ELSEVIER

Computer Aided Geometric Design 14 (1997) 13–30

COMPUTER AIDED GEOMETRIC DESIGN

Curve interpolation in recursively generated B-spline surfaces over arbitrary topology

A. Nasri [1]

American University of Beirut, Department of Mathematics, P.O. Box 11-236, Beirut, Lebanon

Received May 1995; revised April 1996

Abstract

Recursive subdivision is receiving a great deal of attention in the definition of B-spline surfaces over arbitrary topology. The technique has recently been extended to generate interpolating surfaces with given normal vectors at the interpolated vertices. This paper describes an algorithm to generate recursive subdivision surfaces that interpolate B-spline curves. The control polygon of each curve is defined by a path of vertices of the polyhedral network describing the surface. The method consists of applying a one-step subdivision of the initial network and modifying the topology in the neighborhood of the vertices generated from the control polygons. Subsequent subdivisions of the modified network generate sequences of polygons each of which converges to a curve interpolated by the limit surface. In the case of regular networks, the method can be reduced to a knot insertion process.

Keywords: Recursive subdivision; Curve interpolation; B-spline; Arbitrary topology

1. Introduction

Recursive subdivision provides definition of surfaces over irregular networks. Research is still ongoing to enhance the capability of such surfaces making them important tools in Computer Aided Geometric Design for modeling complex surfaces. Recent results, for instance, show that recursively generated surfaces, often called recursive subdivision surfaces will have an essential role in establishing a theoretical basis for applying a multiresolution analysis of surfaces of arbitrary topological nature (DeRose et al., 1993). The two well known methods appeared in (Catmull and Clark, 1978; Doo and Sabin, 1978) based on an idea introduced by Chaikin (1974) to generate a curve from a polygon

[1] E-mail: anasri@aub.edu.lb.

SSDI 0167-8396(96)00018-0

by successively cutting its corners. A surface is described by a polyhedral network P_0 which is refined by cutting its corners and its edges by a plane. This will result in a new polyhedron P_1 with new vertices and faces which can be refined in a similar manner to generate another polyhedron P_2. By infinitely repeating this process, a surface S can be generated at the limit as:

$$S = \lim_{i \to \infty} P_i. \tag{1}$$

DeRose et al. (1993) have nicely described the subdivision process in two main steps: *splitting* and *averaging*. In the former, an intermediate polygon $\widehat{P}_i$ is generated by inserting additional temporary vertices in the polyhedron P_i to be subdivided. To generate the vertices of P_{i+1}, the vertices of $\widehat{P}_i$ are then subject to an averaging process using *masks*. Basically a mask m of weight m_i, where $m = (m_i)_{-n \leqslant i \leqslant n}$, is applied to a set of vertices $(\widehat{v}_i)$ of $\widehat{P}_i$ to generate a new vertex v_i of P_{i+1}. v_i is given by:

$$v_i = \frac{\sum_{j=-n}^{n} m_j . \widehat{v}_i}{\sum_{j=-n}^{n} m_j}. \tag{2}$$

In Chaikin's method, the temporary vertices are the midpoints of the edges of P_i. The mask $\{0, 1, 1\}$ is then applied to a sequence of 3 vertices $\{\widehat{v}_{i-1}, \widehat{v}_i, \widehat{v}_{i+1}\}$. This process will not only eliminate all midpoints inserted in the split process but also excludes the corners or vertices of the original polygon.

Doo and Sabin extended Chaikin's algorithm to generate quadratic surfaces. In their method, for every vertex w_i of the polyhedron P_i, a new vertex is generated on each face adjacent to w_i. The new polyhedron P_{i+1} is then obtained by connecting these generated vertices giving an F-*face* for each face f_i of P_i, an E-*face* for each edge e_i of P_i and a V-*face* for each vertex v_i of P_i (see Fig. 1).

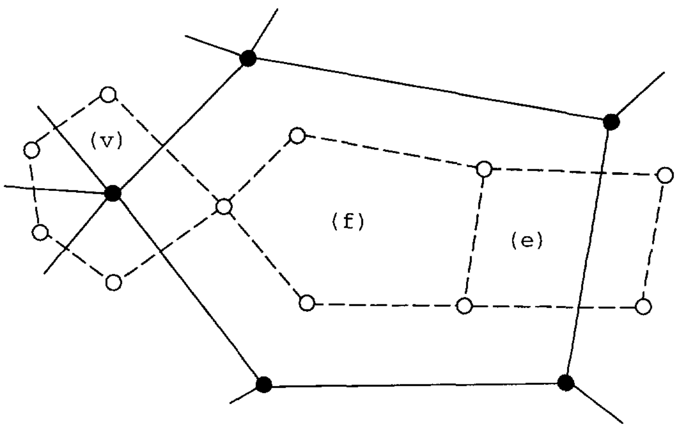


Fig. 1. Doo–Sabin's subdivision. A polyhedron P_i (solid vertices) and the generated one P_{i+1} (hollow vertices). The latter is made of three types of faces: a V-face (v), an E-face (e) and an F-face (f).

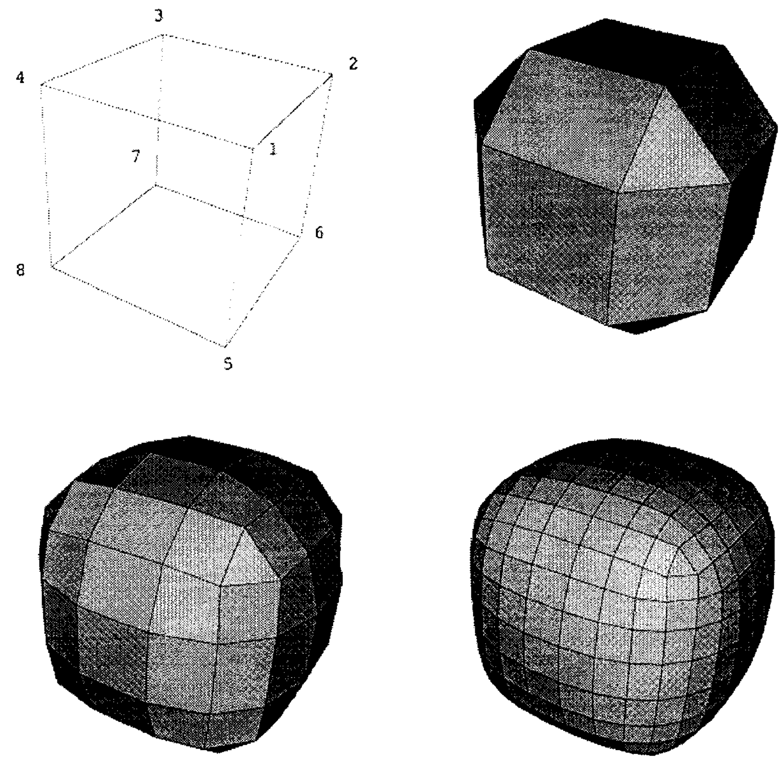


Fig. 2. An example of Doo–Sabin's algorithm surface. A cube network (top left), its first subdivision (top right), its second subdivision (bottom left) and its third subdivision (bottom right).

