

Taxonomy of interpolation constraints on recursive subdivision curves

Ahmad H. Nasri¹,
Malcolm A. Sabin²

¹ Dept. of Math. & Computer Science, American University of Beirut, Beirut, PO Box 11-236, Lebanon

E-mail: anasri@aub.edu.lb

² Numerical Geometry Ltd, 26 Abbey Lane, Lode, Cambridge CB5 9EP, United Kingdom

E-mail: malcolm@geometry.demon.co.uk

Published online: 14 May 2002

© Springer-Verlag 2002

This paper is the first of two, which together describe and classify the various situations that any complete study of interpolation constraints for a recursive subdivision surface needs to consider. They do so in the form of a systematic taxonomy of situations. Presented here are curve cases, which provide good illustrations of principles which will be used in both contexts; surfaces will be addressed in the second paper. Known results are classified and open questions identified.

Key words: Recursive subdivision – Interpolation constraints – End control – points – target directions – B-spline curves – Bézier

Correspondence to: A.H. Nasri

Part of this work was done during a visit by A. Nasri to Cambridge University

1 Introduction

The scenario that we consider in these papers is that a surface is defined by a collection of interpolation constraints, either of points or of curves, such curves being themselves subdivision curves characterized by control polygons and possibly having interpolation constraints. The interpolation points and the vertices of the curve control polygons are then joined to form the network of the desired topological structure, possibly with additional vertices to help provide some geometric control in regions sparsely provided with hard constraints (Nasri and Sabin 2001). Some preprocessing may then be done on this network to produce a new network which can be interpreted by one of the standard recursive division constructions. Alternatively the interpolations may be taken into the subdivision process dynamically.

The result of this is a surface of the required topological structure which can be controlled qualitatively in some regions and by hard interpolation constraints in others. Although this procedure is not yet complete in all details, specific interpolation results fit into this framework. This paper describes and classifies the known results relating to curves, while its sequel describes and classifies the known surface results, using the same structures.

A recursive division surface, S , is basically defined by a duple, (P_0, R) , where P_0 is an initial configuration and R is a refinement procedure. The configuration consists of a set of vertices, edges, and faces. This is often referred to as a *polyhedron*, even though the faces need not have coplanar vertices. The refinement procedure is a set of rules applied to a configuration to generate another with more vertices, edges and smaller faces than the initial one. At each level, i , of the refinement process, the polyhedron P_{i-1} is taken as input to the refinement R , which produces another polyhedron, P_i , which may itself be taken as input to the next refinement step and so on. If R satisfies some conditions (Doo and Sabin 1978; Reif 1995; Zorin and Schröder 2000), then at the limit the sequence of polyhedra P_i converges to a smooth surface.

Similarly, a subdivision curve is defined by a sequence, P_0 , of vertices and edges (the *polygon*), and a refinement procedure, R .

There are many alternative sets of rules, R . They may typically be generated by using the knot insertion procedure of some Box-spline. This in-

duces a set of rules to be applied in the regular case. These rules take the form of coefficients used to form the coordinates of the vertices of P_i , by taking linear combinations of the vertices of P_{i-1} . These rules are then extended by defining appropriate linear combination coefficients to cover the irregular situations of non-regular vertices or faces. When the rules are produced in this way, the limit surface typically consists of polynomial or bipolynomial patches in regular regions with infinite regresses around the singularities, which do not increase in number as subdivision proceeds.

The best known examples are the Doo–Sabin extension of the quadratic B-spline (Doo and Sabin 1978) and the Catmull–Clark extension of the cubic B-spline (Catmull and Clark 1978). Although in principle higher-order tensor-product B-splines can be extended in this way, the templates involved are larger, and therefore require many more cases of interactions between irregularities to be considered in the extension process.

The three-direction quartic box-spline was partially extended by Loop (1994), and the four-direction quadratic was examined (though not identified) by Doo (1978) and found to be rather unsatisfactory. It was later considered in more detail by Sabin (1986) and by Peters and Reif (1997, 1998).

Recursive division procedures have been applied by Peters (1993) and Loop (1994) within methods which replace the infinite regresses around the irregularities either by n -sided patches or by n -sided combinations of 3- or 4-sided patches. In these situations, one or two steps of the recursive division construction are used to separate the irregularities.

A final important class of constructions contains the interpolatory constructions of Dyn et al. (1987, 1990), Dyn and Levin (1990), Dubuc and Deslauries (1989), and Kobbelt (1996), which are designed to have the property that every vertex lies on the limit surface. We do not explore these in detail because they do not give bounded curvature, and because they apply only to the case where every vertex is to be interpolated. Binary subdivision schemes over irregular topology were also considered by Warren (1995). The analysis of the subdivision surfaces at the extraordinary point was carried out by Doo and Sabin (1978), Ball and Storry (1984), Reif (1995, 1999), Zorin and Schröder (2000), and the C^k case was considered by Prautzsch (1995). Sub-

division schemes were recently modified to satisfy various interpolation constraints such as creases, vertices, normals, and curves by Nasri (1987, 1991, 1997a,c, 1998), Hoppe et al. (1994), Schweitzer (1996), Levin (1999a,b), and Biermann et al. (2000). Non-uniform subdivision schemes also emerged, as in Gregory and Qu (1996), Habib (1996), and Sederberg (1998).

This paper focuses on interpolation constraints in the curve case, both because these are important in their own right as part of the overall procedure, and because ideas can be introduced in this simpler context which will also be found useful in the surface context. It is described for the quadratic and the cubic contexts, but *most of the principles applied are equally applicable to other schemes*. The paper is divided as follows: Section 2 introduces some notation that classifies various types of points used in the interpolation and subdivision processes. In Sect. 3, five basic methods are established to handle a number of interpolation constraints. The remaining sections discuss how to possibly achieve each of these interpolation constraints using one or more of the five proposed methods. Namely, the closed case is discussed in Sect. 4, the open case in Sect. 5, the forced interpolation of an original interior vertex in Sect. 6, the same constraint with C^1 in Sect. 7, and the interpolation of point and tangent direction in Sect. 8. Finally Sect. 9 draws conclusions and outlines the open questions still requiring future work.

2 Notation

We base our descriptions on the simplest quadratic and cubic subdivisions, since there are exact analogues for the higher orders which do not need detailed elaboration.

