www.cadanda.com Computer-Aided Design AND APPLICATIONS

The Analysis of T-spline Blending Functions Linear Independence

Wang Ai-zeng and Zhao Gang*

School of Mechanical Engineering & Automation, Beihang University, Beijing, China
* Corresponding author: Zhao Gang (zhaog@buaa.edu.cn)

ABSTRACT

In 2003, Sederberg and Zheng proposed T-splines which belongs non-uniform rational B-splines with T-knot. Compared to B-splines, T-splines have more advantages especially in the surface merging, local refinement, and data compression. The T-spline blending function plays an important role in the T-spline surface modeling. In this paper we give an equivalent condition of the open question that whether T-spline blending functions are linearly independent. According to the equivalent condition the paper presents an algorithm to determine whether T-spline blending functions are linearly independent or not. Finally, we give a sufficient condition of the linear independence, and by using the sufficient condition it can be easily to determine the linear independence of some T-spline blending functions. Several examples are given to illustrate the feasibility and effectiveness of this approach.

Keywords: T-spline, linear independence, surface modeling.
DOI: 10.3722/cadaps.2011.735-745

1 INTRODUCTION

With the development of the computer, the curve and surface modeling technology has been attracting people's research interest. In 1963 Ferguson first proposed the parametric representation of curves and surfaces and since then people have been constantly investigating new methods of the curve and surface representation. In the 1980s non-uniform rational B-splines (NURBS) have become an important tool for describing curves and surfaces in CAD. In 2003, Sederberg and Zheng proposed T-splines which are non-uniform rational B-splines with T-knots[1][2]. The T-knot has make the lines of control points need not traverse the entire control grid, which has broken the limitation that the control grid of the traditional B-spline surface must meet the topological requirement. Compared to B-splines, T-splines have more advantages especially in the surface merging, local refinement, and data compression.

The T-spline method is one of the latest surface modeling techniques. Many scholars have done extensive research in the application and theory of T-splines since the introduction of T-splines in 2003[3][4][5][6]. However several fundamentally theoretical questions about T-splines are still open. In 2004 Sederberg and Zheng posed an open question that whether the T-spline surface blending functions are linearly independent or not and how to classify the T-spline space with T-meshes[2]. In 2006 literature[7] gave a classification of T-splines with T-meshes. In 2007 literature[8] proposed a T-spline local refinement method based on NURBS. In 2010 Buffa gave a proof of the linear independence of T-spline blending functions associated with some particular T-meshes[9]. The linear independence of T-spline surface blending functions plays an important role in T-spline theory. So far, the open question posed by Sederberg and Zheng has not been resolved and this greatly affects the development of the T-spline modeling theory. Based on the above literatures, this paper gives a mathematical proof of the open question.

The paper is organized as follows. In Section 1, T-spline blending functions and the T-space are introduced. In Section 2, the linear independence of T-spline blending functions is analysed and an equivalent condition of the open question is given. Then an algorithm to determine the linear independence of T-spline blending functions is proposed. In Section 3, the result described in Section 2 to present several examples of linearly independent T-splines is illustrated. Finally, the concluding remarks and a prospect about T-splines in Section 4 are given.

2 PREPARATION KNOWLEDGE

T-splines are a generalization of B-splines, it is easily compatible with the existing CAD/CAM system and accepted by the technical staff. In addition, T-spline control grids permit T-junctions, so lines of control points needn't traverse the entire control grids. Therefore it has more applications than NURBS with the rectangle mesh topology. Compared to B-splines, T-splines have more potential in development. From literature[1] we know that the T-spline is a PB-spline for which some order has been imposed on the control points by means of a control grid which is called a T-mesh. Simply if a T-mesh is a rectangular grid with no T-junctions, the T-spline reduces to a B-spline. T-spline control grids permit T-junctions, so lines of control points needn't traverse the entire control grid. This results that T-splines support many valuable operations within a consistent framework. Such as local refinement, data compression and the merging of several B-spline surfaces[1]. While the notion of T-splines can extend to any degree, we restrict our discussion to cubic T-splines $\deg ree = 3$ and default that the knot repeatability is less than $\deg ree + 2$. Then we give the definition of the T-spline and several properties of it.

