ELSEVIER

Computer Aided Geometric Design 17 (2000) 863–877

COMPUTER AIDED GEOMETRIC DESIGN

www.elsevier.com/locate/comaid

Efficient evaluation of triangular B-spline surfaces

Michael Franssen [a,*], Remco C. Veltkamp [b], Wieger Wesselink [a]

[a] *Technical University of Eindhoven, Department Computing Science, Den Dolech 2, 5612 AZ Eindhoven, The Netherlands*

[b] *Utrecht University, Department Computing Science, Padualaan 14, 3584 CH Utrecht, The Netherlands*

Received October 1999; revised May 2000

Abstract

Evaluation routines are essential for any application that uses triangular B-spline surfaces. This paper describes an algorithm to efficiently evaluate triangular B-spline surfaces with arbitrary many variables. The novelty of the algorithm is its generality: there is no restriction on the degree of the B-spline surfaces or on the dimension of the domain. Constructing an evaluation graph allows us to reuse partial results and hence, to decrease computation time. Computation time gets reduced even more by making choices in unfolding the recurrence relation of simplex splines such that the evaluation graph becomes smaller. The complexity of the algorithm is measured by the number of leaves of the graph.

Keywords: Evaluation algorithms; Triangular B-splines

1. Introduction

Evaluation routines for splines are important for any application that uses splines. These applications vary from scattered data approximation to variational surface modeling or 3D-morphing applications. In the end, the resulting spline is always sampled to compute the results or for visualization.

Efficient evaluation schemes have been developed and implemented for many classes of splines, e.g., for Bezier-surfaces (Bézier, 1972) and B-patches (Seidel, 1991). However, for triangular B-spline surfaces, which are widely used for their many desirable properties, efficient evaluation routines are restricted to the quadratic bivariate case (Fong and Seidel, 1993; Pfeifle and Seidel, 1994).

* Corresponding author.

E-mail addresses: mfranssen@usa.net (M. Franssen), Remco.Veltkamp@cs.uu.nl (R.C. Veltkamp), wieger@win.tue.nl (W. Wesselink).

PII: S0167-8396(00)00030-3

In this paper we present an algorithm for efficient evaluation of triangular B-spline surfaces as introduced by Dahmen, Michelli and Seidel (1992). The novelty of this algorithm is that it works for triangular B-spline surfaces of arbitrary degree and with an arbitrary number of dimensions of the domain. For simplicity, however, the main part of this paper concentrates on the bivariate case. The generalization towards arbitrary domains is discussed in Section 7.

Efficiency is obtained by re-using partial results. When Grandine (1987) attempted this approach, he found that tabulating those partial results for reuse is more costly than simply re-computing the required value. He attributes this to the need of the entire knot-set to identify a simplex spline. Pfeifle and Seidel (1994) use a triple of integers to identify the simplex splines encountered during evaluation of a triangular B-spline surface of degree 2. Unfortunately, their numbering does not scale up to higher order triangular B-spline surfaces. In this paper it is shown that the identification problem can be avoided by constructing a directed graph (an evaluation graph) representing the simplex splines and B-spline surfaces. In Section 3 we describe how this graph is built and how it is used to avoid multiple evaluation of simplex splines.

To further reduce computation cost, we cut down on the number of partial results that are required. This is done by using our degrees of freedom when unfolding the recurrence relation for simplex splines.

Since our selection scheme is designed to reduce the size of the evaluation graph, it may produce ill-shaped triangles as domain for constant simples splines. The numerical problems arising from this can be solved by treating linear simplex splines separately (see (Franssen, 1995)) or by using a software library to represent numbers accurately (see (Mehlhorn and Näher, 1999)).

The selection scheme for simplex splines is described in Section 4 and the selection scheme for triangular B-spline surfaces is described in Section 5. The complexity of the algorithm is computed in Section 6 and the results are discussed in Section 8.

2. Definitions

The notations used for simplex- and triangular B-spline surfaces differ in some papers. Therefore, we briefly review the definition of the splines we consider in this paper. This section assumes that the reader is already familiar with triangular B-spline surfaces.

2.1. General definitions

Definition 1 (*Determinant of points*). Let $V = (v_0, v_1, v_2)$ be a triple of points in $\mathbb{R}^2$. Then the determinant of V, denoted as $\det(V)$ is defined as

$$\det(V) = \det \begin{pmatrix} 1 & 1 & 1 \\ v_{0x} & v_{1x} & v_{2x} \\ v_{0y} & v_{1y} & v_{2y} \end{pmatrix}. \tag{1}$$

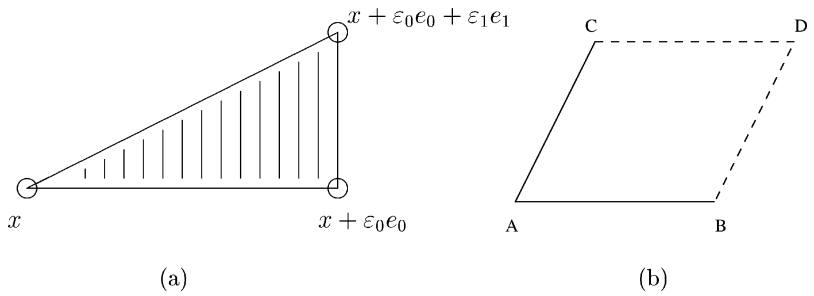

Fig. 1. Sketches to illustrate the half-open convex hull.

