text{and} \quad y \leq 0. \end{cases}$$

Fig. 34 shows the adaptive hierarchical T-meshes after 1, 6 and 12 refinements. Fig. 35 gives the convergence results for $S^3(\mathscr{T})$ over hierarchical T-meshes and for $S^3(\mathscr{T})$ over tensor-product meshes. In Table 3, we show the iteration number (n), the dimensions of the spline spaces, the CPU time and the error $\|u_{\text{exact}} - u^{\text{h}}\|_{\text{L}^2}$ for $S^3(\mathscr{T})$ over T-meshes and $S^3(\mathscr{T})$ over tensor-product meshes.

For this example, the advantage of the adaptive refinement is distinct. From the two examples of the application on IGA, we know that whether the advantage of the adaptive refinement is distinct or not is based on the property of the solution.

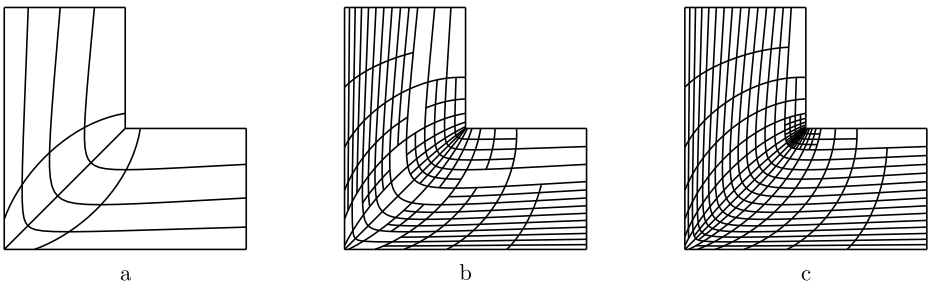


Fig. 34. Resulting meshes after 1, 6, 12 refinements.

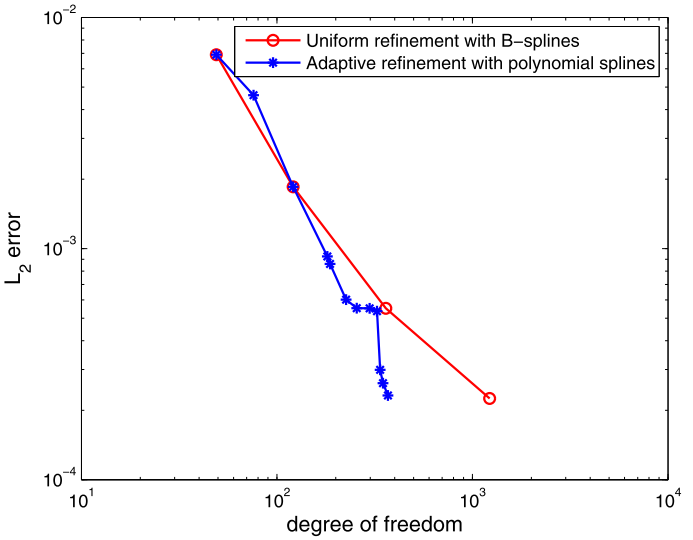


Fig. 35. Comparison of the L^2 error.

Table 3
Experiment data.

Spline over hierarchical T-meshes				B-splines over tensor-product meshes			
n	dim	$t(s)$	error	n	dim	$t(s)$	error
1	49	0.358	0.00690528	1	49	0.358	0.00690528
3	121	0.691	0.00185187	2	121	0.544	0.00185187
6	226	1.538	0.000602485	3	361	1.260	0.00055188
12	370	4.237	0.000231736	4	1225	18.686	0.000224806

8.3. Surface fitting

Given an open surface triangulation with vertices V_j, $j = 1, \ldots, N$ in 3D space, the corresponding parameter values (u_j, v_j), $j = 1, \ldots, N$ are obtained from shape-preserving parametrization in Floater (1997), and the parameter domain is assumed to be $[0, 1] \times [0, 1]$. To construct a spline to fit the given surface, we need only to compute all the control points P_i, $i = 1, \ldots, m$ associated with all the basis functions b_i, $i = 1, \ldots, m$. We denote the fitting spline $\sum_{i=1}^{m} P_i b_i(u, v)$ as $S(u, v)$. To find the control points, we just need to solve an over-determined linear system $S(u_j, v_j) = V_j$, $j = 1, \ldots, N$ using least squares fitting. The surface fitting scheme repeats the following two steps until the fitting error in each cell is less than the given tolerance ε:

1. Compute all the control points for all the basis functions.
2. Find all the V_j whose errors are greater than the given error tolerance ε and the nearest crossing vertices in the T-mesh of (u_j, v_j). Refine around these crossing vertices to form a new mesh, and construct the basis for the new mesh. We use $\|S(u_j, v_j) - V_j\|$ to define the fitting error of V_j.