The first recursive subdivision algorithm was due to de Rham (1947), but this early work was forgotten until it was discovered some time after a lot of development had been stimulated by Chaikin's (1974) work. Chaikin's algorithm creates a new edge, called an *E-edge* in each old edge, whose vertices are at the $1/4$ and $3/4$ points of the old edge. It then creates a new edge, called a *V-edge* for each old vertex, joining the near vertices of the two new E-edges whose old edges met that old vertex (see Fig. 1). The new vertices, $P_i[j]$, at iteration i are given in

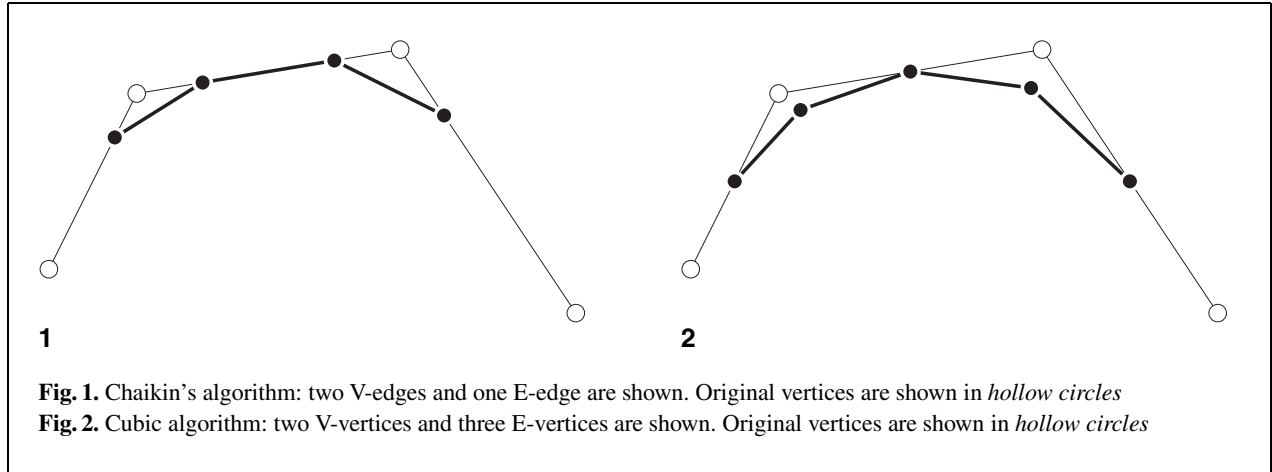


Table 1. Methods applied in various situations. An asterisk means that the method is applicable

	Method A	Method B	Method C	Method D	Method E
Situation 1					
Situation 2	*	*	*	*	*
Situation 3		*	*	*	*
Situation 4		*	*		*
Situation 5					*

terms of the level $i - 1$ by the following matrix notation¹:

$$\begin{bmatrix} P_{i+1}[2j - 1] \\ P_{i+1}[2j] \\ P_{i+1}[2j + 1] \\ P_{i+1}[2j + 2] \end{bmatrix} = \frac{1}{4} \begin{bmatrix} 1 & 3 & 0 & 0 \\ 0 & 3 & 1 & 0 \\ 0 & 1 & 3 & 0 \\ 0 & 0 & 3 & 1 \end{bmatrix} \begin{bmatrix} P_i[j - 1] \\ P_j[i] \\ P_i[j + 1] \\ P_i[j + 2] \end{bmatrix} \cdot (1)$$

are given in terms of the level $i - 1$ by a similar equation to (1) with the following matrix (See Fig. 2):

$$\frac{1}{8} \begin{bmatrix} 4 & 4 & 0 & 0 \\ 1 & 6 & 1 & 0 \\ 0 & 4 & 4 & 0 \\ 0 & 1 & 6 & 1 \end{bmatrix} \cdot (2)$$

The limit curve is made up of pieces, each a parabola, defined by the locations of three consecutive control points.

It was soon recognized (Forrest 1974; Riesenfeld 1975) that Chaikin's algorithm was a special case of knot insertion in the quadratic B-spline, and corresponding algorithms for all orders of B-spline were soon developed. The cubic algorithm creates for every old edge a new vertex, called an E-vertex, at its midpoint, and for every old vertex it creates an E-vertex. The new vertices, $P_i[j]$, at iteration i

¹ Note that the matrix actually shown in (1) and (2) are just parts of much larger matrices. The columns are all the same but shifted by 2 from column to column, and consist of values taken from a row of the Pascal triangle.

3 Recursive subdivision curves

This section identifies various curve situations and interpolation methods to provide a reference basis for our taxonomy. The situations are

1. The closed loop.
2. End-constraints for an open curve.
3. Interpolation of a single point maintaining only C^0 continuity.
4. Interpolation of a single point maintaining only C^1 continuity.
5. Interpolation of a single point matching a given tangent vector there.

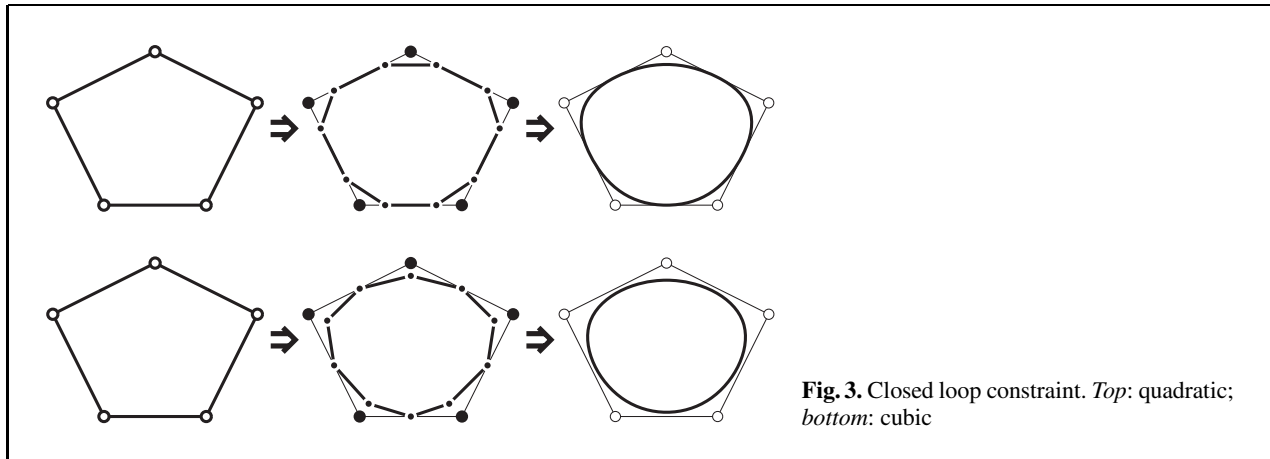


Fig. 3. Closed loop constraint. *Top:* quadratic; *bottom:* cubic

Methods which can be applied to these situations are

- A. Truncation.
- B. Polygon modification.
- C. Subdivision modification.
- D. Multiple vertices.
- E. Modification after the first iteration.

Not all methods apply in every situation. The applicability is summarized in Table 1, and the details are elaborated in the following sections.

4 Closed curves

The simplest curve case, without constraint, is that of the closed curve. Recursive division then operates uniformly, applying the same rule everywhere, giving at the limit a curve which typically interpolates none of the specified vertices but which has a global shape echoing that of the polygon. This is the base situation, where there are no complications, and we include it as the starting point (see Fig. 3).

5 Open curves

Unless the original polygon is very short, the two ends of an open curve may be treated independently, each end being considered on its own.

Normally we find it convenient to locate positively the end of the curve being defined, typically by requiring that the curve interpolates the end point of the given polygon. This can be achieved by each of our methods.

The methods are described in the following subsections, including the application of the standard refinement rules, which does not achieve interpolation of the ends (see Figs. 4, 5 and 6).

5.1 Method A: Truncation

The default case, if no other action is defined, is for only those vertices and spans which are defined by the standard rules to be generated. For instance, in the Chaikin case, the first span fails to generate a new edge from the old vertex at the end of the control polygon. As a result, the subdivision process continually shortens the polygon, the final limit curve reaching its end point at the midpoint of the first edge.

The original polygon can be sufficiently short: Only two vertices and only one edge in the quadratic case, or only three vertices and two spans in the cubic case. In the quadratic case, the limit curve reduces to just the single midpoint, whereas in the cubic case, the limit curve reaches its end at the limit point corresponding to the first internal vertex of the original polygon.