Definition 2 (*Barycentric determinant*). Let V be a triple of points in $\mathbb{R}^2$ in general position and let x be a point in $\mathbb{R}^2$. Then the ith barycentric determinant of x $(0 \leqslant i \leqslant 2)$ is defined as

$$d_i(x \mid V) = \det(V[x/v_i]), \tag{2}$$

where $V[x/v_i]$ denotes the set V in which v_i is replaced by x.

Definition 3 (*Barycentric coordinates*). Let V be a triple of points in $\mathbb{R}^2$ and let x be a point in $\mathbb{R}^2$. Then the ith barycentric coordinate of x $(0 \leqslant i \leqslant 2)$ is defined as

$$\lambda_i(x \mid V) = \frac{d_i(x \mid V)}{\det(V)}. \tag{3}$$

Barycentric coordinates have the following important properties:

- $\sum_{i=0}^{2} \lambda_i(x \mid V) = 1$ and $\displaystyle\sum_{i=0}^{2} \lambda_i(x \mid V) v_i = x$.
- If x lies within the convex hull of V, then $0 \leqslant \lambda_i(x \mid V)$ for $0 \leqslant i \leqslant 2$.

Definition 4 (*Half-open convex hull*). Let V be a set of points in $\mathbb{R}^2$ and let e_i denote the unit vector for dimension i for $i = 0, 1$. Then the half-open convex hull of V is defined as

$$[V) = \left\{x \in [V] \mid \exists_{\varepsilon_0,\varepsilon_1>0}\left(\forall_{0\leqslant\alpha_1\leqslant\alpha_0\leqslant 1}(x + \alpha_0\varepsilon_0 e_0 + \alpha_1\varepsilon_1 e_1 \in [V])\right)\right\}, \tag{4}$$

where $[V]$ denotes the convex hull of V. I.e., x belongs to the half-open convex hull $[V)$ of V, if there exists a small triangle as depicted in Fig. 1(a), that lies entirely within the convex hull $[V]$ of V. For instance, in Fig. 1(b), edges (A, B) and (A, C) belong to $[A, B, C, D)$; but edges (B, C) and (C, D) do not.

The half-open convex hull is a generalization of the half-open domain in $\mathbb{R}$. Its purpose is to ensure that for any subdivision of a domain in $\mathbb{R}^2$, the points on the edges of the subdivision belong to exactly one sub-area.

2.2. Definition of simplex splines

Definition 5 (*Simplex splines*). A simplex spline is a piecewise polynomial function defined by a finite set V of points in $\mathbb{R}^2$. The points in V are called *knots* and the set

V itself is called the *knot-set* of the simplex spline. A simplex spline defined over a set of $n+3$ knots is a piecewise polynomial of degree n. The definition of a simplex spline is given by the following recursive equation:

$$M(x \mid V) = \begin{cases} 0 & x \notin [V), \\ \dfrac{1}{|\det(V)|} & |V| = 3 \text{ and } x \in [V), \\ \displaystyle\sum_{i=0}^{2} \lambda_i(x \mid W) M(x \mid V \setminus \{w_i\}) & |V| > 3. \end{cases} \tag{5}$$

The elements in $W = (w_0, w_1, w_2)$ can be chosen arbitrarily from V, hence $W \subset V$. W is called the *split set* for V. The only restriction is that $\det(W)$ may not be zero.

If all knots are in general position, i.e., the knot-set does not contain a collinear triple of knots, a simplex spline of degree n defined over these knots is C^{n-1} continuous. For more information about simplex splines, we refer to Traas (1990).

2.3. Definition of triangular B-spline surfaces

Definition 6 (*Triangular B-spline surfaces*). A triangular B-spline surface is a piecewise polynomial function defined over an arbitrary polygonal domain in $\mathbb{R}^2$. For clarity, we present the construction of a triangular B-spline surface in a number of steps:

(1) One starts by constructing a triangulation $\mathcal{I}$ of the polygonal domain. This triangulation has to be proper, i.e., triangles may not overlap and they can only share a single edge or a single vertex.

(2) Assign to every vertex v_i occurring in the triangulation $n+1$ knots, denoted by $v_{i,0}, \ldots, v_{i,n}$, where n is the degree of the triangular B-spline surface, such that $v_i = v_{i,0}$. There are two important restrictions on the placement of these knots:
   (a) For every edge (v_i, v_j) at the boundary of the polygonal domain, the entire area $[\{v_{i,0}, \ldots, v_{i,n}, v_{j,0}, \ldots, v_{j,n}\})$ must lie outside the polygonal domain.
   (b) For every triangle $I = (i_0, i_1, i_2)$ in $\mathcal{I}$, the determinants $\det(v_{i_0,k}, v_{i_1,l}, v_{i_2,m})$ with $k + l + m \leqslant n$ must have the same sign.

   Often these requirements are not mentioned, even though they are essential to guarantee the desired B-spline surface properties. More information on these restrictions can be found in (Franssen, 1995).

(3) Let $I = (i_0, i_1, i_2)$ be a triangle in $\mathcal{I}$. Let β be a triple of indices $(\beta_0, \beta_1, \beta_2)$ such that $|\beta| = \beta_0 + \beta_1 + \beta_2 = n$ and $\beta_j \geqslant 0$. Then the set V_β^I, containing $n+3$ knots, is defined as

$$V_\beta^I = \{v_{(i_0,0)}, \ldots, v_{(i_0,\beta_0)}, v_{(i_1,0)}, \ldots, v_{(i_1,\beta_1)}, v_{(i_2,0)}, \ldots, v_{(i_2,\beta_2)}\}. \tag{6}$$

   Each of these V_β^I will serve as the knot-set of a simplex spline needed to define a triangular B-spline surface.