Fig. 36 is the fitting result of the female head. The iteration number (n), the dimensions of the spline spaces, the max error and the CPU time are shown in Table 4.

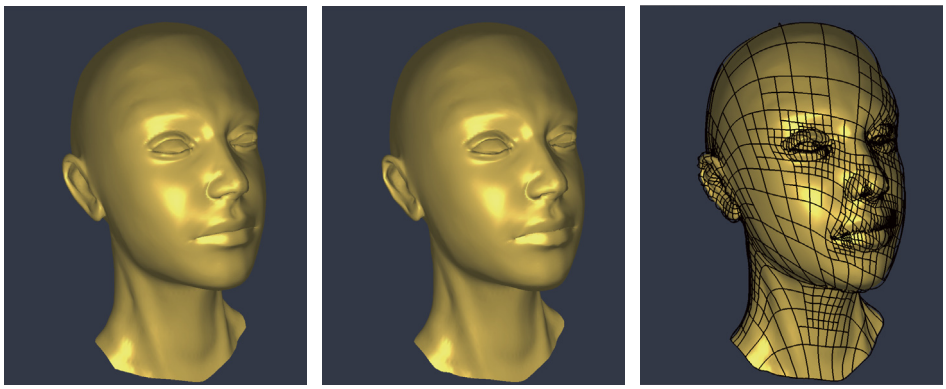


Fig. 36. Fitting female head with $S^3(\mathscr{T})$ over a hierarchical T-mesh.

Table 4
Experiment data.

n	dim	max error	$t(s)$
1	49	0.913	0.328
3	148	0.388	2.464
6	205	0.175	8.176
9	594	0.065	32.715
12	1036	0.025	96.130

9. Conclusions and future work

We discuss the completeness of the bicubic spline spaces over hierarchical T-meshes. The rules of refinement to guarantee the completeness are obtained. Under these rules, the basis are constructed, which have many useful properties for application. The new splines are compared with HB-splines and T-splines. We give three examples for the preliminary applications, which show the efficiency of the new basis.

Although we only discuss bicubic splines in this paper, with the conclusions in Deng et al. (2013), we can analyze biquadratic splines similarly. The difference is that the anchors of the basis functions of biquadratic splines are cells.

In the future, more applications will be explored. The 3-variate case is also a considerable question.

Acknowledgements

The authors are supported by a NKBRPC (2011CB302400), the National Natural Science Foundation of China (11426236 and 11371341), and the 111 Project (No. b07033).

References

Bai, Y., Wang, D., 2010. On the comparison of trilinear, cubic spline, and fuzzy interpolation methods in the high-accuracy measurements. IEEE Trans. Fuzzy Syst. 18, 1016–1022.
Bazilevs, Y., Calo, V.M., Cottrell, J.A., Evans, J.A., Hughes, T.J.R., Lipton, S., Scott, M.A., Sederberg, T.W., 2010. Isogeometric analysis using T-splines. Comput. Methods Appl. Mech. Eng. 199, 229–263.
Berdinsky, D., Kim, T., Bracco, C., Cho, D., Mourrain, B., Oh, M., Kiatpanichgij, S., 2014. Dimensions and bases of hierarchical tensor-product splines. J. Comput. Appl. Math. 257, 86–104.
Berdinsky, D., Kim, T., Cho, D., Bracco, C., Kiatpanichgij, S., 2015. Bases of T-meshes and the refinement of hierarchical B-splines. Comput. Methods Appl. Mech. Eng. 283, 841–855.
Buffa, A., Cho, D., Sangalli, G., 2010. Linear independence of the T-spline blending functions associated with some particular T-meshes. Comput. Methods Appl. Mech. Eng. 199, 1437–1445.
Deng, J., Chen, F., Feng, Y., 2006. Dimensions of spline spaces over T-meshes. J. Comput. Appl. Math. 194, 267–283.
Deng, J., Chen, F., Li, X., Hu, C., Tong, W., Yang, Z., Feng, Y., 2008. Polynomial splines over hierarchical T-meshes. Graph. Models 74, 76–86.
Deng, J., Chen, F., Jin, L., 2013. Dimensions of biquadratic spline spaces over T-meshes. J. Comput. Appl. Math. 238, 68–94.
Desquilbeta, L., Mariottib, F., 2010. Dose-response analyses using restricted cubic spline functions in public health research. Stat. Med. 29, 1037–1057.
Dokken, T., Lyche, T., Pettersen, K.F., 2013. Polynomial splines over locally refined box-partitions. Comput. Aided Geom. Des. 30, 331–356.
Dorfel, M., Jüttler, B., Simeon, B., 2010. Adaptive isogeometric analysis by local h-refinement with T-splines. Comput. Methods Appl. Mech. Eng. 199, 264–275.
Floater, M.S., 1997. Parameterization and smooth approximation of surface triangulations. Comput. Aided Geom. Des. 14, 231–250.
Forsey, D., Bartels, R., 1988. Hierarchical B-spline refinement. Comput. Graph. 22, 205–212.
Giannelli, C., Jüttler, B., Speleers, H., 2012. THB-splines: the truncated basis for hierarchical splines. Comput. Aided Geom. Des. 29, 485–498.
Giannelli, C., Jüttler, B., 2013. Bases and dimensions of bivariate hierarchical tensor-product splines. J. Comput. Appl. Math. 239, 162–178.