In light of the above, the new vertices of P_{i+1} are linear combinations of the vertices of P_i. Therefore the key idea in these methods lies in the rules by which the new vertices are generated. In the Doo–Sabin technique, for instance, the following rules apply to generate the new vertices v_i, called *images*, on a face F having the vertices $(w_i)_{1 \leqslant i \leqslant n}$:

$$v_i = \sum_{j=1}^{n} \alpha_{ij} w_j, \tag{3}$$

where the α_{ij}'s are given by:

$$\alpha_{ii} = \frac{n+5}{4n}, \tag{4}$$

$$\alpha_{ij} = \frac{3 + 2\cos(2\pi(i-j)/n)}{4n}. \tag{5}$$

Fig. 2 shows the results of applying these rules to a cube network. The behavior of the limit surfaces has been analyzed (Doo and Sabin, 1978; Ball and Storry, 1984). The technique was also extended to generate interpolating surfaces over irregular networks (Nasri, 1991; Halstead et al., 1993). Furthermore, equations for points on the limit surface and normals to the surface were also established. It was also shown that the surface generated is a B-spline surface which is tangent plane continuous everywhere. Peters (1993), and Loop and DeRose (1994) have extended the technique to generate smooth surfaces over arbitrary topology by separating singular regions after a few steps of subdivision; these regions were filled in by closed form patches.

In this paper we describe a method to force a recursively generated surface over arbitrary topology to interpolate predefined quadratic B-spline curves. Curve interpolation is an essential tool in CAGD. For instance, in car design a surface can be moved to interpolate predefined curves such as feature lines. In our method, the control polygon of each interpolated curve, called CP, is a sequence of edges and vertices of the polyhedral network describing a surface. The method consists of modifying the topology in the neighborhood of these vertices generated from CP. This is to ensure that subsequent subdivisions of the network will subdivide CP according to Chaikin's method. At the limit the subdivided polygon converge to a curve on the limit surface. Section 2 discusses the conditions to achieve this goal using the similarity between the two processes of knot insertion and subdivision, in the case of regular networks. Section 3 describes the curve interpolation method by considering first a control polygon whose vertices make a row or a column path of a regular network. After that the method is generalized to a control polygon defined by an arbitrary sequence of edges and vertices of the network. Section 4 discusses implementation issues and Section 5 draws conclusions and outlines future work.

2. Knot insertion and recursive subdivision

In this section, we discuss the similarity between Chaikin's subdivision and repeated knot insertion. This helps us to establish the conditions under which a subdivided network generates a Chaikin's polygon from a given one.

2.1. Polygonal subdivision conditions

A piecewise polynomial B-spline curve is defined by:

$$S^n = \sum_{i=0}^{L+n-1} \mathbf{d}_i N_i^n(u), \tag{6}$$

where the vector valued coefficients $(\mathbf{d}_i)$ form the *de Boor* control polygon and $N_i^n(u)$ are piecewise polynomials of degree n which form a basis for the linear space of the piecewise polynomial of degree n. The $N_i^n(u)$ are defined over a knot sequence: $\cdots < u_0 < u_1 < u_2 < \cdots$. This knot sequence forms a partition of the real axis whereas the $N_i^n(u)$ form a partition of unity.

The de Boor polygon can be subdivided by inserting a knot in each domain intervals of the parameter u. The process results in a refined polygon and a refined knot sequence, however the curve remains unchanged. Riesenfeld (1975) has shown that by repeated subdivision, the refined polygon will eventually converge to the underlying curve.

In the quadratic case, inserting a knot $\widehat{u}$ in the middle of an interval $[u_i\ u_{i+1}]$ results in the two de Boor points P_i^1 and P_{i+1}^1 replacing the old point P_i of the original polygon. These are given by the formula:

$$P_k^1 = (1 - \alpha_k)P_{k-1} + \alpha_k P_k, \tag{7}$$

where

$$\alpha_k = \frac{\widehat{u} - u_k}{u_{k+n} - u_k}. \tag{8}$$

Chaikin's method can be thought of as a repeated process of knot insertion in quadratic B-spline curves defined over uniform knot sequence. The refined points P_i^1 are then given by:

$$P_{2i-1}^1 = \tfrac{3}{4}P_i + \tfrac{1}{4}P_{i-1}, \tag{9}$$

$$P_{2i+1}^1 = \tfrac{3}{4}P_i + \tfrac{1}{4}P_{i+1}, \tag{10}$$

hence each leg, *i.e., edge*, of the de Boor polygon is divided into the ratio 1 : 3 and 3 : 1 which corresponds to the mask $\{0, 1, 1\}$ mentioned in Section 1.

In light of the above, we can establish the following subdivision condition for a control polygon (P_i^0) to converge to its corresponding quadratic B-spline curve:

Condition (A)

The control vertices of each subdivided polygon (CP_i) must be obtained from (CP_{i-1}) by applying the mask {0,1,1} to every sequence of three vertices; hence equation (9) and (10) must be satisfied

It should be noted that the midpoint of each leg will be a point on the curve.