(4) To use the simplex splines defined over the sets V_β^I as a basis for a triangular B-spline surface, we have to 'normalize' them. That is, we have to multiply every

simplex spline $M(x \mid V_\beta^I)$ with the factor $d_\beta^I = \det(v_{i_0,\beta_0}, v_{i_1,\beta_1}, v_{i_2,\beta_2})$. As a result, for every point x in the polygonal domain, we get

$$\left(\sum_{I\in\mathcal{I}}\sum_{|\beta|=n} d_\beta^I M\big(x \mid V_\beta^I\big)\right) = 1. \tag{7}$$

Hence, the normalized simplex splines form a partition of unity, making control points easy to use.

(5) For every triangle I and every triple of indices β with $|\beta| = n$, we define a control point c_β^I in $\mathbb{R}^3$. The triangular B-spline surface is then defined as

$$F(x) = \sum_{I\in\mathcal{I}}\sum_{|\beta|=n} d_\beta^I M\big(x \mid V_\beta^I\big)c_\beta^I. \tag{8}$$

Since the normalization factors and the control points do not depend on the evaluation point x, they need not to be considered in the evaluation algorithm. The normalization factors are pre-computed once and the control coefficients are typically set (indirectly) by the user of the application.

3. Reusing partial results

If we naively evaluate a simplex spline of degree n recursively, the amount of constant simplex splines we encounter will be 3^n. For every unfolding of Eq. (5) for a simplex spline of degree i we have to evaluate 3 simplex splines of degree $i-1$. Since a constant simplex spline is obtained after n unfoldings, we obtain 3^n simplex splines of degree zero.

Not all of the simplex splines of degree i with $0 \leqslant i < n$ that we evaluate during the recursion are different. The knot-set V of a simplex spline of degree n contains $n+3$ knots. Any simplex spline of degree i we encounter by recursively unfolding Eq. (5), has a knot-set V' of $i+3$ knots that is a subset of V. Therefore there exist no more than $\binom{n+3}{i+3}$ different simplex splines of degree i.

Evaluation will be accelerated if every simplex spline of degree i is computed only once. The problem when re-using partial results is the identification of simplex splines, because this requires comparing the entire knot-sets. In this section we will present a data structure that makes identifying simplex splines during evaluation superfluous.

We construct a directed graph, in which every node represents a simplex spline. This graph is built only once (during preprocessing) and then used for all future evaluations. Every simplex spline with degree i greater than 0 has three outgoing edges that connect it with three (different) simplex splines of degree $i-1$. These three simplex splines are determined by choosing a split set W and unfolding the recurrence relation 5 for simplex splines. In our implementation, we treat barycentric determinants as simplex splines of degree -1, such that they too will be computed at most once. Note that we do not need to choose a split set for every point x in which we evaluate the simplex spline: once the graph is built, it can be used to evaluate the simplex spline in arbitrary points.

The graph is used as follows: We number all the evaluation points. Whenever we computed the value of a simplex spline in a point, we store this value, along with the

number of the point, in the node of the simplex spline in the graph. When, during evaluation of the same point, we encounter the same simplex spline through another incoming edge, we do not re-compute the value, but use the stored value instead. To limit memory consumption, we only store the value of the last computed point in each node.

4. Choosing split sets for simplex splines

We can use the number of nodes in the graph as a measure for the efficiency, since the data-structure from the previous section avoids multiple evaluation of simplex splines in this graph. To increase the efficiency, we have to decrease the number of nodes in our graph. Which simplex splines occur in the graph depends on our choice of split sets when unfolding Eq. (5). In this section, we present a selection scheme for split sets that strongly decreases the number of nodes in the graph of a single simplex spline.

Throughout this section we will use $V = \{v_0, \ldots, v_{n+2}\}$ to denote the knot-set of the simplex spline of degree n that we want to evaluate. Furthermore, i will always denote a degree between 0 and $n-1$ of some simplex spline in the graph. For simplicity, we assume that every choice for the split set is legal, i.e., V does not contain a triple of linearly dependent knots. In Section 7, we discuss how this restriction is eliminated.

To minimize the number of simplex splines in the graph, we want to use as few different simplex splines of degree i as possible. Therefore, we want to keep the intersections of different knot-sets as large as possible. By choosing the correct split sets we will then create less different simplex splines, since more simplex splines of lower degree become shared.

This is established as follows: for $0 \leqslant i < j < k \leqslant n+2$ we define $M_{i,j,k}$ to be the set $\{v_i, v_j, v_k, v_{k+1}, \ldots, v_{n+2}\}$. Then we choose $\{v_i, v_j, v_k\}$ as split-set for each $M_{i,j,k}$, yielding the knot-sets $M_{j,k,k+1}$, $M_{i,k,k+1}$, $M_{i,j,k+1}$. Hence, to evaluate a degree n simplex spline, we start with $M_{0,1,2}$ and after m recursive steps we need to evaluate all simplex splines of degree $n-m$ having knot-sets $M_{i,j,m+2}$ with $0 \leqslant i < j < m+2$. Hence, the number of simplex splines of degree $n-m$ in the evaluation graph will be $\binom{m+2}{2}$.