Johannessen, K.A., Kvamsdal, T., Dokken, T., 2014. Isogeometric analysis using LR B-splines. Comput. Methods Appl. Mech. Eng. 269, 471–514.
Li, X., Deng, J., Chen, F., 2007. Surface modeling with polynomial splines over hierarchical T-meshes. Vis. Comput. 23 (12), 1027–1033.
Li, X.X., 2012. Numerical solution of fractional differential equations using cubic B-spline wavelet collocation method. Commun. Nonlinear Sci. Numer. Simul. 17, 3934–3946.
Li, X., Scott, M.A., 2014. Analysis-suitable T-splines: characterization, refineability, and approximation. Math. Models Methods Appl. Sci. 24 (06), 1141–1164.
Li, X., Zheng, J., Sederberg, T.W., Hughes, T.J.R., Scott, M.A., 2012. On the linear independence of T-splines blending functions. Comput. Aided Geom. Des. 29, 63–76.
Mohanty, R., Gopal, V., 2011. High accuracy cubic spline finite difference approximation for the solution of one-space dimensional non-linear wave equations. Appl. Math. Comput. 218, 4234–4244.
Mokriš, D., Jüttler, B., Giannelli, C., 2014. On the completeness of hierarchical tensor-product B-splines. J. Comput. Appl. Math. 271, 53–70.
Mokriš, D., Jüttler, B., 2014. TDHB-splines: the truncated decoupled basis of hierarchical tensor-product splines. Comput. Aided Geom. Des. 31, 531–544.
Nguyen-Thanh, N., Nguyen-Xuan, H., Bordas, S.P.A., Rabczuk, T., 2011a. Isogeometric analysis using polynomial splines over hierarchical T-meshes for two-dimensional elastic solids. Comput. Methods Appl. Mech. Eng. 200, 1892–1908.
Nguyen-Thanh, N., Kiendl, J., Nguyen-Xuan, H., Wchner, R., Bletzinger, K.U., Bazilevs, Y., Rabczuka, T., 2011b. Rotation free isogeometric thin shell analysis using PHT-splines. Comput. Methods Appl. Mech. Eng. 200, 3410–3424.
Schumaker, L.L., Wang, L., 2012. Approximation power of polynomial splines on T-meshes. Comput. Aided Geom. Des. 29, 599–612.
Sederberg, T.W., Zheng, J.M., Bakenov, A., Nasri, A., 2003. T-splines and T-NURCCs. ACM Trans. Graph. 22 (3), 477–484.
Sederberg, T.W., Cardon, D.L., Finnigan, G.T., North, N.S., Zheng, J.M., Lyche, T., 2004. T-spline simplification and local refinement. ACM Trans. Graph. 23 (3), 276–283.
Sederberg, T.W., Finnigan, G.T., Li, X., Lin, H., 2008. Watertight trimmed NURBS. ACM Trans. Graph. 27 (3). Article No. 79.
Vuong, A.-V., Giannelli, C., Jüttler, B., Simeon, B., 2011. A hierarchical approach to adaptive local refinement in isogeometric analysis. Comput. Methods Appl. Mech. Eng. 200, 3554–3567.
Wang, R.-H., 2001. Multivariate Spline Functions and Their Applications. Science Press/Kluwer Academic Publishers.
Wang, J., Yang, Z., Jin, L., Deng, J., Chen, F., 2011a. Parallel and adaptive surface reconstruction based on implicit PHT-splines. Comput. Aided Geom. Des. 28, 463–474.
Wang, P., Xu, J., Deng, J., Chen, F., 2011b. Adaptive isogeometric analysis using rational PHT-splines. Comput. Aided Des. 43, 1438–1448.
Wu, M., Xu, J., Wang, R., Yang, Z., 2012. Hierarchical bases of spline spaces with highest order smoothness over hierarchical T-subdivisions. Comput. Aided Geom. Des. 29, 499–509.
Zeng, C., Deng, F., Li, X., Deng, J., 2015. Dimensions of biquadratic and bicubic spline spaces over hierarchical T-meshes. J. Comput. Appl. Math. 287, 162–178.