5.2 Method B: Polygon modification

This can be mapped into the case of Method A by modifying the polygon before applying the corresponding algorithm, and most of the interpolation constraints to be considered here are viewed in terms of modifying the given polygon to achieve the required interpolation.

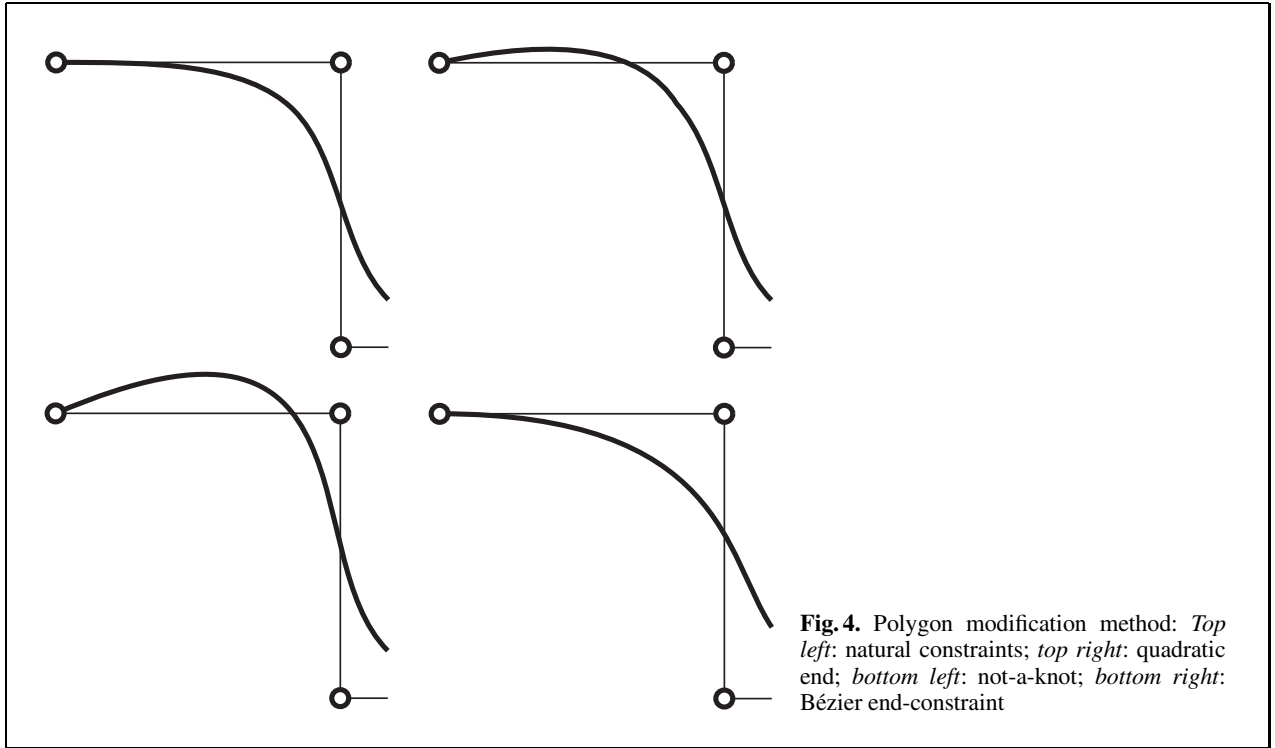


Fig. 4. Polygon modification method: *Top left:* natural constraints; *top right:* quadratic end; *bottom left:* not-a-knot; *bottom right:* Bézier end-constraint

In the quadratic case, the end point is moved to a new position such that its original position is at the midpoint of the end edge. The linear constraint giving this is

$$2A = A' + B,$$

and so A' is determined by

$$A' = 2A - B,$$

where A' and B' are the new vertices.

In the cubic case, vertices are generally moved by taking linear combinations in a way related to the linear combinations of the knot-inserted basis functions in the original basis functions. There are a number of variants that can be used (see Fig. 4):

- Adding a new vertex, Z' , at $2A - B$ gives the natural² end:

$$A' = A, \quad (3)$$

$$Z' = 2A - B. \quad (4)$$

² *Natural* is the term given to this constraint in the cubic interpolating spline, and it is characterized by having a zero second derivative at the end.

- Modifying the position of A and adding another point Z' as follows gives the quadratic end:

$$A' = \frac{6A + 2B - C}{7}, \quad (5)$$

$$Z' = \frac{18A - 15B + 4C}{7}. \quad (6)$$

- Modifying the position of A and adding another point Z' as follows gives the not-a-knot end:

$$A' = \frac{6A + 5B - 4C + D}{8}, \quad (7)$$

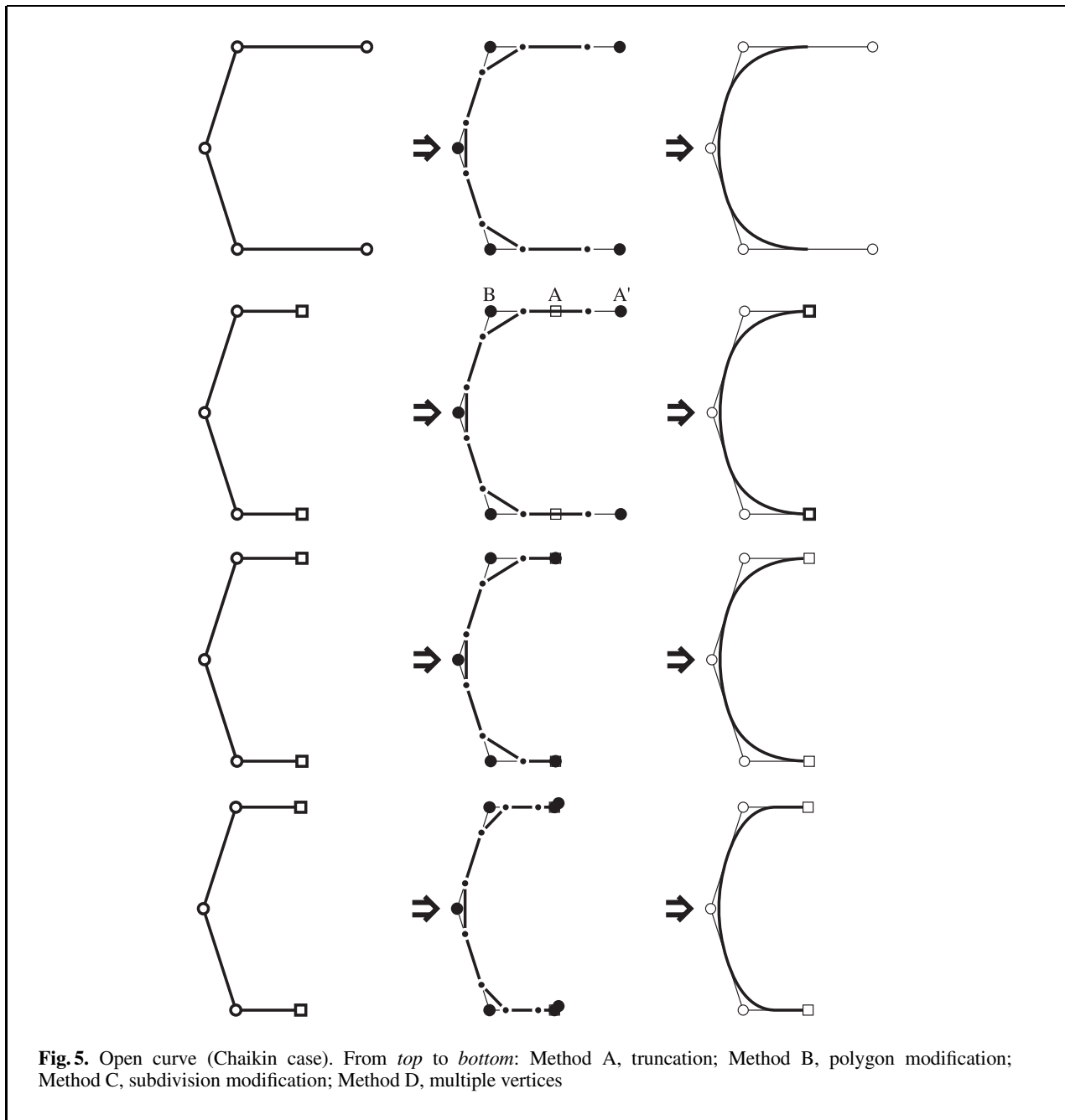
$$Z' = \frac{6A - 7B + 4C - D}{2}. \quad (8)$$