2.2. *Polyhedral subdivision conditions*

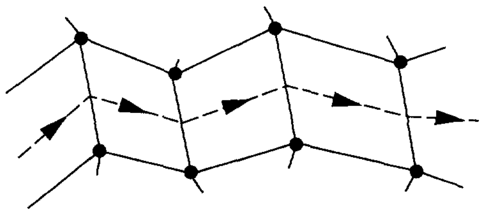
In the case of a regular mesh or network, where all vertices are 4-valent, the same process of repeated subdivision can be applied to surfaces. A piecewise polynomial tensor product B-spline $S^{n,m}$ is defined by:

$$S^{n,m} = \sum_i \sum_j \mathbf{d}_{ik} N_k^m(v) N_i^n(u). \tag{11}$$

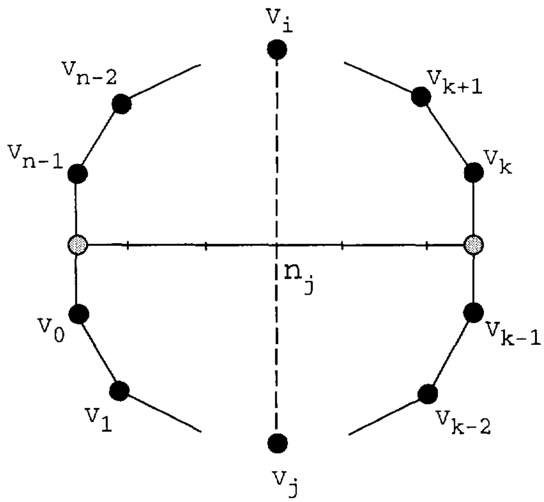
The $\mathbf{d}_{ik}$ are the de Boor points which form a rectangular 4-*sided* control mesh of a surface. The N_k^m and N_i^n are the same B-spline basis functions given in Section 2.1. These are defined over a rectangular grid of knot sequence forming the u, v domain. Similar to the curve case, repeated knot insertion can be used to subdivide the control mesh of the surface producing a refined mesh which will eventually converge to the surface.

Chaikin's algorithm has been extended to surfaces by inserting knots in the middle of each u- and v-interval leading to the same rules (3)–(5) to generate a new network.

It is easy to show that by inserting a knot $\widehat{u}$ at the middle of the interval $[u_i\ u_{i+1}]$ a row of faces (f_i), called a *strip*, is added to the network as will be discussed in the



(a)



(b)

Fig. 3. Subdivision conditions: (a) Legs of a control polygon made of midface edges of 4-sided faces. (b) A leg as a midface edge of an n-sided face ($n \neq 4$).

following section. These faces define the control polygon of a curve that corresponds to the knot-line $u = \widehat{u}$. Each leg of this polygon is a *midface* edge of a face, where a midface edge of f_i joins the midpoints of two shared edges of f_i with other faces of the strip as depicted in Fig. 3(a). Two edges are nonadjacent if they do not share a common vertex. Note that inserting a knot in a v-interval can be done in an analogous manner.

In the case of a regular network, a condition can now be established for a control polygon to converge to a curve on the limit surface. For example, in the Doo–Sabin's subdivision, this condition is as follows: *each leg of the polygon should be a midface edge of a* 4*-sided face.* This condition guarantees that in subsequent subdivisions, the legs of the subdivided polygon will also be midface edges of the generated F-, E-, or V-faces. The condition can be generalized to handle a leg which is a midface edge of an n-sided face having an even number ($n = 2k$) of vertices $(v_i)_{0 \leqslant i \leqslant n-1}$ (see Fig. 3(b)). Let m_0 and m_{k-1} be the midpoints of the edges $v_0 v_{n-1}$ and $v_{k-1} v_k$, respectively. The corresponding midface edge will be the leg $m_0 m_{k-1}$ of the control polygon if the following condition holds:

Condition (B)

The midpoint n_j of every couple of vertices v_i and v_j such that $i + j = n - 1$, called *opposite* vertices, is the Chebyshev point defined on the interval $[m_0 m_{k-1}]$ as follows:

$$n_j = \frac{(1+\beta_j)m_0 + (1-\beta_j)m_{k-1}}{2}, \tag{12}$$

where $j = 0, \ldots, k-1$ and the β_j are given by

$$\beta_j = \frac{\cos((2j+1)\pi/2k)}{\cos(\pi/2k)}. \tag{13}$$

Note that a similar condition was used to control the boundary curves of recursive subdivision surfaces in (Nasri, 1987). It was also shown that subdivision of a face f_i having a midface edge e_i and satisfying this condition will generate an F-face f_s having a midface edge e_s, where e_s can be obtained from e_i by condition (A). Furthermore, the F-face f_s satisfies the condition (B).

3. Curve interpolation method

Consider a polyhedral network defined by the triplet $P = (V, E, F)$, where

$$V = (v_i)_{1 \leqslant i \leqslant m}, \tag{14}$$

$$E = (e_i)_{1 \leqslant i \leqslant n}, \tag{15}$$

$$F = (f_i)_{1 \leqslant i \leqslant r}. \tag{16}$$

An edge e_i has two vertices denoted by v_i^0 and v_i^1 and two common faces f_i^0 and f_i^1.

Given: a control polygon on this network which is defined by

$$\mathrm{CP} = (E_p, V_p), \tag{17}$$

where E_p and V_p are subsets of E and V, respectively. The method described here can handle closed curves only. Other types of curves such as open and boundary curves are the subject of a subsequent paper. For a closed curve, the control polygon, CP must define a closed path from a vertex v_1 to v_k, where $k = \mathrm{Card}(V_p)$, as an alternating sequence of interior vertices and edges as follows: $v_1, e_1, v_2, e_2, \ldots, e_{k-1}, v_k, e_k, v_1$. Such a path can be simply denoted by $[v_1, v_2, \ldots, v_k, v_1]$. Accordingly an edge e_i of E_p is common to vertex v_i and $v_{(i+1) \bmod k}$. Furthermore, no vertex v_i can appear twice in the path of a polygon and a vertex must not be used in two different control polygons.

Problem: how to generate a surface from the polyhedron P that interpolates the quadratic B-spline curve having CP as a control polygon?

The solution can be devised by considering first the case of regular networks as follows.

3.1. The regular case: row or column

Consider the case of a regular network where the set V_p forms a row or a column of the network. The problem can be easily solved by knot insertion.