T-spline surface: $P(u,v) = \sum_{i=1}^{n} P_i B_i(u,v)$, where P_i is the control points, correspondingly, $B_i(u,v)$ is the blending function, $B_i(u,v) = N_{i0}^3(u) N_{i0}^3(v)$, $N_{i0}^3(u)$ and $N_{i0}^3(v)$ are B-spline basis functions associated with the knot vector $\mathbf{U}_i[u_{i0}, u_{i1}, u_{i2}, u_{i3}, u_{i4}]$ and $\mathbf{V}_i[v_{i0}, v_{i1}, v_{i2}, v_{i3}, v_{i4}]$ respectively[1].

Property 1: If a set of T-spline blending functions are defined in the same grid line, then they are linearly independent.

As B-spline basis functions that defined in the same grid line are linearly independent, the T-spline blending functions that constructed by the B-spline basis functions must be linearly independent.

Property 2: Let $B(u,v) = \sum_{i=1}^{M} \mu_i * B_i(u,v)$, $\mu_i > 0$, then the T-spline blending functions $B_i(u,v)$ and $B(u,v)$ cannot be in the same grid line.

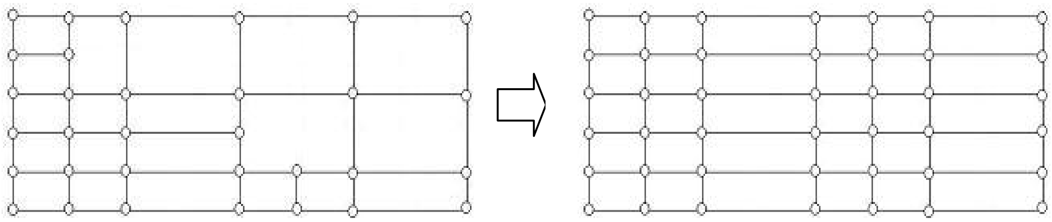
Because $\mu_i > 0$, from the property of B-spline basis functions, if $B_i(u,v)$ and $B(u,v)$ have been defined in the same grid line, then the domain of $B(u,v)$ cannot contain the domain of $B_i(u,v)$. This make a contradiction between them. From this we can get property 2.

Property 3: $B(u,v) = \sum_{i=1}^{M} \omega_i * B_i(u,v)$, $\omega_i \neq 0$, $B(u,v)$ and $B_i(u,v)$ are T-spline blending functions, then $B(u,v)$ and $\left\{ B_i(u,v) \middle| 1 \le i \le M \right\}$ cannot be in the same grid line. If these functions are in the same grid line, then it will contradict with property 1.

T-spline space: The T-spline space is a set of T-splines that have the three common features: 1) the same T-mesh topology; 2) the same knot intervals; 3) the same knot coordinate system.

If local refinement of a T-spline in S_1 will produce a T-spline in S_2, then we say that S_1 is a subspace of S_2, denoted $S_1 \subset S_2$. If T_1 is the T-spline, then $T_1 \in S_1$ means that the topology and knot intervals of T_1 are specified by S_1[2].

The least B-mesh of T-splines (LB-mesh): Throughout all the T-junctions in T-meshes, we can get a B-mesh corresponding with the T-mesh and the B-mesh is the unique and the least rectangle mesh that contains the T-mesh. The mesh is defined as the least B-mesh of T-splines. The T-spline blending functions defined in the LB-mesh are linearly independent. Figure 1 shows a T-mesh(Figure 1.a) and it's LB-mesh(Figure 1.b)



(a) A T-mesh (b) The LB-mesh of the T-mesh

Fig. 1: A T-mesh and its LB-mesh.

From the above definition, each given T-mesh has a unique LB-mesh. Based on the T-spline local refinement algorithm posed by Sederberg and Zheng[2], we can convert a T-spline into a B-spline surface by refinement and keep the surface unchanged. Based on the above relation we will analyse the linear independence of T-spline blending functions in section 2.