- Modifying the position of A and B as follows gives the equivalent of the Bézier end-constraint:

$$B' = \frac{3B - C}{2}, \quad (9)$$

$$A' = C + 6(A - B). \quad (10)$$

Once these changes are made, the standard subdivision can be applied at all subdivision steps. Note that



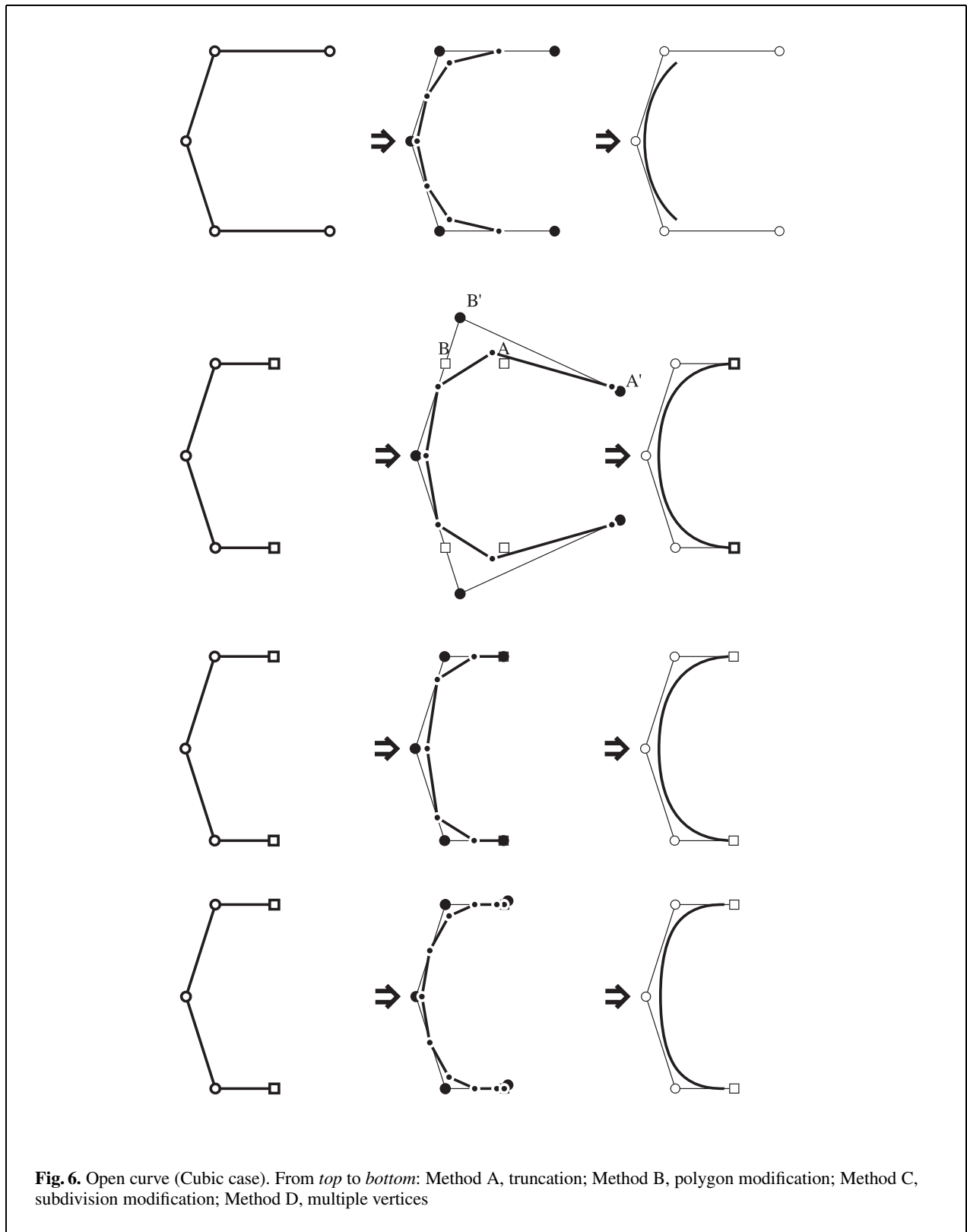
this does not deal satisfactorily with the case where the original polygon has only one edge.

5.3 Method C: Subdivision modification

The second option is to modify the construction in the end span, so that the new edge corresponding to

the first old edge runs from the end-point to the midpoint of the first old edge. Again the end vertex does not get a corresponding new edge.

In the quadratic case, the result of this construction has the end of the limit curve at the original first polygon vertex, and so this may be regarded as the first interpolation constraint. The cor-



responding matrix applied to the first (and last) three points is:

$$\begin{bmatrix} A' \\ B' \\ C' \\ \cdot \\ \cdot \end{bmatrix} = \frac{1}{4} \begin{bmatrix} 4 & 0 & 0 & 0 \\ 2 & 2 & 0 & 0 \\ 0 & 3 & 1 & 0 \\ \cdot & \cdot & \cdot & \cdot \end{bmatrix} \begin{bmatrix} A \\ B \\ C \\ \cdot \\ \cdot \end{bmatrix}. \quad (11)$$

In the cubic case, several constructions can achieve this result, each satisfying different curvature constraints at the end. The first is just to add a new vertex at the given end-point and is given by

$$\begin{bmatrix} A' \\ B' \\ C' \\ D' \\ \cdot \\ \cdot \end{bmatrix} = \frac{1}{8} \begin{bmatrix} 8 & 0 & 0 & 0 & 0 \\ 4 & 4 & 0 & 0 & 0 \\ 1 & 6 & 1 & 0 & 0 \\ 0 & 4 & 4 & 0 & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot \end{bmatrix} \begin{bmatrix} A \\ B \\ C \\ D \\ \cdot \\ \cdot \end{bmatrix}. \quad (12)$$

This method has the advantage of easy coding and is widely used, but has the disadvantage that the curvature at the end converges to zero as subdivision proceeds. This is analogous to the (most unnatural) “natural” interpolating spline.