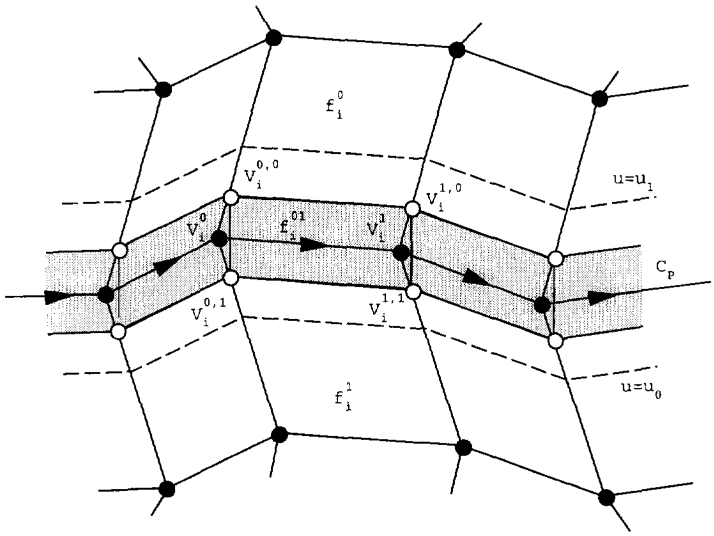


Fig. 4. Curve interpolation in the regular case where a polygon forms a row of vertices. Shaded faces are added by knot insertion in the middle of the corresponding u-interval.

Assume that the two arrays of faces common to the edges of E_p are given by (f_i^0) and (f_i^1) which define two knot-lines, called $u = \widehat{u}_0$ and $u = \widehat{u}_1$, respectively (see Fig. 4). The idea is to insert a knot $\widehat{u}$ in the middle of the interval $[\widehat{u}_0, \widehat{u}_1]$ which defines a new knot-line that will eventually correspond to the B-spline curve of CP. The knot insertion process will generate a 4-sided face f_i^{01} for each edge e_i in E_p. A strip of faces is therefore added to the network. These faces should be modified in order to have CP as the control polygon of their corresponding knot-line $u = \widehat{u}$. Each face f_i^{01} of this strip is made up of the four vertices $v_i^{0,0}$, $v_i^{0,1}$, $v_i^{1,1}$ and $v_i^{1,0}$ as depicted in Fig. 4. Assume that I_0 and I_1 are the midpoints of the new edges $v_i^{0,0}v_i^{0,1}$ and $v_i^{1,0}v_i^{1,1}$, respectively. By shifting these edges by a vector $I_1v_i^0$ and $I_2v_i^1$, respectively, their midpoints become v_i^0 and v_i^1. Each edge e_i will be a midface edge of f_i^{01}, hence a leg of the polygon which converges to a curve on the surface associated with the knot-line $u = \widehat{u}$.

Note that similar modification can be applied when V_p forms a column instead of a row.

3.2. The general case

For irregular meshes, the situation is more complicated as the notion of a row or a column is not valid. Furthermore, even in the regular case, the definition of control polygons should not be restricted to rows or columns. In order to release this restriction, a network is thought of as a polyhedron and a control polygon CP is defined according to Eqs. (14)–(16).

Initially, the method consists of applying a one-step division of the polyhedron P_0 resulting in a polyhedron P_1 with a set of faces associated with CP; namely the E-faces and V-faces generated from its edges and vertices respectively. These faces will play a similar role to that of the faces generated by knot insertion in the regular case. For instance, the E-faces are subject to a linear transformation to cut the legs of $CP_0 = CP$ in the ratio proposed in condition (A). This modification, however, may create additional vertices which will disturb the topology of the network. Furthermore, the V-faces must be modified to satisfy condition (B). The following algorithm provides an outline of the proposed solution:

```
    For each edge e_i in CP Do
        Begin
        Fe := E-face(e_i);
        Modify Fe by shifting its shared edges as suggested in Section 3.2.1;
                {its midface edge is then obtained from e_i by condition (A)}
        Modify the adjacent faces to Fe accordingly
        End;
For each vertex v_i in CP Do
        Begin
        Fv := V-face(v_i);
        Modify Fv by moving its vertices as suggested in Section 3.2.2;
                {its midface edge is then obtained by condition (B)}
        Modify the adjacent faces to Fv accordingly
        End;
```

Note that the adjacent F-faces to the E- and V-faces are not involved directly in the modification process, but are adjusted according to the outcome of the modification process. The following sections discuss this modification.

3.2.1. Modification of E-faces

Consider Fig. 5 which shows an edge e_i and its E-face F. As all E-faces are 4-sided, F is made of v_i^{00}, v_i^{01}, v_i^{11} and v_i^{10} which are the images of v_i^0 and v_i^1 on the faces common to e_i. The segments $v_i^{00}v_i^{01}$ and $v_i^{10}v_i^{11}$ must be shifted so that their midpoints I_1 and I_2 will coincide with the points J_1 and J_2 respectively. J_1 and J_2 are obtained by applying the masks given in condition (A) as follows:

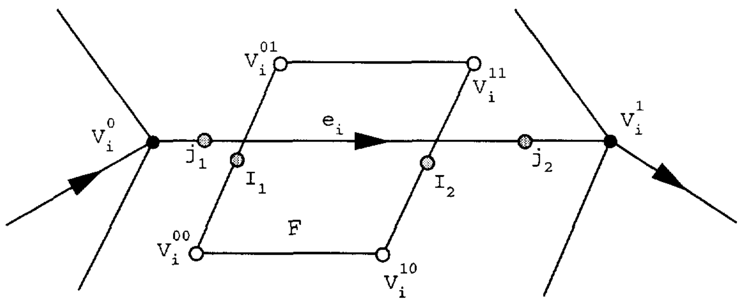


Fig. 5. Modification of an E-face (hollow vertices). J_1J_2 is a leg of the subdivided polygon.

$$J_1 = \frac{3v_i^0 + v_i^1}{4}, \tag{18}$$

$$J_2 = \frac{3v_i^1 + v_i^0}{4}. \tag{19}$$

Thus the two edges $v_i^{00}v_i^{01}$ and $v_i^{10}v_i^{11}$ are shifted by $T(\overrightarrow{I_1J_1})$ and $T(\overrightarrow{I_2J_2})$, respectively. $T(\vec{V})$ is a translation in the direction of the vector $\vec{V}$. As a result, the segment J_1J_2 will be a leg of the polygon CP_1 and a midface edge of F.

3.2.2. Modification of V-faces

For each edge e_i of CP_0 the subdivision process creates a 4-sided E-face which determines a leg of CP_1. The remaining problem is embodied in the generation of the legs of the CP_1 which correspond to the V-faces. This can be achieved by modifying the latter according to condition (B).

To begin, let us discuss how a 3-sided face can be modified before generalizing to n-sided faces.