3 A PROOF OF THE LINEAR INDEPENDENCE OF T-SPLINE BLENDING FUNCTIONS

3.1 The necessary and sufficient condition

Set the original T-spline surface is T_0, and S_0 is the corresponding T-spline space of T_0, $T_0 \in S_0$. We can get the B-spline surface T_b by the local refinement algorithm of T-splines[2], and set S_b with the LB-mesh is the T-spline space of T_b, $T_b \in S_b$. By the T-spline local refinement algorithm[2], we can get a B-spline surface from the successive refinement. In order to facilitate the explanation, the paper supposes that we have employed several times of the local refinement algorithm to get the B-spline surface. From the first time applying the local refinement algorithm we get the T-spline surface T_1 and the T-spline space S_1 correspondingly, $T_1 \in S_1$. By successively employing the local refinement algorithm, we get the T-spline surface T_i after employing the algorithm i times, and get the T-spline space S_i, $T_i \in S_i$. At last, we get the B-spline surface T_m after employing the algorithm m times and the B-spline space S_m with the LB-mesh correspondingly. Then we have a sequence of T-spline spaces: $S_0 \subset S_1 \subset \cdots \subset S_m(S_b)$, $S_m \equiv S_b$.

Define the original T-spline surface T_0

$$T_0 = \sum_{i=1}^{n0} P_i^0 B_i^0 (u,v) \tag{1}$$

Define the function space V_0 which is expanded by T-spline blending functions $B_j^0(u,v)$

$$V_0 = span\{B_1^0(u,v) \quad B_2^0(u,v) \quad \cdots \quad B_{n0}^0(u,v)\} \tag{2}$$

Similarly define T_i and V_i

$$T_i = \sum_{j=1}^{ni} P_j^i B_j^i(u,v) \tag{3}$$

Where $B_j^i(u,v)$ is the T-spline blending function of T_i.

$$V_i = span\{B_1^i(u,v) \quad B_2^i(u,v) \quad \cdots \quad B_{ni}^i(u,v)\} \tag{4}$$

$$T_m = \sum_{i=1}^{nm} P_i^m B_i^m(u,v) \tag{5}$$

$$V_m = span\{B_1^m(u,v) \quad B_2^m(u,v) \quad \cdots \quad B_{nm}^m(u,v)\} \tag{6}$$

Where T_m defined by the LB-mesh is a B-spline surface which is a special condition of T-splines. From the property of the B-spline we know $B_1^m(u,v) \quad B_2^m(u,v) \quad \cdots \quad B_{nm}^m(u,v)$ are linearly independent, so $\dim V_m = nm$.

In this paper the sequence of $B_j^i \ (1 \le j \le ni)$ is sorted by the row order firstly or the column order firstly. For example: $B^i = \left[B_{11}^i, B_{12}^i, B_{13}^i, B_{21}^i, B_{22}^i, B_{23}^i\right]^T$ or $B^i = \left[B_{11}^i, B_{21}^i, B_{12}^i, B_{22}^i, B_{13}^i, B_{23}^i\right]^T$.

The sequence of adding knot: in the paper each local refinement meets the local refinement algorithm[2] and the order of adding knot corresponds to the sequence of $B_j^m \ (1 \le j \le nm)$.

For example: $B^0 = \left[B_{11}^0, B_{21}^0, B_{22}^0, B_{23}^0\right]^T$, $B^m = \left[B_{11}^2, B_{12}^2, B_{13}^2, B_{21}^2, B_{22}^2, B_{23}^2\right]^T$, then B_{12} is added firstly, and B_{13} is added secondly.

Based on the local refinement algorithm[2], we obtain T_{i+1} from T_i. Compared with the blending function of T_i, the blending function of T_{i+1} is either the same or is changed near the knot which has been added. It is illustrated in equation(7).

$$\begin{cases} B_c^i(u,v) = B_c^{i+1}(u,v) \\ or \\ B_c^i(u,v) = a * B_c^{i+1}(u,v) + b * B_d^{i+1}(u,v) \\ or \\ B_c^i(u,v) = a * B_c^{i+1}(u,v) + \sum_{j=1}^{q} b_{dj} * B_{dj}^{i+1}(u,v) \end{cases} \qquad a, b, b_{dj} \ge 0 \tag{7}$$

From equation(7), we get

$$\begin{bmatrix} B_1^i(u,v) \\ B_2^i(u,v) \\ \vdots \\ B_{ni}^i(u,v) \end{bmatrix} = A_{i+1} * \begin{bmatrix} B_1^{i+1}(u,v) \\ B_2^{i+1}(u,v) \\ \vdots \\ B_{n(i+1)}^{i+1}(u,v) \end{bmatrix} \tag{8}$$