Note that the ordering used for the knot-sets is not important as long as it remains fixed.

5. Choosing split sets for triangular B-spline surfaces

The computation of a triangular B-spline surface requires the evaluation of an entire set of simplex splines instead of just one. Since many knots occur in several of these simplex splines, there is hope that evaluating this set of simplex splines can be done more efficiently than simply evaluating all the simplex splines separately. In this section we will exploit similarities between the knot-sets of the simplex splines to get an efficient evaluation scheme.

Since the triangulation of the domain of the B-spline surface, is arbitrary, we will restrict our attention to the computation of the contribution of a single triangle. For the remainder of this section we will denote this triangle as $I = (i_0, i_1, i_2)$. The same evaluation scheme can then be used for all triangles in the domain. By using a single look-up table of simplex

splines for the entire domain of the B-spline surface, the evaluation time is reduced even further, but we will not discuss this effect in this paper.

We start by introducing the concept of fingerprints. Fingerprints are special subsets of the knot-sets of the simplex splines in the graph of the triangular B-spline surface. They will be used to distinguish two groups of simplex splines in the graph of this B-spline surface: those that contain a fingerprint and those that do not.

Definition 7 (*Fingerprints*). For every triangle $I = (i_0, i_1, i_2) \in \mathcal{I}$ and index β with $|\beta| = n$ we define a fingerprint F_β^I as

$$F_\beta^I = \{v_{i_j,\beta_j} \mid 0 \leqslant j \leqslant 2 \wedge 1 \leqslant \beta_j\}. \tag{9}$$

Hence, the fingerprint F_β^I contains from every knot cloud the knot with the highest second index occurring in V_β^I, provided that this index is at least 1. For example: $F_{(2,0,1)}^I = \{v_{i_0,2}, v_{i_2,1}\}$ (and not $\{v_{i_0,1}, v_{i_0,2}, v_{i_2,1}\}$). In the graph of a triangular B-spline surface of degree n, only the F_β^I with $|\beta| = n$ are called fingerprints.

The name 'fingerprint' is not chosen arbitrarily. A fingerprint F_β^I is the smallest set uniquely identifying the degree n simplex spline $M(. \mid V_\beta^I)$. Hence, if a fingerprint $F \subseteq V$ of some simplex spline $M(. \mid V)$, then F, in conjunction with V_β^I regarded as a triple of knot sequences, uniquely defines a degree n simplex spline, which is the only one whose evaluation requires the evaluation of $M(. \mid V)$.

Lemma 8. *Let $x \in V_\beta^I$ such that $x \in F_\beta^I$. Then there exists a γ with $|\gamma| = |\beta| - 1$ such that $V_\gamma^I = V_\beta^I \setminus \{x\}$.*

Lemma 9. *Let V be the knot-set of a simplex spline in the graph of a triangular B-spline surface of degree $n-1$. Let β be an index with $|\beta| = n$. Then $F_\beta^I \not\subseteq V$.*

Using the concept of fingerprints and the corresponding lemmas, we will define our selection scheme for the evaluation of triangular B-spline surfaces. The evaluation scheme will be based on Lemma 8 and the evaluation scheme for a single simplex spline.

To compute the value of a B-spline surface of degree n, we have to compute $M(x \mid V_\beta^I)$ for all β with $|\beta| = n$. For this computation we will construct a single evaluation graph G_n. Our idea is to use a graph G_{n-1} that efficiently computes all $M(x \mid V_\gamma^I)$ with $|\gamma| = n-1$ and add as few new simplex splines to it as possible to obtain G_n. Initially, we use the evaluation graph for a B-spline surface of degree 0, which consists of a single simplex spline of degree 0. Clearly, this graph is optimal.

Using G_{n-1} as a subgraph of G_n is accomplished by the following choice: If a simplex spline contains a fingerprint then this fingerprint must be a subset of the split set. From this decision and Lemma 8 it follows that evaluating all required simplex splines of degree n (i.e., all simplex splines with a knot-set V_β^I with $|\beta| = n$) requires the evaluation of all simplex splines with a knot-set V_γ^I with $|\gamma| = n-1$. This is done efficiently by a graph G_{n-1} of a B-spline surface of degree $n-1$ defined over the same knot-set, which we

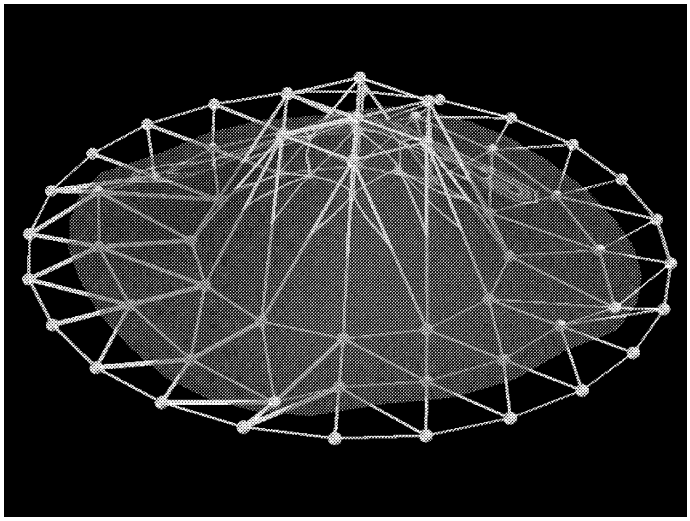

Fig. 2. A B-spline surface of degree 4 with 6 patches.

already have. However, not every fingerprint contains 3 knots, so we still have to select a few elements for the split sets.

