Choosing nodes in parametric curve interpolation

E T Y Lee

In parametric curve interpolation, the choice of the interpolating nodes makes a great deal of difference in the resulting curve. Uniform parametrization is generally unsatisfactory. It is often suggested that a good choice of nodes is the cumulative chord length parametrization. Examples presented here, however, show that this is not so. Heuristic reasoning based on a physical analogy leads to a third parametrization, (the 'centripetal model'), which almost invariably results in better shapes than either the chord length or the uniform parametrization. As with the previous two methods, this method is 'global' and is 'invariant' under similarity transformations. (It turns out that, in some sense, the method has been anticipated in a paper by Hosaka and Kimura.)

computer-aided design, parametric curve interpolation, heuristic reasoning, centripetal model

Consider the problem of passing a planar curve through a given set of points $P_i = (x_i, y_i)$, $0 \leqslant i \leqslant n$, in the order indicated. It is assumed that no two consecutive points are the same. The curves of particular interest are those defined parametrically by *vector-valued* polynomials of degree *n*, or splines, generally of degrees lower than *n*. To fix ideas, the discussion will start with polynomials; what is said applies equally to splines with minor modifications. This is done partly for simplicity in description (the idea being the same), and partly because the effect of node choices is most pronounced in the polynomial context. Thus, let the curve be $P(t) = (x(t), y(t))$, with $x(t)$, $y(t)$ polynomials of degree *n*. The interpolating conditions are

$$P(t_i) = P_i, \quad 0 \leqslant i \leqslant n \qquad (1)$$

where $t_0 < t_1 < \cdots < t_n$ are certain chosen parameter values called the interpolating *nodes*. Equation (1) is a nonsingular linear system for the $n+1$ vector coefficients of $P(t)$. Clearly, different choices of the nodes $(t_i)_0^n$ lead to different curves. A designer is often concerned with obtaining a 'fair' or 'pleasing' curve through the data points, so the question is: how are the nodes chosen to best achieve this? Thus, the problem is one of parametrization of the data points.

In this paper, symbols in capital letters are reserved for points or vectors in the plane, with $|P|$ denoting the Euclidean norm of P.

Boeing Commercial Airplanes, PO Box 3707, M/S 6F-27, Seattle, WA 98124, USA

Perhaps a few words might be inserted here to avoid any possible confusion. In the usual *function* interpolation setting, the problem is of the form $P_i = (x_i, y_i)$ where the x_i are increasing, and one seeks a *real-valued* polynomial $y = y(x)$ so that $y(x_i) = y_i$. This is identical to the *vector-valued* polynomial

$$P(x) = (x, y(x))$$

with *x* as the parameter, except with the important distinction that here the interpolating conditions $y(x_i) = y_i$ are

$$P(x_i) = P_i, \quad 0 \leqslant i \leqslant n$$

that is, the nodes are fixed as the *x*-coordinates of the data points; there is no room for choice.

From now on the parametric interval shall be normalized so that $t_0 = 0$, $t_n = 1$, to conform to usual customs. One simple choice of the nodes is the *uniform* parametrization,

$$t_i = i/n, \quad 0 \leqslant i \leqslant n \qquad (2)$$

But this is generally unsatisfactory, and is usually dismissed for the obvious reason that the nodes have nothing to do with the distribution of the data points. (See Figure 1(a). The appearance of the cusp may seem