Where A_{i+1} is a non-negative sparse matrix. From the above equation (8) we can get

$$\begin{bmatrix} B_1^0(u,v) \\ B_2^0(u,v) \\ \vdots \\ B_n^0(u,v) \end{bmatrix} = \prod_{i=1}^{m} A_i * \begin{bmatrix} B_1^m(u,v) \\ B_2^m(u,v) \\ \vdots \\ B_{nm}^m(u,v) \end{bmatrix} \tag{9}$$

Let

$$B^i = \begin{bmatrix} B_1^i(u,v) & B_2^i(u,v) & \cdots & B_{ni}^i(u,v) \end{bmatrix}^T \quad (i = 0, \cdots, m) \tag{10}$$

Then equation(8) and equation(9) can be simplified as equation(11) and equation(12).

$$B^i = A_{i+1} * B^{i+1} \quad (i = 0, \cdots, m-1) \tag{11}$$

$$B^0 = \prod_{i=1}^{m} A_i * B^m \tag{12}$$

Let

$$A = \prod_{i=1}^{m} A_i \tag{13}$$

As

$$\dim\left\{B_1^m(u,v), B_2^m(u,v), \cdots, B_{nm}^m(u,v)\right\} = nm \tag{14}$$

then we can see $\dim\left\{B_1^0(u,v), B_2^0(u,v), \cdots, B_n^0(u,v)\right\}$ depends on $R(\prod_{i=1}^{m} A_i)$.

The necessary and sufficient condition of the linear independence of T-spline blending functions: set $B^0 = \prod_{i=1}^{m} A_i * B^m$, if $R(\prod_{i=1}^{m} A_i) = n$, n is the number of T-spline control points, in other word $\prod_{i=1}^{m} A_i$ has full rank, then $\dim(V_0) = n$. $\Leftrightarrow$ T-spline blending functions are linearly independent.

3.2 Algorithm

From the necessary and sufficient condition, for any given T spline we can determine the linear independence of T-spline blending functions automatically.

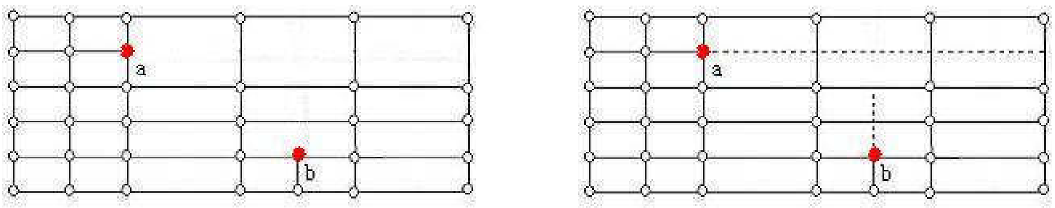
Algorithm: (to determine the linear independence of T-spline blending functions)

step1: Input the T-mesh that corresponds with the T-spline;

step2: Refine the T-spline to the B-spline by the local refinement algorithm and calculate the transformation matrix A_i $(1 \le i \le m)$ for each refinement process;

step3: Determine whether $R(\prod_{i=1}^{m} A_i)$ equals n, where n is the number of T-spline control points. If $R(\prod_{i=1}^{m} A_i) = n$ then the T-spline blending functions are linearly independent. If $R(\prod_{i=1}^{m} A_i) < n$ then the T-spline blending functions are linearly correlative;
End.

From the algorithm, we can determine whether T-spline blending functions are linearly independent by calculating $\prod_{i=1}^{m} A_i$. Here we give a sufficient condition of the linear independence and through the sufficient condition it can be more easily to determine whether the T-spline blending functions are linearly independent or not.



(a) T-mesh 1 (b) The T-knot extension line

Fig. 2: T-mesh 1 and its T-knot extension line.

The Sufficient condition: Given a T-spline, if each T-knot through line doesn't intersect with other T-knot extension lines, then the T-spline blending functions are linearly independent. We are now to prove the sufficient condition. As Figure 2 shown, the dotted line of T-knot a is the T-knot through line; the dotted line of T-knot b is the T-knot extension line.