A more useful construction may be given by the subdivision of a spline with pseudo-Bézier end constraints. This has the following matrix:

$$\begin{bmatrix} A' \\ B' \\ C' \\ D' \\ E' \\ F' \end{bmatrix} = \begin{bmatrix} 16 & 0 & 0 & 0 & 0 \\ 8 & 8 & 0 & 0 & 0 \\ 0 & 12 & 4 & 0 & 0 \\ 0 & 3 & 11 & 2 & 0 \\ 0 & 0 & 8 & 8 & 0 \\ 0 & 0 & 2 & 12 & 2 \end{bmatrix} \begin{bmatrix} A \\ B \\ C \\ D \\ E \\ F \end{bmatrix}. \quad (13)$$

Note that this treats the second control point as a slope control point, and so there is no knot there. This means that the number of new vertices after a subdivision step is slightly smaller than would be expected. The not-a-knot and quadratic end constraints give rise to matrices with negative terms, thus losing the variation-diminishing property. It is therefore probably more useful to modify the polygon and use the standard subdivision cases.

The single edge constraint can be handled by using the following special matrix:

$$\begin{bmatrix} A' \\ B' \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} A \\ B \end{bmatrix}. \quad (14)$$

5.4 Method D: Multiple vertices

This approach, of just superimposing sufficient vertices at the end, so that the truncation constraint lands up with the end of the curve at the right place, is applicable to all curve methods, but it is not particularly satisfactory. It leaves the curve with zero derivatives at the end, with the first span straight (demonstrating higher-order versions of l’Hopital’s rule).

Again, the curve of a single-edge case is the edge itself.

5.5 Method E: Modification after the first iteration

The simple technique of just moving the end vertex after each iteration to the required interpolation position is not recommended for quadratics, because it gives a non-polynomial end span with zero end curvature. For cubics, it is equivalent to the first alternative under Sect. 5.3.

6 Forced C^0 interpolation of an original interior vertex

The original polygon defines the sequence of the required limit curve, and in places away from the vertex to be interpolated, it also defines the shape in the standard way. However, near the interpolated vertex, V , the position of the vertex defines a point through which the curve has to pass, and the original mapping from polygon to limit curve is overridden.

If we require only C^0 interpolation and wish the interpolation to cause a deviation of the limit curve as small as possible, we may regard the vertex V to be interpolated as an end point as seen from both sides (see Figs. 7 and 8).

6.1 Method A: Truncation

This case is not relevant, since V must definitely be interpolated here.

6.2 Method B: Polygon modification

Regarding the vertex V to be interpolated as the common end-point of two halves of the curve breaks the curve at this point. The breaks must be rejoined at the end of the generation of the limit curve, which is somewhat artificial.

This case retains the convex hull properties.

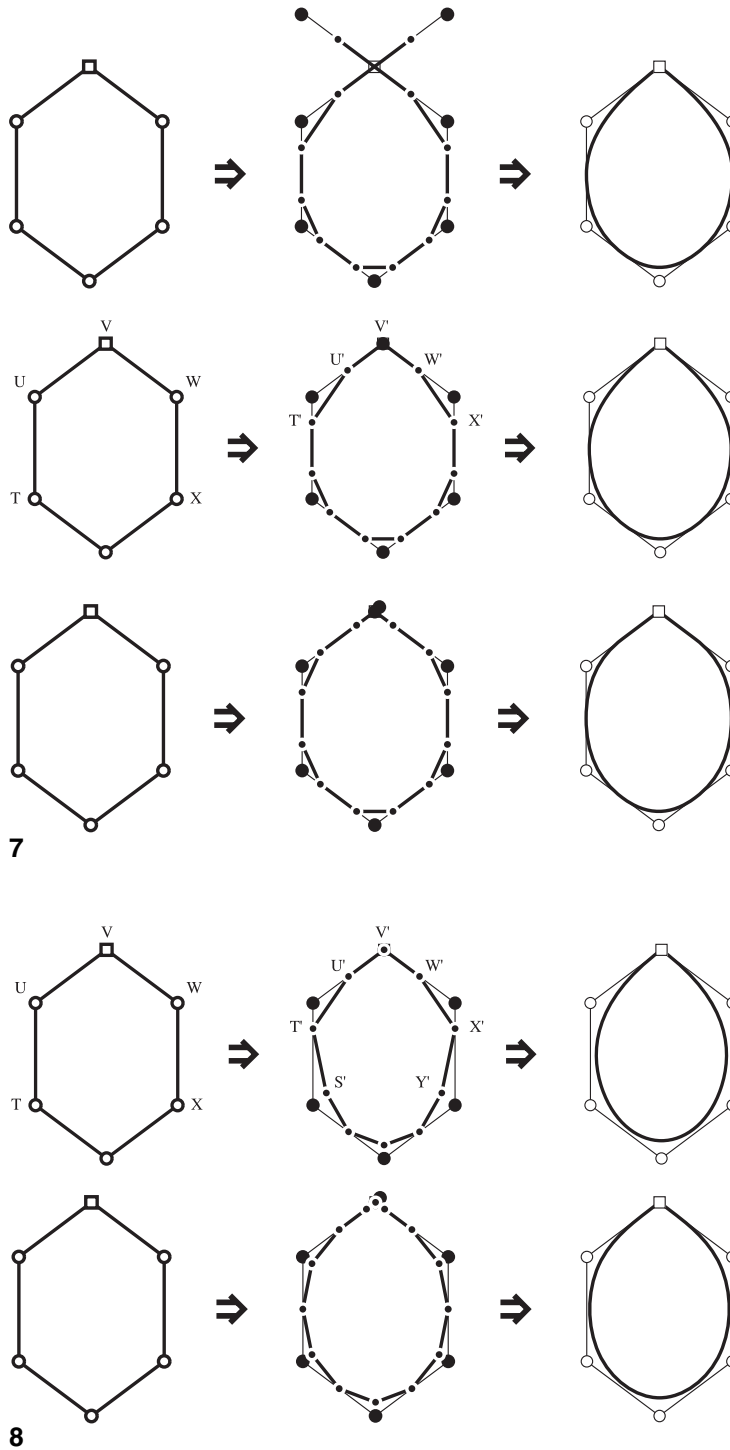


Fig. 7. C^0 vertex interpolation (quadratic case). *Top:* Method B; *middle:* Method C; *bottom:* Method D

Fig. 8. C^0 vertex interpolation (cubic case). *Top:* Method C; *bottom:* Method D

6.3 Method C: Subdivision modification

Regarding the vertex V to be interpolated as an end-point as in Sect. 5.3 with a modified construction rule gives what we want for the quadratic subdivision, but with the additional rule that no edge is generated from the old vertex V . This is given by the following matrix (here T' is the subdivided vertex of T , and so on):

$$\begin{bmatrix} T' \\ U' \\ V' \\ W' \\ X' \end{bmatrix} = \frac{1}{4} \begin{bmatrix} 1 & 3 & 0 & 0 & 0 \\ 0 & 2 & 2 & 0 & 0 \\ 0 & 0 & 4 & 0 & 0 \\ 0 & 0 & 2 & 2 & 0 \\ 0 & 0 & 0 & 3 & 1 \end{bmatrix} \begin{bmatrix} T \\ U \\ V \\ W \\ X \end{bmatrix}. \quad (15)$$

The matrix for cubic subdivision which is equivalent to using the Bézier end-constraint on each side is

$$\begin{bmatrix} S' \\ T' \\ U' \\ V' \\ W' \\ X' \\ Y' \end{bmatrix} = \frac{1}{16} \begin{bmatrix} 2 & 11 & 3 & 0 & 0 & 0 & 0 \\ 0 & 4 & 12 & 0 & 0 & 0 & 0 \\ 0 & 0 & 8 & 8 & 0 & 0 & 0 \\ 0 & 0 & 0 & 16 & 0 & 0 & 0 \\ 0 & 0 & 0 & 8 & 8 & 0 & 0 \\ 0 & 0 & 0 & 0 & 12 & 4 & 0 \\ 0 & 0 & 0 & 0 & 3 & 11 & 2 \end{bmatrix} \begin{bmatrix} S \\ T \\ U \\ V \\ W \\ X \\ Y \end{bmatrix}. \quad (16)$$

The fact that the middle of this is the same as the quadratic case is an interesting coincidence. Note that again fewer new edges are issued than is usual.