The remaining knots of the split sets are chosen in the same way as split sets for single simplex splines are chosen: we use the first $3 - |F_\beta^I|$ knots from $V \setminus F_\beta^I$. To define 'first' we may use an arbitrary ordering on the knots, e.g., alpha-lexicographical ordering on the two indexes. However, once an ordering is chosen, it must be used for all simplex splines used in the evaluation graph. Fig. 2 shows a triangular B-spline surface of degree 4 with 6 patches.

6. Complexity

The simple selection scheme of split sets presented above yields a surprisingly efficient evaluation graph. In order to compute the number of nodes in the graph, we prove the following lemma and theorem:

Lemma 10. *Let V be the knot-set of a simplex spline in the graph G_n of a B-spline surface of degree n, such that $F_\beta^I \subseteq V$. Let v be an element in the split set of V. Then*

- *if $v \notin F_\beta^I$ then $F_\beta^I \subseteq V \setminus \{v\}$;*
- *if $v \in F_\beta^I$ then $M(. \mid V \setminus \{v\})$ exists in the graph G_{n-1} of the B-spline surface of degree $n-1$, where G_{n-1} is defined over the obvious subset of the knot-set of G_n.*

Proof. The first case is trivial. The second case is proved by induction on i, where $n - i$ is the degree of the simplex spline corresponding to V.

case $i = 0$: Given by Lemma 8.

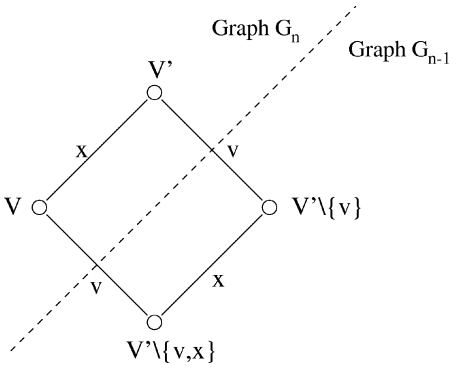

Fig. 3. Sketch of the proof of Lemma 10.

case $i > 0$**:** We need to prove that $M(. \mid V \setminus \{v\})$ exists in the G_{n-1}. Let $M(. \mid V')$ be an ancestor of $M(. \mid V)$, say $V' = V \cup \{x\}$ (see Fig. 3). Then V' contains F_β^I and hence, by the induction hypothesis $M(. \mid V' \setminus \{v\})$ exists in G_{n-1}. $V' \setminus \{v\}$ will contain some fingerprint F_γ^I of a B-spline surface of degree j with $j < n$. Clearly $|F_\gamma^I| \leqslant |F_\beta^I|$ and since x is one of the first $3 - |F_\beta^I|$ elements of V', it will be one of the first $3 - |F^I\gamma|$ elements of $V' \setminus \{v\}$ or it will be an element of F_γ^I. Therefore, x is an element of the split set of $V' \setminus \{v\}$. Now, since $M(. \mid V' \setminus \{v\})$ exists in G_{n-1}, $M(. \mid V' \setminus \{v, x\}) = M(. \mid V \setminus \{v\})$ will also exist in G_{n-1} as was to be proved. □

Theorem 11. *Using the split set selection scheme above, we will now prove that every simplex spline $M(. \mid V)$ in the graph of a degree n B-spline surface*

- *is either a simplex spline in the graph G_{n-1} of a B-spline surface of degree $n - 1$;*
- *or its knot-set V contains a fingerprint F_β^I for certain β with $|\beta| = n$.*

Proof. By induction on i, where $n - i$ is the degree of the simplex spline V:

case $i = 0$**:** $n - i = n$ and hence V contains a fingerprint by definition.

case $i > 0$**:** Let $M(. \mid V')$ be an ancestor of $M(. \mid V)$; i.e., $V = V' \setminus \{x\}$. If $M(. \mid V')$ is in G_{n-1} then $M(. \mid V)$ certainly is. If $M(. \mid V')$ is not G_{n-1} we get from the induction hypothesis that V' contains a fingerprint F_β^I for certain β. If $x \in F_\beta^I$ then by Lemma 10 $M(. \mid V' \setminus \{x\}) = M(. \mid V)$ exists in G_{n-1}. If $x \notin F_\beta^I$, then clearly V contains F_β^I. □

As an example, see Fig. 4: The graph G_1 is a subgraph of G_2. The thicker lines indicate irregularities in the structure. The graph G_3 contains many more of these irregularities and does not clarify matters.

The complexity discussed in this section refers to the complexity of the evaluation and does not include the time required to construct the graph. Constructing the graph is done once during preprocessing using a lookup table. Consulting this lookup table costs time, but once the graph is constructed, the B-spline surface can be evaluated in an arbitrary number of points without ever using the lookup table again.