Modification of 3-sided faces. Consider the case of a 3-valent vertex of the control polygon which is depicted in Fig. 6(a) where the control polygon has two edges e_{i-1} and e_i incident to v_i and lying on the same face. The two generated E-faces from e_{i-1} and e_i will have a common vertex v_i^j. However after carrying out the two necessary transformations of these E-faces, the vertex v_i^j will generate two vertices $v_i^{j,1}$ and $v_i^{j,2}$— one from each transformation. Consequently the 3-sided face will become a 4-sided one. This topological modification must be reflected in the neighborhood of v_i^j. Thus the F-face and V-face meeting at this vertex must be modified resulting in a new vertex and a new edge added to both of them. Assuming that M_1 and M_2 define the matrices of the necessary translations applied to the two E-faces involved, respectively, the following algorithm summarizes this modification:

```
For each vertex v_i of CP Do
    IF v_i is 3-valent then
        Begin
        Ff := F-face adjacent to v_i;
        Fv := V-face adjacent to v_i;
        Fe_1 := First E-face adjacent to v_i;
        Fe_2 := Second E-face adjacent to v_i;
            { M_1 is the matrix of the translation on Fe_1 }
        v_i^{j,1} := ShiftVertex (v_i, M_1);
            {ShiftVertex applies the transformation defined by M_1 to v_i }
        v_i^{j,2} := ShiftVertex (v_i, M_2);
        Assign to v_i, which is common to Fe_1, Fv and Ff, the coordinates of v_i^{j,1};
        Assign to v_i of Fe_2 the coordinates of v_i^{j,2};
        Add vertex v_i^{j,2} to Ff and Fv;
        Add edge v_i^{j,1} v_i^{j,2} to Ff and Fv
        End
```

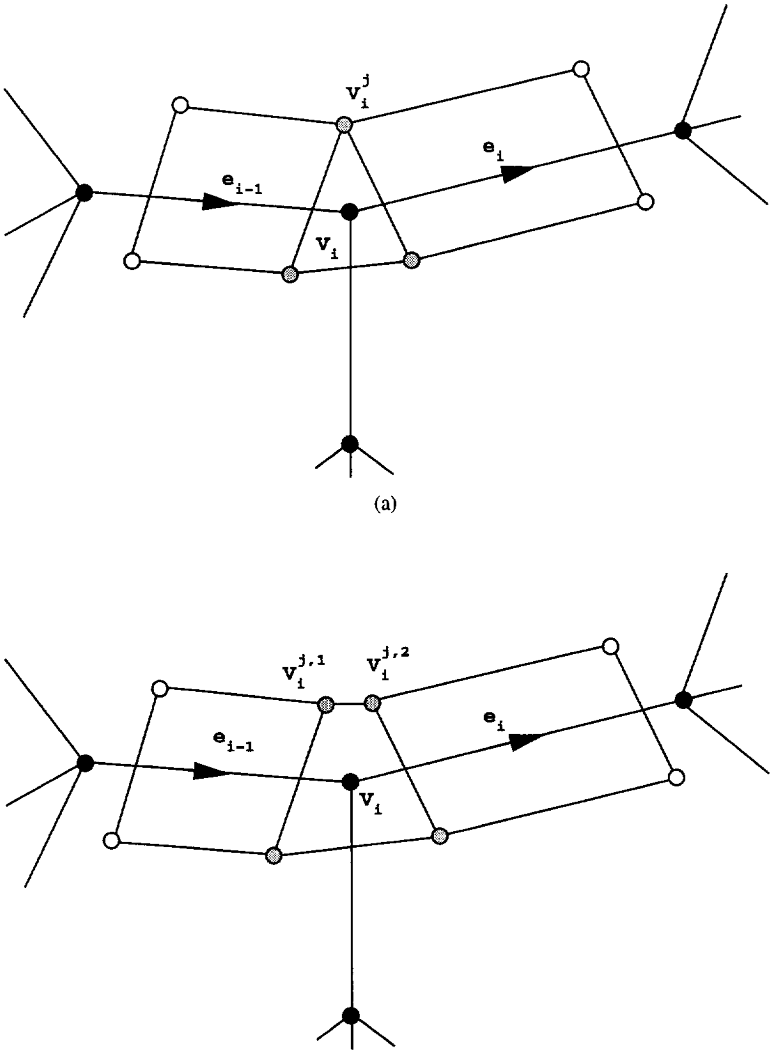


Fig. 6. Modification of a 3-sided face. (a) A 3-sided face (shaded vertices) generated after one-step of subdivision. (b) The corresponding 4-sided face after modification. v_i^j generates two vertices $v_i^{j,1}$ and $v_i^{j,2}$.

Fig. 6(b) shows the results of modifying the V-face of Fig. 6(a).

Modification of n-sided faces ($n \geqslant 4$). Consider the configuration of Fig. 7 where the following sequence:

$$v_{i-1},\ e_{i-1},\ v_i,\ e_i,\ v_{i+1} \tag{20}$$

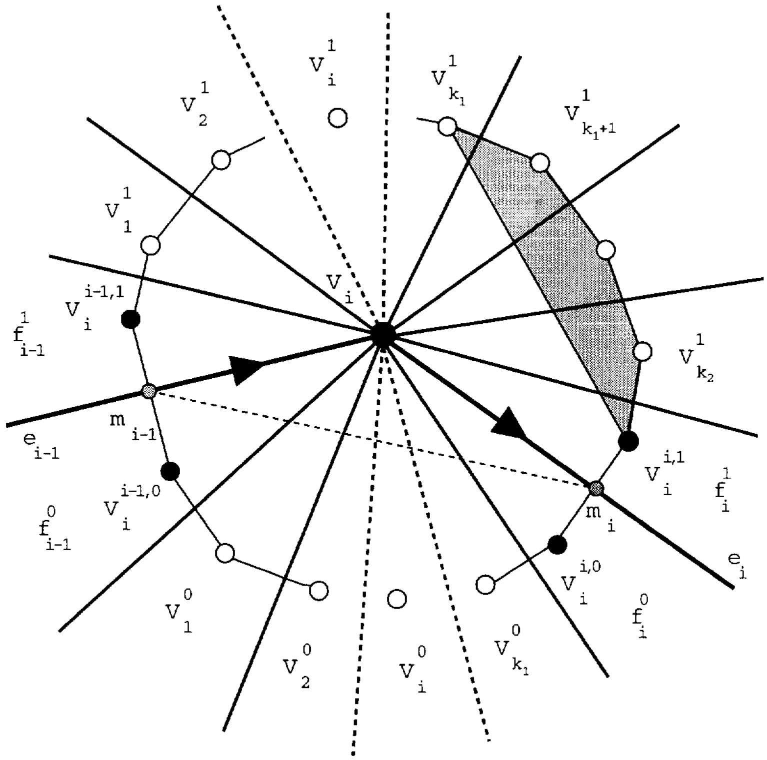


Fig. 7. Modification of an n-sided face. The midface edge $m_{i-1}m_i$ is a leg of the subdivided polygon.