Proof: From the previous conclusion, suppose the T-spline is refined to the B-spline through inserting knots m times, As each T-knot through line doesn't intersect with other T-knot extension lines, each knot-insertion is a local refinement process and we can apply the local refinement algorithm directly. The following steps are the knot insertion operations:

(1) Based on the previous definition, put the sequence B_j^0 $(1 \le j \le n0)$ in the row order first, for example $B^0 = \left[B_{11}^0, B_{12}^0, B_{13}^0, B_{21}^0, B_{22}^0, B_{23}^0 \right]^T$.

(2) According to the order of B_j^0 $(1 \le j \le n0)$, we determine whether B_j^0 is a T-junction or not. If it isn't a T-junction, then we continue to judge B_{j+1}^0; if it is a T-junction, then we insert knots along the direction of the T-knot extension line until the line throughout the mesh.

(3) The original T-mesh is refined into LB-mesh after several knot insertion operations.

Suppose after inserting knots m times, the original T-spline is refined to the B-spline surfafce $T_m = \sum_{i=1}^{nm} P_i^m B_i^m(u,v)$ and the T-spline space S_m. From equation(6), (10) and (14), we know

$$B^m = \left[B_1^m(u,v) \quad B_2^m(u,v) \quad \cdots \quad B_{nm}^m(u,v) \right]^T$$
$$V_m = span\left\{ B_1^m(u,v) \quad B_2^m(u,v) \quad \cdots \quad B_{nm}^m(u,v) \right\}$$

$$\dim V_m = R\left\{B_1^m(u,v), B_2^m(u,v), \cdots, B_{nm}^m(u,v)\right\} = nm$$

From equation (12), we know $B^0 = \prod_{i=1}^{m} A_i * B^m$. B^0 consists of the original T-spline blending functions, B^m consists of the B-spline basis functions, The definition of B^i and V_i are the same as before.

We now consider in the reversed order: $m \leftarrow m-1 \leftarrow \cdots \leftarrow 1$. Firstly we consider the knot insertion in the last time, i.e. $m \leftarrow m-1$:

$$B^{m-1} = \left[B_1^{m-1}(u,v) \quad B_2^{m-1}(u,v) \quad \cdots \quad B_{n(m-1)}^{m-1}(u,v) \right]^T$$

$$V_{m-1} = span\left\{ B_1^{m-1}(u,v) \quad B_2^{m-1}(u,v) \quad \cdots \quad B_{n(m-1)}^{m-1}(u,v) \right\}$$

We get S_m from S_{m-1} by knot inserting. Then, we define that the function space V_{m-1}^o consists of unchanged T-spline blending functions in S_{m-1} by means of the knot insertion and V_{m-1}^g consists of T-spline blending functions that are changed in S_{m-1} by means of the knot insertion. Definitely we get $V_{m-1}^g + V_{m-1}^o = V_{m-1}$. The new T-spline blending functions by means of the knot insertion constitute V_m^d. We get $V_m^d + V_{m-1}^o = V_m$.

Let

$$V_{m-1}^g = span\left\{ B_{g1}^{m-1}(u,v) \quad B_{g2}^{m-1}(u,v) \quad \cdots \quad B_{gq}^{m-1}(u,v) \right\} \tag{15}$$

$$V_m^d = span\left\{ B_{d1}^m(u,v) \quad B_{d2}^m(u,v) \quad \cdots \quad B_{d(q+1)}^m(u,v) \right\} \tag{16}$$

$$B_g^{m-1} = \left[B_{g1}^{m-1}(u,v) \quad B_{g2}^{m-1}(u,v) \quad \cdots \quad B_{gq}^{m-1}(u,v) \right]^T \tag{17}$$

$$B_d^m = \left[B_{d1}^m(u,v) \quad B_{d2}^m(u,v) \quad \cdots \quad B_{d(q+1)}^m(u,v) \right]^T \tag{18}$$

From the knot insertion algorithm, we know $B_g^{m-1} = T_g^m * B_d^m$, and the transformation matrix T_g^m has full rank, then $V_{m-1}^g \subset V_m^d$, $\dim V_{m-1}^g = \dim V_m^d - 1$.