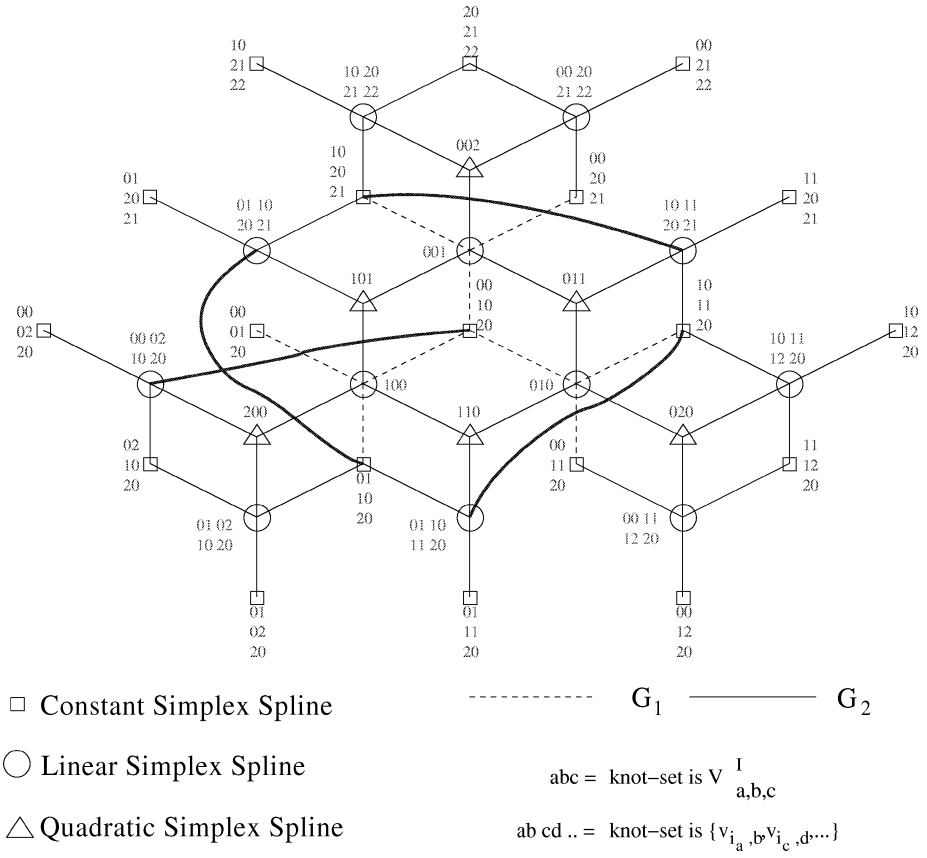

Fig. 4. The graph G_2 of a B-spline surface of degree 2. The dashed lines indicate edges from graph G_1.

We compute the number A_n of constant simplex splines in the evaluation graph G_n. Let B_n denote the number of constant simplex splines in G_n that do not exist in G_{n-1}. Then we find the equation

$$A_n = A_{n-1} + B_n, \tag{10}$$

or written differently

$$A_n = A_0 + \sum_{i=1}^{n} B_i. \tag{11}$$

In a B-spline surface of degree 0 the only simplex spline is already a constant simplex spline, hence $A_0 = 1$. It remains to compute B_n.

From Theorem 11 it follows that every simplex spline in $G_n \setminus G_{n-1}$ contains a fingerprint. Since a simplex spline with fingerprint F_β^I can only be used to compute V_β^I, the graph $G_n \setminus G_{n-1}$ contains one connected component for each β. All simplex splines in such

a subgraph contain the same fingerprint. We will now count for each index β the number B_β of constant simplex splines in the subgraph of splines containing F_β^I. To compute B_n, we then use

$$B_n = \sum_{|\beta|=n} B_\beta. \tag{12}$$

B_β is computed similar to the number of constant simplex splines in the graph of a single simplex spline: write the elements of V_β^I as the elements of F_β^I, followed by the elements of $V_\beta^I \setminus F_\beta^I$ in the order used by the selection algorithm. Suppose that the elements in the mentioned order are denoted by $\{u_0, \ldots, u_{n+2}\}$, then the selection scheme essentially is the same as the one in Section 4 for simplex splines. Hence, we can now define $M'_{i,j,k}$ for $0 \leqslant i < j < k \leqslant n+2$ to be the set $\{u_i, u_j, u_k, u_{k+1}, \ldots, u_{n+2}\}$, such that $F_\beta^I = \{u_0, \ldots, u_f\}$, where $f = |F_\beta^I|$. V_β^I is then equal to $M'_{0,1,2}$. After m recursive steps we get all knot-sets $M'_{i,j,m+2}$ with $0 \leqslant i < j < m+2$. Since we are only interested in constant simplex splines from $G_n \setminus G_{n-1}$, we can use Theorem 11 and only to count the $M'_{i,j,n+2}$ that contain F_β^I. To compute B_n we now sum over all β, distinguishing between $|F_\beta^I|$:

case $|F_\beta^I| = 3$**:** $F_\beta^I \subseteq M'_{i,j,n+2}$ means that i, j and $n+2$ must be 0,1 and 2 respectively. Even after only one recursive step this will never be the case, hence $B_\beta = 0$.

case $|F_\beta^I| = 2$**:** One out of three indices is 0; the other two vary between 1 and $n-1$, summing up to n. Hence, there are $3(n-1)$ of these cases. In each case, the two elements of F_β^I are u_0 and u_1, hence we only need to count $M'_{0,1,n+2}$, yielding $B_\beta = 1$.

case $|F_\beta^I| = 1$**:** There are 3 of these cases: $(n,0,0)$; $(0,n,0)$; and $(0,0,n)$. In each case, we need to count all $M'_{0,i,n+2}$ with $0 < i < n+2$, hence $B_\beta = n+1$.

Hence, $B_n = 3(n-1) + 3(n+1) = 6n$.