is part of the control polygon CP and v_i is an n-valent vertex. Assume that $v_i^{i-1,0}$ and $v_i^{i-1,1}$ are the images of v_i on the faces f_{i-1}^0 and f_{i-1}^1 common to edge e_{i-1}, respectively. Similarly let $v_i^{i,0}$, $v_i^{i,1}$ be the images of v_i on the faces f_i^0 and f_i^1 common to edge e_i, respectively. Furthermore assume that these vertices have already been shifted in the E-face modification process as suggested in Section 3.2.1. Because of this modification the edge $m_{i-1}m_i$ will be a leg of the new polygon. The points m_{i-1} and m_i are the midpoints of the edges $v_i^{i-1,0}v_i^{i-1,1}$ and $v_i^{i,0}v_i^{i,1}$, respectively. Therefore, the problem reduces to making that leg a midface edge of a V-face of P_1. The method consists of adjusting the topology of the V-face VF_i of v_i in order to satisfy condition (B). This is accomplished in two steps: *spatial* modification and *split* modification. First, the spatial location of some of the vertices of VF_i may need to be altered; second the face VF_i itself may have to be split into two faces. Basically, the idea consists of comparing the number of vertices of VF_i on either side of $m_{i-1}m_i$ (call them k_1 and k_2), excluding $v_i^{i-1,0}$, $v_i^{i-1,1}$, $v_i^{i,0}$ and $v_i^{i,1}$. Without loss of generality, assume that $k_1 \leqslant k_2$. The following algorithm summarizes the remaining steps:

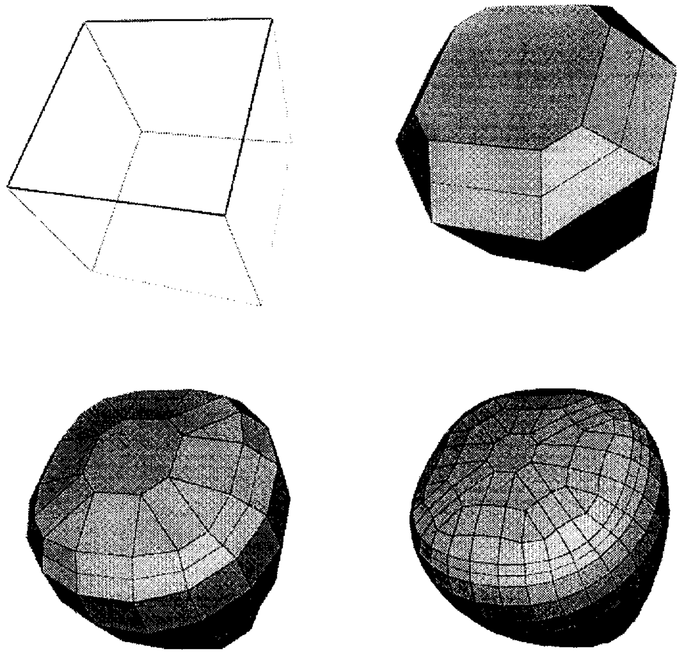


Fig. 8(a)

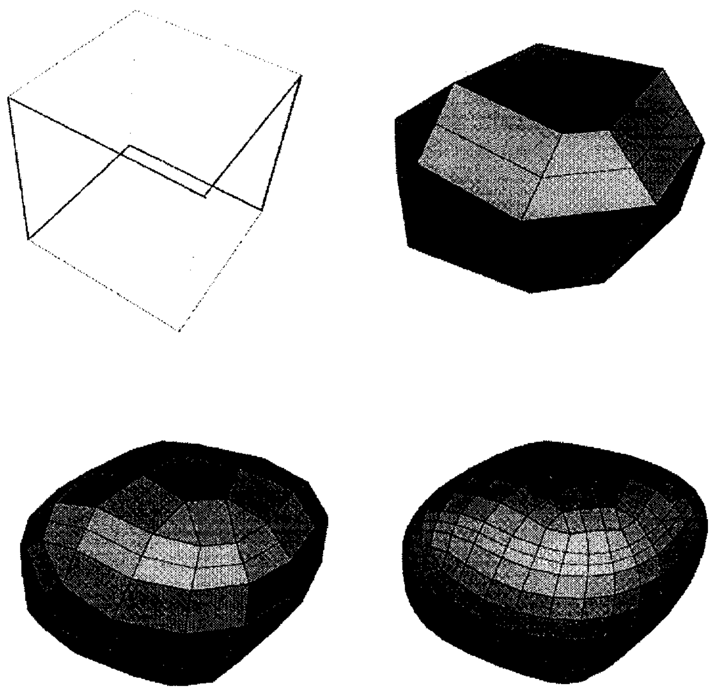
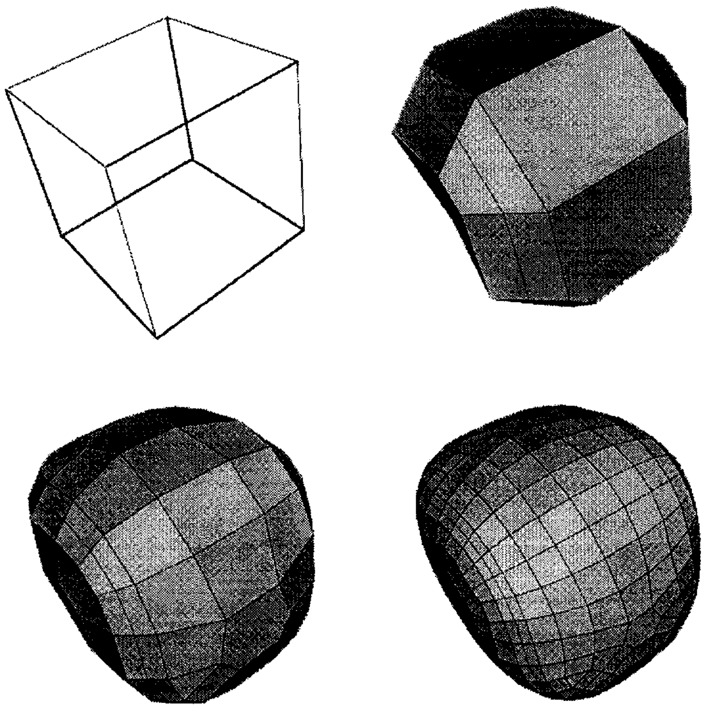