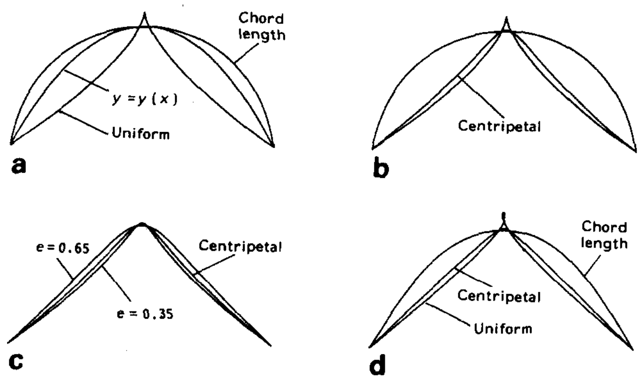

Figure 1. Cubic polynomials (a–c) and natural cubic splines (d) through 4 data points: {x, y} = {(0, 0), (26, 24), (28, 24), (54, 0)}. The data points are at the common intersections of the curves. Note that in (c), with increasing (or decreasing) exponent e from 0.5, the curve tends in shape to that of the chord length (or uniform) parametrization. In (a), y = y(x) is the ordinary polynomial obtained with the nodes 0, 26/54, 28/54, 1

puzzling to the unwary. However, note that even though $x(t)$ and $y(t)$ are polynomials, neither x nor y need be a polynomial function of the other. A cusp may occur where $x'(t)$ and $y'(t)$ both vanish.) It has been suggested[1–4] (in the spline context) and generally accepted that a better choice is the cumulative *chord length* parametrization; that is, $t_0 = 0$, and

$$t_i - t_{i-1} = \frac{|P_i - P_{i-1}|}{\Sigma_{j=1}^{n} |P_j - P_{j-1}|}, \quad 1 \leqslant i \leqslant n \qquad (3)$$

If the data points are more or less evenly spaced, this is approximately the uniform parameterization. So the advantage of this choice must be evidenced when the points are quite unevenly spaced. However, a look at some of the examples (especially Figures 2 and 4) shows that the resulting curve can actually be extremely disturbing: it may wander far off from the defining data polygon. Even for the curve in Figure 1(a) obtained by assignment (3), it may be questionable whether the result approaches what the designer has in mind. If a full rounded curve as in Figure 1(a) is desired, the designer would most likely want to specify the four data points nearly evenly spaced; whereas with the nonuniform distribution of points there, most likely the designer has something else in mind: something that's rounded at the top but close to being straight on the two sides. Similar comparisons of the effects of uniform against chord length parametrizations have been noted elsewhere[5,6].

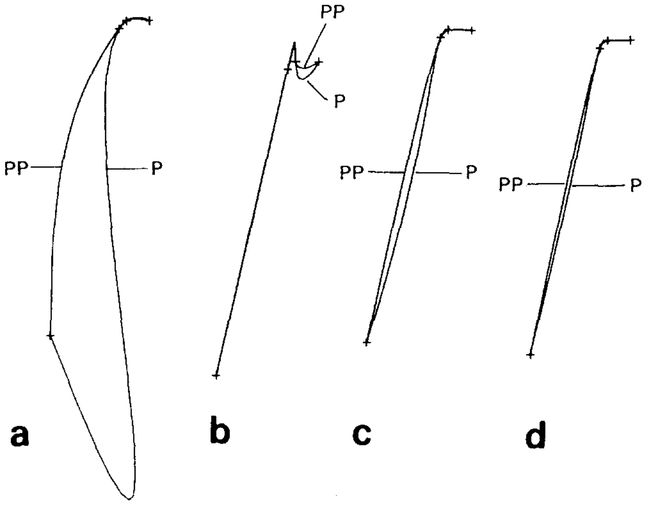

Figure 2. Cubic polynomials (denoted by P) and natural cubic splines (denoted by PP) through 4 points: $\{x, y\} = \{(0, 0), (9, 39), (10, 40), (13, 40)\}$. (a) Chord length, (b) uniform, (c) centripetal, (d) e = 0.37