We can now finally compute the number A_n of constant simplex splines in the evaluation graph G_n of a B-spline surface of degree n:

$$A_n = 1 + \sum_{i=1}^{n} 6i = 1 + 3n + 3n^2. \tag{13}$$

Hence, to compute the contribution of one triangle $I \in \mathcal{I}$ to the value of the B-spline surface of degree n in a point x, we only have to evaluate $1 + 3n + 3n^2$ constant simplex splines. Since for every k, there will be more simplex splines of degree $k-1$ in the graph than simplex spline of degree k, the number of simplex splines in the graph will be $\mathcal{O}(n^3)$. However, since barycentric determinants are tabulated also, and their number is no bigger than $\binom{3n+3}{2}$, the number of barycentric determinants computed will be $\mathcal{O}(n^2)$.

In Table 1, some experimental results are listed of evaluations of triangular B-spline surfaces. Note that more time is used for each point for surfaces with a domain consisting of 2 triangles (both triangles may contribute to any point), but not twice as much. Every simplex spline is computed once for each point, even if this simplex spline occurs in the graph for another domain-triangle.

Table 1
Time measurements of the implementation, using 10302 points per triangle

Degree	2		3		4		5	
Dom.triangles	1	2	1	2	1	2	1	2
Total time(s)	0.989	3.673	2.314	8.682	4.627	17.182	8.345	30.541
t/point(μs)	96.0	178.3	224.6	421.4	449.1	833.9	810.0	1482.3

7. Generalizations

Although our algorithm can deal with B-spline surfaces of arbitrary degree, there are still some restrictions on its use. In this section we discuss how these restrictions are eliminated.

7.1. Arbitrary knot-sets

One annoying restriction is that all knot-sets V must be in general position, i.e., every triple of knots in V is linearly independent. Sometimes one deliberately introduces a few linearly dependent knot-sets to model splines with continuity less than C^{n-1}.

We can allow arbitrary knot-sets if we consider a special case in our selection scheme for split sets: if a selected split set W is linearly dependent ($\det(W) = 0$), we arbitrarily choose a different split set. If no suitable split set can be found then obviously V is linearly dependent. But then the half-open convex hull of V is empty and hence $M(x \mid V) = 0$ by definition of Eq. (5).

These alternative choices yield a less efficient evaluation graph. Since all determinants are tabulated already, their number will not increase. However, the overload of computing the values of simplex splines from these determinants *will* increase.

7.2. Domains in $\mathbb{R}^s$

Another important restriction is that the algorithm is still limited to domains in $\mathbb{R}^2$. Lifting this restriction to higher dimensional spaces is straightforward, although it requires more work than the previous generalization.

The generalizations of determinants, barycentric determinants, barycentric coordinates and the half-open convex hull are straightforward. Therefore, we only explicitly give those generalizations that affect our algorithm.

Definition 12 (*s-variate simplex splines*). For domains in $\mathbb{R}^s$, simplex splines are defined over sets of at least $s+1$ points in $\mathbb{R}^s$. The recursive equation then becomes:

$$M(x \mid V) = \begin{cases} 0 & x \notin [V), \\ \dfrac{1}{|\det(V)|} & |V| = s+1 \text{ and } x \in [V), \\ \displaystyle\sum_{i=0}^{s} \lambda_i(x \mid W) M(x \mid V \setminus \{w_i\}) & |V| > s+1. \end{cases} \tag{14}$$

But now W is a tuple of $s+1$ knots and hence, the selection algorithm must be extended.

When evaluating a single simplex spline it is sufficient to choose the first $s+1$ elements of each knot-set, like we selected the first 3 knots in the case where $s=2$ (see Section 4). The number of constant simplex splines in the graph then becomes $\binom{n+s}{s}$.

Definition 13 (*s-variate simplicial B-spline spaces*). To define simplicial B-spline spaces we need a subspace $\mathcal{I}$ of $\mathbb{R}^s$ that is properly divided into simploids (subspaces bounded by $s+1$ vertices). "Properly" means that the simploids do not intersect and that if they share an edge, a hyperplane, etc, they share the *entire* edge, hyperplane, etc.

To form a basis for the B-spline space we assign $n+1$ knots $v_{i,0},\ldots,v_{i,n}$ to every vertex v_i in the domain, such that $v_i = v_{i,0}$. Furthermore, let $I \in \mathcal{I}$ be a simploid in the domain and let $\beta = (\beta_0,\ldots,\beta_s)$ be a tuple with $|\beta| = \sum_{i=0}^{s} \beta_i = n$, then V_β^I is defined as

$$\begin{aligned} V_\beta^I &= \{v_{i_0,0},\ldots,v_{i_0,\beta_0},\ldots,v_{i_s,0},\ldots,v_{i_s,\beta_s}\} \\ &= \{v_{i_j,k} \mid 0 \leqslant j \leqslant s \wedge 0 \leqslant k \leqslant \beta_j\}. \end{aligned} \tag{15}$$

After generalizing the normalization factors and defining the control coefficients, the formula for a B-spline space remains exactly the same:

$$F(x) = \sum_{I\in\mathcal{I}} \sum_{|\beta|=n} d_\beta^I M\big(x \mid V_\beta^I\big) c_\beta^I. \tag{16}$$

Hence we have to compute the simplex splines V_β^I for all β with $|\beta| = n$ and $I \in \mathcal{I}$.