We know

$$V_m = V_{m-1}^o + V_m^d \tag{19}$$

$$V_{m-1} = V_{m-1}^o + V_{m-1}^g \tag{20}$$

$$V_{m-1} \subset V_m \tag{21}$$

$$\dim V_m = \dim V_{m-1}^o + \dim V_m^d \tag{22}$$

From $V_{m-1}^g \subset V_m^d$, we get

$$\dim V_{m-1} = \dim V_{m-1}^o + \dim V_{m-1}^g \tag{23}$$

From $\dim V_{m-1}^g = \dim V_m^d - 1$, we get

$$\dim V_{m-1} = \dim V_m - 1 \tag{24}$$

It is to say that T-spline blending functions B_j^{m-1} $(1 \le j \le n(m-1))$ are linearly independent.

From the linear independence of $B_j^i\ (1 \le j \le m(j))$, we can induce that $B_j^{i-1}\ (1 \le j \le m(i-1))$ are linearly independent too. Analogously, we can get $\dim(V_0) = n$, that is $R(\prod_{i=1}^{m} A_i) = n$, $\prod_{i=1}^{m} A_i$ has full rank. Based on the necessary and sufficient condition, we know the T-spline blending functions are linearly independent.

The sufficient condition is a special situation of the necessary and sufficient condition, and we can directly determine the linear independence of some special T-spline blending functions using the sufficient condition. When we cannot give the direct determination through the sufficient condition, the necessary and sufficient condition can be used. This section gives a detection algorithm of the linear independence about T-spline blending functions based on the necessary and sufficient condition, and we can determine whether the T-spline blending functions are linear independent or not. The computational complexity of the algorithm is related to matrix $\prod_{i=1}^{m} A_i$. If T-spline has n control points, then the computational complexity of the algorithm is $O(n^3)$.

Inference: If T-mesh lines of a T-spline are all run through along one parameter direction, it is to say the T-mesh only has T-knots in one parameter direction, then the T-spline blending functions are linearly independent(here each T-knot through line doesn't intersect with the others, this inference can be obtained by the sufficient condition).

4 EXAMPLES OF LINEARLY INDEPENDENT T-SPLINES

In this section, we show how section 2 can be used to prove the linear independence of T-spline blending functions. Based on the sufficient condition given in section 2 we can easily determine whether the T-spline blending functions are linearly independent or not. We can also use the algorithm based on the necessary and sufficient condition in the previous section. Although sometimes the calculation is complex, it is without loss of generality in determination and we can consider this algorithm when the sufficient condition is invalid.

In this section the sequence of T-space is obtained by applying the local refinement algorithm and a new T-mesh is always a refinement of the given T-mesh. For clarity, this paper draw the underlying T-mesh in the parametric space and choose it to be tensor product for simplicity. $B_{ij}^k(u,v)$ is the T-spline blending function of T-space that has employed the local refinement algorithm k times. Figure 3.a is the original T-mesh; Figure 3.b is the T-mesh being simplified; after employing applying the local refinement algorithm 1 time, we get Figure 3.c; after employing the local refinement algorithm 2 times, we get B-mesh(LB-mesh: Figure 3.d). From the sufficient condition of section 2 we know the T-spline blending functions are linearly independent. Without loss of generality, we can also use the algorithm based on the necessary and sufficient condition in section 2 to prove and give the calculation process, which is to determine whether the transformation matrix $A = \prod_{i=1}^{m} A^i$ has full rank.

Let

$$V_0 = span\left\{B_{11}^0(u,v) \quad B_{12}^0(u,v) \quad \cdots \quad B_{45}^0(u,v)\right\} \tag{25}$$

$$V_1 = span\left\{B_{11}^1(u,v) \quad B_{12}^1(u,v) \quad \cdots \quad B_{45}^1(u,v)\right\} \tag{26}$$

$$V_2 = span\left\{B_{11}^2(u,v) \quad B_{12}^2(u,v) \quad \cdots \quad B_{45}^2(u,v)\right\} \tag{27}$$

$$B^0 = \left[B_{11}^0(u,v) \quad B_{12}^0(u,v) \quad \cdots \quad B_{45}^0(u,v)\right]^T \tag{28}$$

$$B^1 = \left[B_{11}^1(u,v) \quad B_{12}^1(u,v) \quad \cdots \quad B_{45}^1(u,v)\right]^T \tag{29}$$

$$B^2 = \left[B_{11}^2(u,v) \quad B_{12}^2(u,v) \quad \cdots \quad B_{45}^2(u,v) \right]^T \tag{30}$$