6.4 Method D: Multiple vertices

This method is just as unsatisfactory for achieving internal interpolation as for matching end-points because of the zero-length tangent there and because of the straight spans resulting from the multiplicity.

6.5 Method E: Modification after the first iteration

In the cubic case it is possible to force interpolation merely by moving the logically nearest point to the required interpolation position after every iteration. This is not recommended because it gives zero curvature on both sides of the link.

7 Forced C^1 interpolation of an original interior vertex

Here the limit curve is pulled to pass through an original vertex but smoothness is maintained. The interpolated point is not an end-point but a normal point on the curve. The curve can take different shapes as long as the C^1 constraint is satisfied. To achieve this, the interpolated vertex must, at some stage in the subdivision, either become the midpoint of an edge of the polygon defining the curve or else become the limiting position of a vertex, depending on whether the degree is even or odd. This can be done in the following ways (see Fig. 9):

7.1 Method A: Truncation

This case is again not applicable.

7.2 Method B: Polygon modification

The simple procedure to achieve the somewhat complex objective is to create a modified polygon by moving the vertex originally at the interpolation point. It goes to such a position that the midpoint of the V -edge replacing that vertex at the first subdivision, or the vertex limit, lies at the interpolation position, thus ensuring the required interpolation. The linear constraint for the quadratic Chaikin is (B is the point to interpolate)

$$8B = 6B' + A + C, \quad (17)$$

and so B' may be computed by

$$B' = \frac{(8B - A - C)}{6}; \quad (18)$$

there are corresponding rules for higher-order constructions, derived from the equation which links a point on the curve to a minimal collection of control points. For example, in the cubic case we have

$$B' = \frac{(6B - A - C)}{4}. \quad (19)$$

If interpolation points are isolated, each perturbation may be computed independently. If, however, two or more consecutive vertices are to be interpolated, the linear system consisting of the equations each must satisfy is coupled, and must be solved as a whole. For example, if only two consecutive vertices are to be

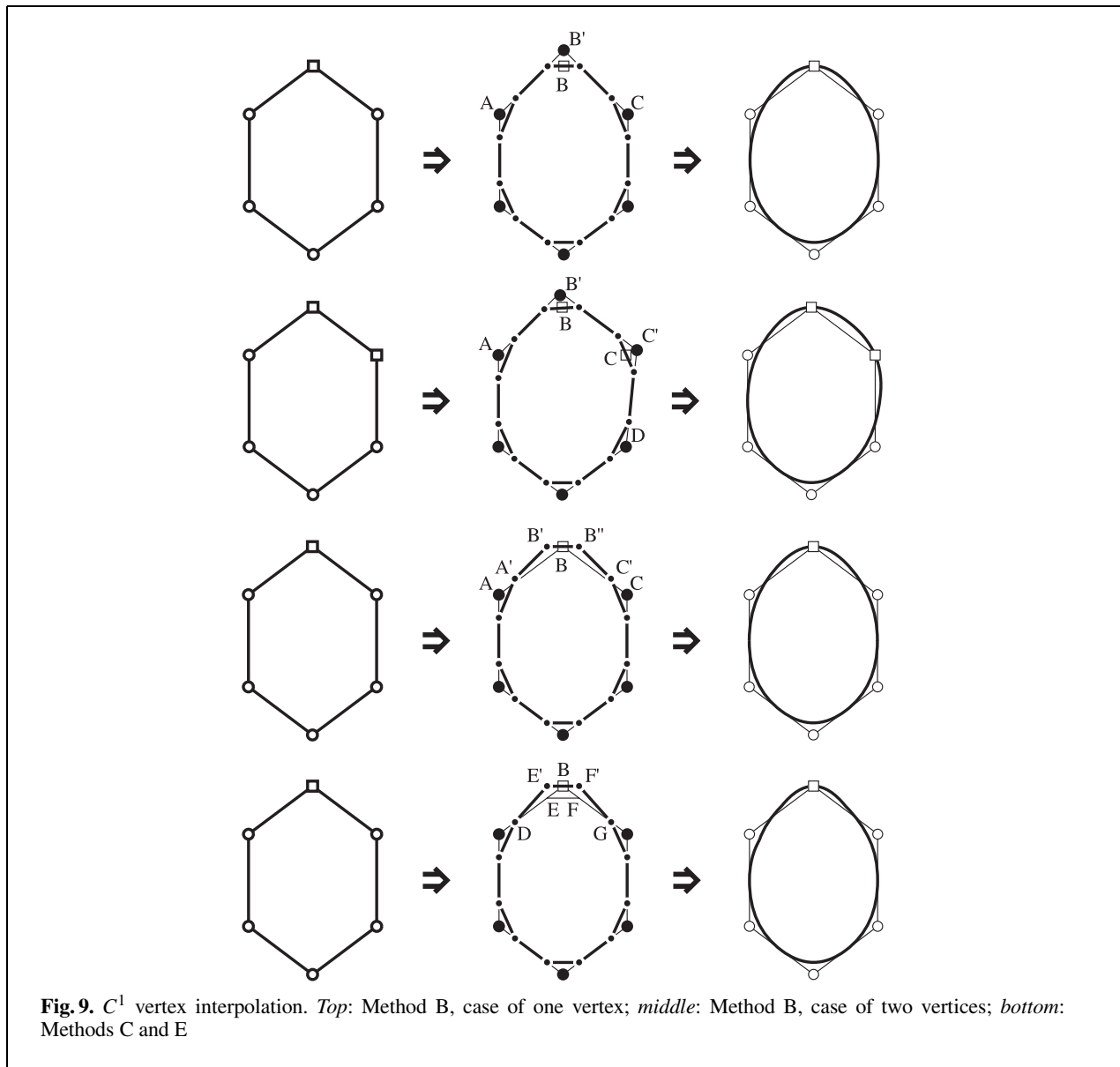


Fig. 9. C^1 vertex interpolation. Top: Method B, case of one vertex; middle: Method B, case of two vertices; bottom: Methods C and E

interpolated we have, in the Chaikin quadratic case, the following linear equations (see Fig. 8):

$$8B = 6B' + A + C' \quad (20)$$

$$8C = 6C' + B' + D, \quad (21)$$

so that we need to solve the linear system

$$\begin{bmatrix} 6 & 1 \\ 1 & 6 \end{bmatrix} \begin{bmatrix} B' \\ C' \end{bmatrix} = \begin{bmatrix} 8B - A \\ 8C - D \end{bmatrix}. \quad (22)$$

In the limit we have an interpolating spline. The linear system is always constrained well enough for an

iterative scheme to function well, but the matrix is highly diagonal (tridiagonal in the quadratic and cubic cases), and so a direct solution will be just as fast. Adding C^1 interpolation constraints loses the variation-diminishing convex hull properties of the limit curve.