Definition 14 (*s-variate fingerprints*). To compute s-variate B-spline spaces we generalize the definition of fingerprints to

$$F_\beta^I = \{v_{i_j,\beta_j} \mid 0 \leqslant j \leqslant s \wedge 1 \leqslant \beta_j\}. \tag{17}$$

Computing the number of constant simplex splines in the graph is done in the same way as for $s=2$. The number of constant simplex splines in the graph of an s-variate B-spline space of degree n is

$$A_n^s = 1 + \sum_{i=1}^{n} \sum_{k=1}^{s+1} \binom{i-1}{k-1} \binom{s+1}{k} \binom{i+s-k}{s-k}, \tag{18}$$

which is a generalization of Eq. (13). The number of s-variate barycentric determinants is limited by $\binom{(s+1)(n+1)}{s}$, which is $\mathcal{O}((sn)^s)$. Since the number of simplex splines in the graph increases as the degree decreases, the total amount of nodes in the graph will be $\mathcal{O}(n(sn)^s)$.

Note that the algorithm described in the previous sections is exactly the special case of the general algorithm for $s=2$.

Pfeifle and Seidel (1995) also introduced a class of spherical triangular B-splines. Since this class of splines uses the same recursive pattern, it is also straightforward to use our algorithm for these splines.

8. Discussion

In this paper we introduced selection schemes for the efficient evaluation of simplex- and triangular B-spline surfaces. In contrast with previous approaches these evaluation schemes are able to deal with splines of arbitrary degree, any number of dimensions in the domain, arbitrary (non-general) knot-placements, and different variants of the B-spline surface scheme. We derived that the complexity of algorithm for the bivariate case is $\mathcal{O}(n^3)$ where n is the degree of the spline.

To a large extent the efficiency of the algorithm is the result of our look-up table. Grandine's conclusion, that searching for previously computed results is more expensive than simply re-computing the required value, does not hold, since the graph we construct does not depend on the point in which we evaluate the spline. Hence, we only have to look-up simplex splines during the construction of the evaluation graph and can then evaluate the spline in any point without ever searching the table again.

The selection schemes to obtain efficient evaluation graphs are surprisingly simple. For simplex splines we only need to fix the order of the knots and repeatedly select the first three knots in the knot-set. For triangular B-spline surfaces we preferably select elements from the fingerprint and complete the split set with the first knots remaining from the knot-set using a fixed order.

The properties proved in Lemma 10 and Theorem 11 are not straightforward. If the ordering of the knots would not be fixed, but for instance, would depend on the simplex spline under consideration, the algorithm would not work as it turned out during our experiments. Also, if we use a slightly different definition for fingerprints, e.g., $F^I_\beta = \{v_{i_j,\beta_j} \mid 0 \leqslant j \leqslant 2\}$, the required properties do no longer hold.

Since simplex splines are uniquely defined by their knot-sets, enumerating simplex splines is a non-trivial matter. In contrast to (Pfeifle and Seidel, 1994), we avoid explicitly enumerating every simplex spline. The enumeration scheme used by Pfeifle and Seidel could never have been sufficient, since the number of possible names in their enumeration is less than the number of simplex splines that occur during the evaluation of splines of higher degree. Because of this our algorithm scales up to arbitrary degrees and arbitrary domains, and their algorithm does not.

For B-spline surfaces of degree 2 our algorithm yields a slightly different, but equally efficient graph as Pfeifle and Seidel (see Fig. 4). In (Pfeifle and Seidel, 1994) 78 pairs of barycentric coordinates are computed for each triangle. This corresponds to 156 barycentric determinants, provided that the 3rd coordinate is computed as 1 minus the other two coordinates. However, by using our look-up table for barycentric determinants we avoid multiple evaluation. Therefore the number of determinants actually computed by our algorithm is the number of pairs of knots that can be chosen from the $3n + 3$ knots of a B-spline surface, i.e., $\binom{3n+3}{2}$. For $n = 2$ this yields only 36 determinants.

References

Bézier, P. (1972), Numerical Control, Mathematics and Applications, Series in Computing, Wiley.

Dahmen, W., Micchelli, C.A. and Seidel, H.-P. (1992), Blossoming begets B-splines built better by B-patches, Math. Comput. 59 (199), 97–115.

Fong, P. and Seidel, H.-P. (1993), An implementation of triangular b-spline surfaces over arbitrary triangulations, Comput. Aided Geom. Design 10, 267–275.

Franssen, M. (1995), Evaluation of DMS-splines, Master's Thesis, Eindhoven University of Technology.

Grandine, T.A. (1987), The computational cost of simplex spline evaluation, SIAM J. Numer. Anal. 24 (4), 887–890.

Mehlhorn, H. and Näher, S. (1999), LEDA, A Platform for Combinatorial and Geometric Computing, Cambridge University Press.

Pfeifle, R. and Seidel, H.-P. (1995), Spherical triangular B-splines with application to data fitting, in: Post, F. and Göbel, M., eds., Computer Graphics Forum, Vol. 14, Maastricht, the Netherlands, August 28–September 1 1995, Blackwell Publishers, C89–C96.

Pfeifle, R. and Seidel, H.-P. (1994), Faster evaluation of quadratic bivariate DMS spline surfaces, Graphics Interface, 182–189.

Seidel, H.-P. (1991), Symmetric recursive algorithms for surfaces: B-patches and the de boor algorithm for polynomials over triangles, Constructive Approx. 7, 257–279.

Traas, C.R. (1990), Computation of Curves and Surfaces, Chapter: Practice of Bivariate Quadratic Simplicial Splines, Kluwer Academic Publishers, Dordrecht, 383–422.