Compared to B^1, only $B_{14}^0, B_{34}^0, B_{44}^0$ have changed in B^0, Similarly $B_{15}^1, B_{35}^1, B_{45}^1$ have changed in B^1 compared to B^2.

$$\begin{bmatrix} B_{14}^0 \\ B_{34}^0 \\ B_{44}^0 \end{bmatrix} = \begin{bmatrix} \frac{u_2 - u_{-1}}{u_3 - u_{-1}} & \frac{u_4 - u_2}{u_4 - u_0} & 0 & 0 \\ 0 & \frac{u_2 - u_0}{u_4 - u_0} & \frac{u_5 - u_2}{u_5 - u_1} & 0 \\ 0 & 0 & \frac{u_2 - u_1}{u_5 - u_1} & 1 \end{bmatrix} \begin{bmatrix} B_{14}^1 \\ B_{24}^1 \\ B_{34}^1 \\ B_{44}^1 \end{bmatrix} \tag{31}$$

$$\begin{bmatrix} B_{15}^1 \\ B_{35}^1 \\ B_{45}^1 \end{bmatrix} = \begin{bmatrix} \frac{u_2 - u_{-1}}{u_3 - u_{-1}} & \frac{u_4 - u_2}{u_4 - u_0} & 0 & 0 \\ 0 & \frac{u_2 - u_0}{u_4 - u_0} & \frac{u_5 - u_2}{u_5 - u_1} & 0 \\ 0 & 0 & \frac{u_2 - u_1}{u_5 - u_1} & 1 \end{bmatrix} \begin{bmatrix} B_{15}^2 \\ B_{25}^2 \\ B_{35}^2 \\ B_{45}^2 \end{bmatrix} \tag{32}$$

From (31) and (32), we know

$$B^0 = A^1 * B^1 \tag{33}$$

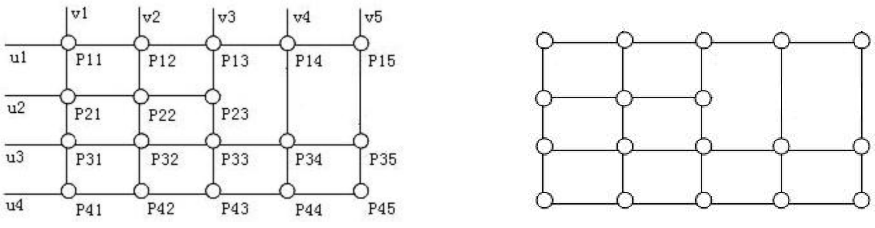
$$B^1 = A^2 * B^2 \tag{34}$$

$$\Rightarrow B^0 = A^1 * A^2 * B^2 \tag{35}$$

$$A = \prod_{i=1}^{2} A^i \tag{36}$$

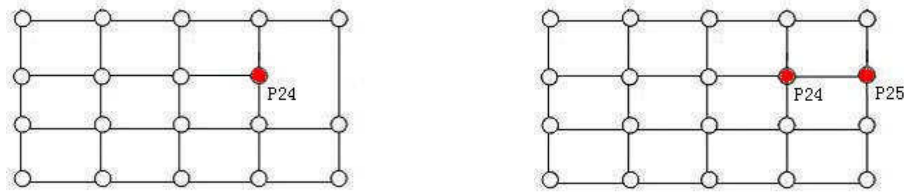
We can verify that the matrix A has full rank, $R(A) = n = 18 \Rightarrow \dim(V_0) = n = 18$, n is the number of T-spline control points. So $B_{11}^0(u,v) \quad B_{12}^0(u,v) \quad \cdots \quad B_{45}^0(u,v)$ are linearly independent.

Figure 4.a is a more complex T-mesh, and figure 4.b is the corresponding B-mesh (LB-mesh). By the sufficient condition of section 2, we know that the T-spline blending functions are linearly independent. We can also use the algorithm in section 2 to determine whether the T-spline blending functions are linearly independent.