Fig. 8(b)



(c)

Fig. 8. The cube surface of Fig. 2 interpolating various B-spline curves. (a) A polygon path of the top vertices [1,2,3,4,1], (b) A polygon path of vertices [2,1,5,8,7,3,2] and (c) A polygon path of vertices [4,1,2,6,7,3,4]. In all figures, the original network (top left), its modified first subdivision (top right), first subdivision (bottom left) and second subdivision (bottom right) of the modified network are shown.

1. Label the vertices of the face VF_i so that the two paths from m_{i-1} to m_i are the following:

$$[m_{i-1}, v_i^{i-1,1}, v_1^1, v_2^1, \ldots, v_{k_2}^1, v_i^{i,1}, m_i], \tag{21}$$

$$[m_{i-1}, v_i^{i-1,0}, v_1^0, v_2^0, \ldots, v_{k_1}^0, v_i^{i,0}, m_i]. \tag{22}$$

2. On $m_{i-1}m_i$ construct k_1 Chebyshev points n_j as given in Eqs. (12)–(13) after replacing k by k_1.
3. For $r := 1$ to k_1 Do adjust vertex v_r^1 so that n_r is the midpoint of the segment $v_r^0\, v_r^1$.
4. Modify VF_i so that it is made of the following vertices which make a path:

$$[v_i^{i-1,1}, v_1^1, v_2^1, \ldots, v_{k_1}^1, v_i^{i,1}, v_i^{i,0}, v_{k_1}^0, v_{k_1-1}^0, \ldots, v_1^0, v_i^{i-1,0}]. \tag{23}$$

5. Construct an additional face whose vertices make the following path:

$$[v_{k_1}^1, v_{k_1+1}^1, v_{k_1+2}^1, \ldots, v_{k_2}^1, v_i^{1,1}]. \tag{24}$$

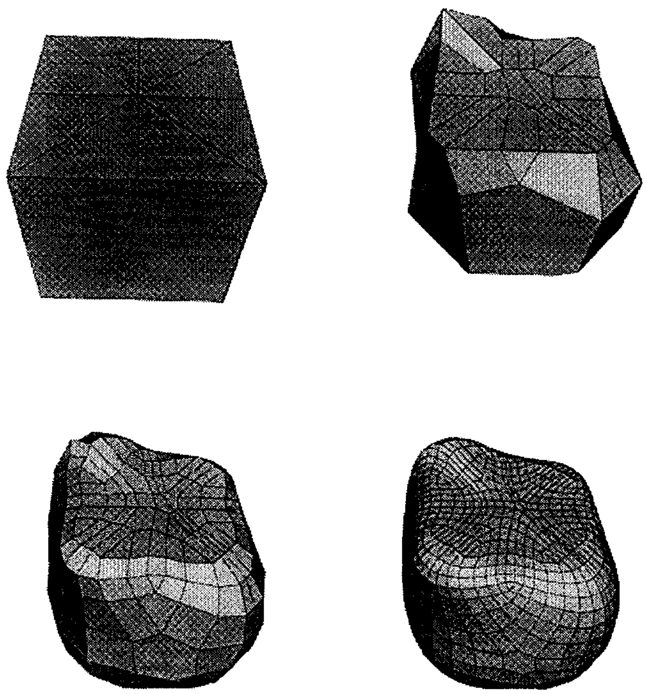


Fig. 9. Curve interpolation illustrating the case of $k_1 > 0$. A cube network with a polygon path through an 8-valent vertex (solid lines) (top left), its modified first subdivision (top right), first subdivision (bottom left) and second subdivision (bottom right) of the modified network are shown. The 8-sided V-face becomes 6-sided and one additional 4-sided face is constructed.

After this modification, the leg $m_{i-1}m_i$ is a midface edge of VF_i which satisfies condition (B); thus a proper subsequent subdivision of this leg is guaranteed. It should be mentioned however that in case of a control polygon where the two edges e_{i-1} and e_i belong to the same face, we have $k_1 = -1$ and hence VF_i will be a 4-sided face. Clearly, for a 3-sided valent vertex, we have also k_1 or $k_2 = -1$.

4. Implementation

The above results were implemented where Figs. 8 and 9 were produced. In Fig. 8, the cube of Fig. 2 was considered with various control polygons. Fig. 8(a), for instance, illustrates the interpolation of the curve that corresponds to the control polygon made up of the top 4 vertices. Notice that each 3-sided face becomes a 4-sided face after applying the necessary modification. The first and second subdivisions of the modified network are also provided. On the same cube, more complicated polygons are considered in Figs. 8(b) and 8(c) where the corresponding first modification and subsequent subdivisions are depicted. The case of a control polygon having an 8-valent vertex on its path and illustrating the case of $k_1 > 0$ is considered in Fig. 9. The 8-sided V-face is