7.3 Method C: Subdivision modification

Different subdivision rules are applied to the spans sharing the interpolated vertex. In the quadratic case,

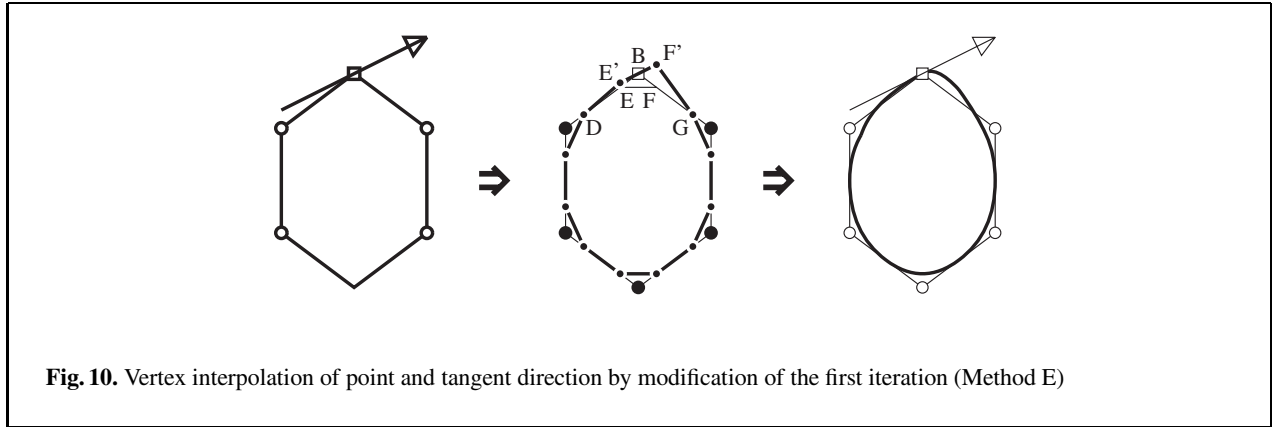


Fig. 10. Vertex interpolation of point and tangent direction by modification of the first iteration (Method E)

for instance, if the spans AB and BC are common to the vertex to be interpolated, B , then their corresponding subdivided edges, $A'B'$ and $B''C'$, are such that B is the midpoint of $B'B''$. Instead of taking a span, we take a sequence of 3 points, A , B , and C , where B is to be interpolated, and replace them by 4 points, A' , B' , B'' , and C' given by

$$\begin{bmatrix} A' \\ B' \\ B'' \\ C' \end{bmatrix} = \frac{1}{24} \begin{bmatrix} 17 & 8 & -1 \\ 3 & 24 & -3 \\ -3 & 24 & 3 \\ -1 & 8 & 17 \end{bmatrix} \begin{bmatrix} A \\ B \\ C \end{bmatrix}. \quad (23)$$

If a span has no vertex to be interpolated, then it is divided in the usual way.

In the cubic case, if at each iteration the vertex to be interpolated is C in the sequence $ABCDE$, the subdivision becomes

$$\begin{bmatrix} A' \\ B' \\ C' \\ D' \\ E' \end{bmatrix} = \frac{1}{32} \begin{bmatrix} 4 & 23 & 6 & -1 & 0 \\ 0 & 12 & 24 & -4 & 0 \\ 0 & 0 & 32 & 0 & 0 \\ 0 & -4 & 24 & 12 & 0 \\ 0 & -1 & 6 & 23 & 4 \end{bmatrix} \begin{bmatrix} A \\ B \\ C \\ D \\ E \end{bmatrix}. \quad (24)$$

If interpolation points are isolated, the new points can be computed easily. However, things become complicated if two or more consecutive vertices are to be interpolated.

In fact, this modified process is applied only at the first subdivision, because with the even-order constructions there is no vertex which becomes the one to be interpolated after that first subdivision. Odd-order constructions, however, do have a natural descendant of the vertex to be interpolated, and it is therefore possible to use a matrix which contains a unit row, so that that point remains invariant during

the subdivision. In the case when all vertices are to be interpolated, we find this reducing to the approach of Dyn et al. (1987).

7.4 Method D: Multiple vertices

This is not applicable to C^1 interpolation.

7.5 Method E: Modification after first iteration

It is also possible to perform the first iteration, and then modify the new vertices adjacent to the interpolation point. This obviously gives a narrower perturbation to the curve, with shorter wavelength effects. It does allow each interpolation point to be handled independently, avoiding the need to solve the linear system. For instance, using Chaikin's subdivision, some transformations can be applied to the V-edges (edges generated from the vertices in the subdivision process), to put the interpolated vertices at their midpoints. A V-edge, EF , depending on three initial control vertices, A , B and C , is transformed into another edge, $E'F'$, as follows:

$$\begin{bmatrix} E' \\ F' \end{bmatrix} = \begin{bmatrix} E \\ F \end{bmatrix} + \frac{1}{4} \begin{bmatrix} -1 & 2 & -1 \\ -1 & 2 & -1 \end{bmatrix} \begin{bmatrix} A \\ B \\ C \end{bmatrix}. \quad (25)$$

It should be mentioned that the subdivision matrix can be combined with this transformation matrix to generate points at the proper positions and put the original vertices at the midpoints of the new V-edges (Nasri 1997b). In short, the new rules consist of taking every triplet, (A, B, C) , of three consecutive vertices and generating a V-edge, $E'F'$. In the cubic case

this is even simpler, essentially being just the application of Method B above after the first iteration, thus moving just a single point.

8 Interpolation of point and tangent direction

Methods A and D cannot be used to interpolate tangent direction. For example, Method D results in zero tangent. In general, Method C consists of using at the first iteration subdivision coefficients different from those used in the subsequent subdivisions. The new coefficients should introduce in the first subdivided polygon an edge collinear with the given tangent direction. Method E can be applied to interpolate both position and a direction at one of the polygon points as follows: In Chaikin's example, the V-edges are subject to a sequence of transformations which first interpolate the vertex and then force the direction of the V-edge to coincide with that of the one given. The transformation matrix involved here naturally depends on the direction of the tangent. Basically, the matrix in Sect. 7.5 is concatenated with a matrix that projects the edge $E'F'$ in the direction of the tangent, giving $E''F''$. The best direction of projection is yet to be determined (see Fig. 10). The same technique, of carrying out one or more steps of refinement and then modifying the polygon, can in principle be used to control higher derivatives of the curve, at least up to the order of the curve. Thus pointwise second could be controlled for the quadratic, and second or even third derivatives for the cubic. Since the curve does not have continuity of the highest derivative thus specifiable, it is not obvious that this is really useful, but the mechanism for it does exist.

9 Conclusion

In this paper we have described various interpolation situations for curves, and identified five methods for dealing with them. While these methods are clearly linked, they involve different implementations, and by using them software may be built which will enable a designer to define an appropriate control polygon which converges to a curve satisfying the required constraints.

Acknowledgements. Thanks are due to the American University of Beirut, the Lebanese National Council for Scientific Research, and the

Royal Society of London for supporting this work. The authors would like to thank the anonymous referees for their valuable comments.