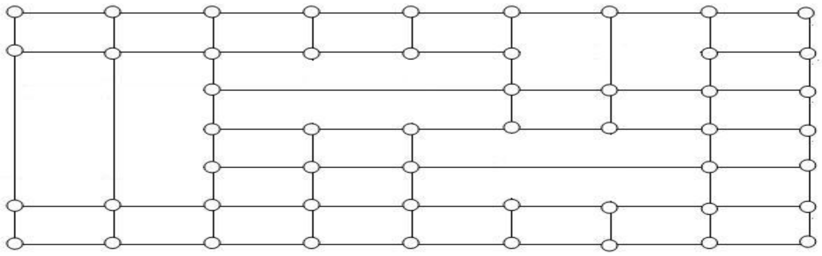
(a) Original T-mesh

(b) T-mesh being simplified

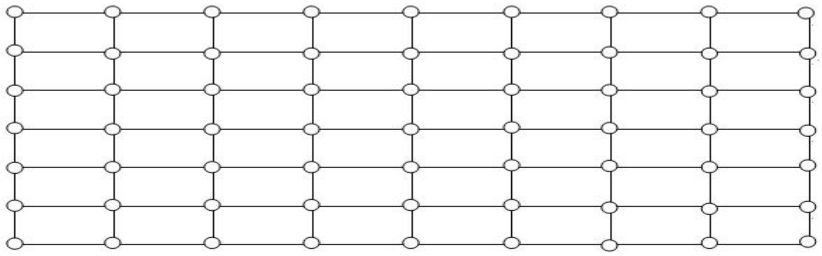


(c) The T-mesh by local (d) LB-mesh

Fig. 3: Process of converting a T-mesh to its LB-mesh.



(a) A complex T-mesh



(b) LB-mesh of the complex T-mesh

Fig. 4: A T-mesh and its corresponding B-mesh.

5 CONCLUSION AND FUTURE WORK

Based on the theoretical analysis of the open question that raised by Sederberg and Zheng, the paper gives the equivalent condition of the linear independence of T-spline blending functions and proposes an algorithm to verify whether T-spline blending functions are linearly independent or not. At last the paper gives an sufficient condition of linear independence. The linear independence of T-spline blending functions plays an important role in the modeling and geometric computing based on T-spline. Hence in the future, we would like to investigate a multiresolution method for T-spline based on the conclusion of the paper.

Acknowledgements

This work was supported by the National 863 plans projects of China (Project Number 2009AA043306).

REFERENCES

[1] Sederberg, T.W.; Zheng, J.; Bakenov, A.; Nasri, A.: T-splines and T-NURCCs, ACM Transaction on Graphics, 22(3), 2003, 477-484. [doi>10.1145/882262.882295]

[2] Sederberg, T.W.; Cardon, D.L.; Zheng, J.; Lyche, T.: T-spline simplification and local refinement, ACM Transactions on Graphics, 23(3), 2004, 276-283. [doi>10.1145/1015706.1015715]

[3] Song, W.; Yang, X.: Free-form deformation with weighted T-spline, The Visual Computer, 21(3), 2005, 139-151. [doi>10.1007/s00371-004-0277-8]

[4] Wang, Y.; Zheng, J.: Control point removal algorithms for T-spline surfaces, Geometric Modeling and Processing, 4077, 2006, 385-396. [doi>10.1007/11802914_27]

[5] Bazilevs, Y.; Calo, V.; Cottrell, J.; Evans, J.; Hughes, T.; Lipton, S.; Scott,; Sederberg, M.T.: Isogeometric analysis using T-splines, Comput. Methods Appl. Mech. Engrg. 199(5-8),2010, 229-263. [doi>10.1016/j.cma.2009.02.036]

[6] Dorfel, M.; Jüttler, B.; Simeon, B.: Adaptive isogeometric analysis by local h-refinement with T-splines, Comput. Methods Appl. Mech. Engrg. 199(5-8),2010, 264-275. [doi>10.1016/j.cma.2008.07.012]

[7] Sederberg, T.W.: On A Conjecture by Zheng, 2006, http://www.tsplines.com/technology /jc.pdf.

[8] Bailey, E.; Carayon, S.: Beyond NURBS: enhancement of local refinement through T-splines, in proceedings of SPIE, Volume: 6670, 2007, 66700J-66700J-10. [doi>10.1117/12.735584]

[9] Buffa, A.; Cho, D.; Sangalli, G.: Linear independence of the T-spline blending functions associated with some particular T-meshes, Comput. Methods Appl. Mech, Engrg, 199(23-24), 2010, 1437-1445. [doi>10.1016/j.cma.2009.12.004]