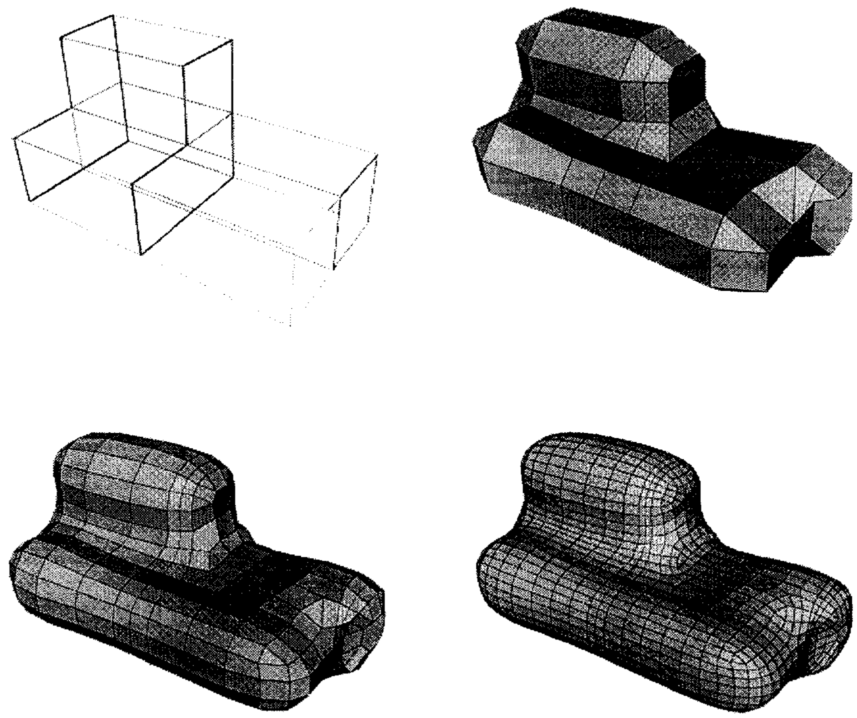


Fig. 10(a). Interpolation of various curves: a network with three control polygons (solid lines) (top left), its modified first subdivision (top right), first subdivision (bottom left) and second subdivision (bottom right) of the modified network are shown.

split into two faces: a 6-sided face which satisfies condition (B) and an additional 4-sided face. Finally, a network with three control polygons is given in Fig. 10(a). The 5-valent vertex on the path of the middle control polygon illustrates the case of $k_1 = 0$ where the additional face is 3-sided. Two successive subdivisions of the modified network are also shown. Different views of the resulting surface interpolating the corresponding curves are shown in Fig. 10(b).

5. Conclusions

In this paper, a method that extends the capability of the recursive subdivision technique to generate surfaces that interpolate predefined curves is described. The technique consists of a one-step division of the initial network and a topological modification of the E-faces and V-faces generated from the edges and vertices of the given control polygon. It should be noted that there are different types of transformation that can be used to adjust these

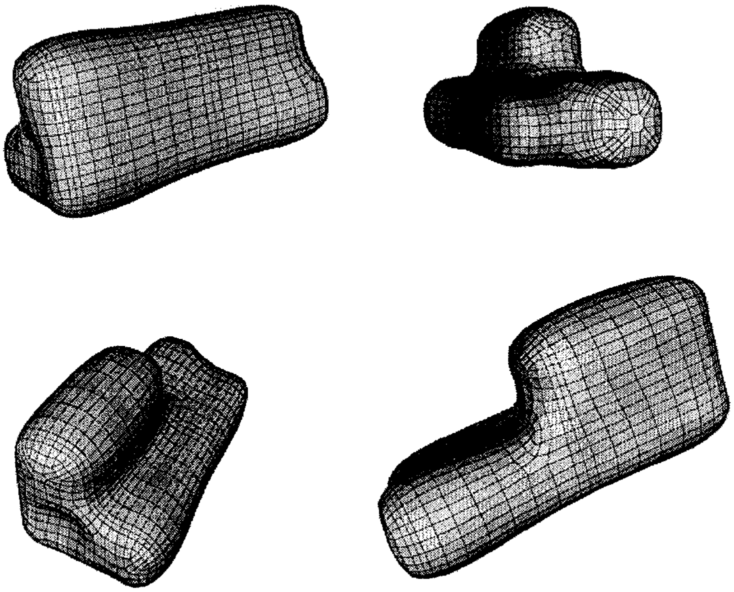


Fig. 10(b). Different views of the resulting surface interpolating the three curves: a bottom view (top left), a front view (top right), a side view (bottom left) and a back view (bottom right).

faces to satisfy the Conditions established in this paper. These degrees of freedom can be used to optimize the shape of the resulting surface and a solution is currently under implementation.

Curve interpolation will make recursively generated surfaces much more attractive in CAGD. As a result, the surface of a car body, for example, can be moved to interpolate specific curves such as feature lines. Furthermore, the generation of surfaces through irregular meshes of curves can also be considered. This however requires the interpolation of open and intersecting meshes of curves, which are the subject of subsequent papers. Open curves present the problem of how and whether to interpolate their endpoints which may lie on the interior or on the boundary curves of the surface. Finally, the extension of the algorithm to higher order curves and surfaces can be inspired.

Acknowledgments

This work was supported by a research grant from the American University of Beirut. The author would like to thank Professor Gerald Farin and Dr. Malcolm Sabin for their helpful comments. Many thanks also go the referees for their fruitful suggestions. Thanks are also due to Mr. R. Jaffar for his help in implementing part of the underlying system.

References

Ball, A. and Storry, D. (1984), Recursively generated B-spline surfaces, in: *Proceedings of CAD 84*, Brighton, England, 112–119.

Catmull, E. and Clark, J. (1978), Recursively generated B-spline surfaces on arbitrary topological meshes, Computer-Aided Design 10(6), 350–355.

Chaikin, G.M. (1974), An algorithm for high speed curve generation, Comput. Graph. Image Process. 3(12), 346–349.

DeRose, T., Lounsbery, M. and Warren, J. (1993), Multiresolution analysis for surfaces of arbitrary topological type, Technical Report 93-10-05, Department of Computer Science and Engineering, University of Washington, Seattle.

Doo, D. and Sabin, M. (1978), Behavior of recursive division surfaces near extraordinary points, Computer-Aided Design 10(6), 356–360.

Halstead, M., Kass, M. and DeRose, T. (1993), Efficient, fair interpolation using Catmull–Clark surfaces, Computer Graphics 27(3), 35–44.

Loop, C.T. and DeRose, T. (1994), Smooth spline surfaces over irregular meshes, Computer Graphics 24(3), 303–310.

Nasri, A. (1987), Polyhedral subdivision methods for free form surfaces, ACM Trans. Graphics 8(1), 89–96.

Nasri, A. (1991), Surface interpolation on irregular network with normal conditions, Computer Aided Geometric Design 8(1), 89–96.

Peters, J. (1993), Smooth free-form surfaces over irregular meshes generalizing quadratic splines, Computer Aided Geometric Design 10(4), 347–361.

Riesenfeld, R.F. (1975), On Chaikin's algorithm, IEEE Comput. Graphics Appl. 4(3), 304–310.