References

1. Ball A, Storry D (1984) Recursively generated B-spline surfaces. Proceeding of CAD 84, Butterworths, Brighton, pp 112–119
2. Biermann H, Levin A, Zorin D (2000) Piecewise smooth subdivision surfaces with normal control. SIGGRAPH '00 Conference Proceedings. ACM, New York, pp 113–120
3. Catmull E, Clark J (1978) Recursively generated B-spline surfaces on arbitrary topological meshes. *Comput Aided Des* 10(6):350–355
4. Chaikin GM (1974) An algorithm for high speed curve generation. *Comput Graph Image Process* 3(12):346–349
5. de Rham G (1947) Un peu de mathématiques a propos d'une courbe plane. *Elem Math* 2:73–76,89–97
6. Doo DW (1978) A method of smoothing highly irregular polyhedrons. In: Proceedings of Computer-Aided Geometric Design, IEEE Computer Society, Bologna
7. Doo D, Sabin M (1978) Behavior of recursive division surfaces near extraordinary points. *Comput Aided Des* 10(6):356–360
8. Dyn N, Gregory JA, Levin D (1987) A 4-point interpolatory subdivision scheme for curve design. *Comput Aided Geom Des* 4:257–268
9. Dyn N, Gregory JA, Levin D (1990) Uniform Subdivision algorithms for curves and surfaces. In: Mason JC, Cox MG (eds) Algorithms for Approximation II. Chapman and Hall, New York, pp 278–295
10. Dyn N, Levin D (1990) A butterfly subdivision scheme for surface interpolation with tension control. *ACM Trans Graphics* 9:160–169
11. Dubuc S, Deslauriers G (1989) Symmetric iterative interpolation processes. *Constr Approx* 5:49–68
12. Forrest AR (1974) Notes on Chaikin's algorithm. Computational Geometry Memo CGP 74-1, University of East Anglia, Norwich
13. Gregory JA, Qu R (1996) Non-uniform corner cutting. *Comput Aided Geom Design* 20:1–10
14. Habib A (1996) Three approaches to building curves and surfaces in CAGD. PhD dissertation, Rice University, Houston
15. Hoppe H, DeRose T, Duchamp T, Halstead M, Jin H, McDonald J, Schweitzer J, Stuetzle W (1994) Piecewise smooth surface reconstruction. *Comput Graph* 28(3):295–302
16. Kobbelt L (1996) Interpolatory subdivision on open quadrilateral nets with arbitrary topology. *Comput Graph Forum* 15:409–420
17. Levin A (1999a) Interpolating nets of curves by smooth subdivision surfaces. In: SIGGRAPH '99 Conference Proceedings. ACM, New York, pp 57–64
18. Levin A (1999b) Combined subdivision schemes for the design of surfaces satisfying boundary conditions. *Comput Aided Geom Des* 16:345–354
19. Loop CT (1994) Smooth spline surfaces over irregular meshes. *Comput Graph* 24(3):303–310

20. Nasri A (1987) Polyhedral Subdivision methods for free form surfaces. *ACM Trans Graph* 8(1):89–96
21. Nasri A (1991) Surface Interpolation on irregular network with normal constraints. *Comput Aided Geom Des* 8(1):89–96
22. Nasri AH (1997a) Curve interpolation in recursively generated surfaces over arbitrary topology. *Comput Aided Geom Des* 14(1):13–30
23. Nasri AH (1997b) A non-uniform Doo-Sabin subdivision scheme with boundary control. Technical Report TR97/1, American University of Beirut
24. Nasri AH (2000c) Recursive subdivision of polygonal complexes and its applications in CAGD. *Comput Aided Geom Des* 17(7):1–25
25. Nasri AH (1998) A 4-sided approach for interpolating B-spline curves by Recursive subdivision surfaces. *Visual Comput* 14(7):343–353
26. Nasri AH, Sabin MA (2001) Designing Subdivision surfaces from various constraints. Technical Report TR-01/3, American University of Beirut
27. Peters J (1993) Smooth free-form surfaces over irregular meshes generalizing quadratic splines. *Comput Aided Geom Des* 10(4):347–361
28. Peters J, Reif U (1998) Analysis of generalized B-spline subdivision algorithms. *SIAM J Numer Anal* 35(2):728–748
29. Peters J, Reif U (1997) The simplest subdivision scheme for smoothing polyhedra. *ACM Trans Graph* 16(4):420–431
30. Prautzsch H (1995) Analysis of C^k -subdivision surfaces at extraordinary points. Preprint, IBDS, Universität Karlsruhe, presented at Oberwolfach
31. Riesenfeld RF (1975) On Chaikin's algorithm. *IEEE Comput Graph Appl* 4(3):304–310
32. Reif U (1995) A unified approach to subdivision algorithms near extraordinary vertices. *Comput Aided Geom Des* 12:153–174
33. Reif U (1999) Analyse und Konstruktion von Subdivisionsalgorithmen für Freiformflächen beliebiger Topologie D93. *Habil-Schr, Universität Stuttgart*. Shaker, Aachen
34. Sabin MA (1986) Recursive subdivision. In: Gregory JA (ed) *The Mathematics of Surfaces*. Clarendon, Oxford, pp 269–282
35. Schweitzer J (1996) Analysis and application of subdivision surfaces. PhD dissertation, Department of Computer Science and Engineering, University of Washington, Seattle
36. Sederberg TW, Zheng J, Sewell D, Sabin M (1998) Non-uniform recursive subdivision surfaces. In: *SIGGRAPH '98 Conference Proceedings*. ACM, New York, pp 387–394
37. Zorin D, Schröder P (2000) Subdivision for modeling and animation. In: *SIGGRAPH '00 Course Notes*. ACM, New York
38. Warren J (1995) Binary subdivision schemes for functions over irregular knot sequences. In: Dæhlen M, Lyche T, Shumaker L (eds) *Mathematical Methods for Curves and Surfaces*. Vanderbilt University, Nashville, pp 543–562



AHMAD H. NASRI is an associate professor at the Dept. of Math. and Computer Science, American University of Beirut, Lebanon. He received a B.S. degree in Mathematics from the Lebanese University (Lebanon) in 1978, and a Ph.D. degree in Computer Graphics from the University of East Anglia, U.K., in 1985. He was a research visitor at MIT, Arizona State University, Purdue University, Seoul National University (Korea), Cambridge University (UK), and Bremen University (Germany). He is on the editorial board of the *International Journal of CAD/CAM* and the *International Journal of Shape Modeling* and has served on the PC committee of several international conferences. He is the Co-editor (with M. Sabin) of the TVC special issue on subdivision (2002). Since 1982 he has been involved in promoting subdivision surfaces and providing solutions to various interpolation constraints on them. His research interests include recursive subdivision in geometric modeling and computer graphics, data visualization, and computer graphics in education.



MALCOLM SABIN was first involved with subdivision surfaces in the mid 1970s – work which led to the joint paper with D.W.H. Doo. Ever since then he has maintained his interest in the subject, making occasional contributions to conferences. More recently, with the upsurge of commercial interest in subdivision, he has been involved in the MINGLE project on multiresolution methods. He gave tutorials on the subject at *Shape Modeling 2001* and at the *Primus Summer School*. His current interest within subdivision is in categorisation and analysis so that we can be certain that all possible refinement schemes can be considered by those responsible for choosing which variant to embody in commercial products, and that we really understand their properties.