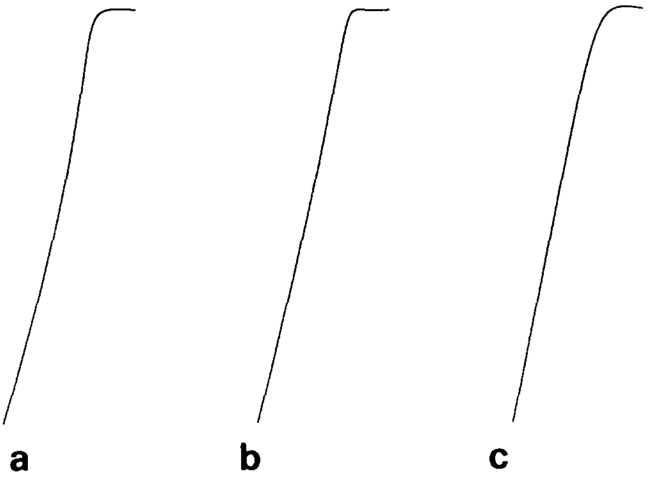

Figure 3. Cubic polynomials, with the same data as in Figure 2: (c) is obtained by directly varying the nodes. This example is particularly sensitive to variation in node locations. (a) Centripetal; (b) e = 0.37; (c) $t_1 = 0.65$, $t_2 = 0.76$

The terms 'fair' and 'pleasing' are vague and difficult to quantify; and so it is hard to say what constitutes a good set of nodes. The comments above, on the intention of the designer, suggest that at least one should strive for a smooth curve that more or less conforms in shape to the polygonal curve defined by the data points. It may be argued that the designer, not satisfied with a given result, can always try to improve by adding in more points. (This is, of course, only practical in the spline context; for polynomials the degree will be correspondingly raised.) But this may take many iterations, and it is clearly desirable to use as *small* a number of data points as possible for an intended shape. Furthermore, there are other applications calling for interpolation in which data are obtained from processes where there is no control over the spacing of the points.

Some recent works have appeared that result in some improvements over the chord length parametrization. In Topfer[7] and Marin[8], the nodes are derived through optimization techniques. But optimization methods are expensive, and moreover it is not entirely clear what objective function should be used (see later discussions). Other experiments have been reported in Hartley and Judd[6].

In this paper a node assignment is proposed which is as simple as but almost invariably works much better than the chord length parametrization. The designer need only input a small number of points; the resulting curve conforms well to the data polygon. The reasoning here is heuristic but is based on a physical analogy. Further refinements should be possible, but even at this stage it should be very useful in practice, especially since most CAD systems at present use either the uniform or the chord length parametrization.

CENTRIPETAL MODEL

To start, it may be asked why the chord length parametrization seems to have been generally taken for granted (indeed, so much so that it has sometimes been named the 'natural parametrization'[9]). Apart from some justification offered in Epstein[10], the main reason for this choice seems to be that it roughly approximates the arc length parametrization. In fact, it has been suggested (though not really recommended) that if desired, one may iterate until the parametrization becomes the arc length parametrization of the resulting curve (see Ahlberg *et al.*[1], p. 51; Späth[4], p. 65; de Boor[2], p. 318; Brodlie[3], p 19). But why arc length? In analysing an *existing* curve, it is customary to use arc length as the parameter since, among other things, the first two derivatives have geometric significance, being the unit tangent vector and the curvature vector, respectively.

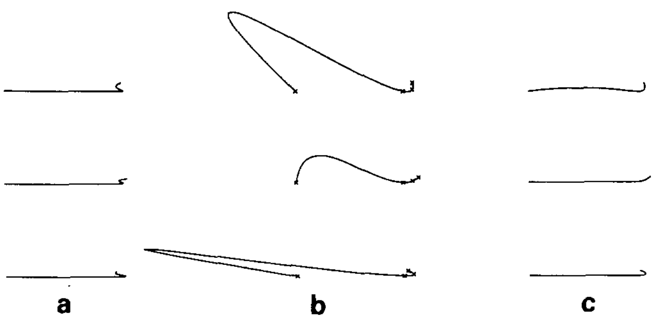

Figure 4. Cubic polynomials through 4 points. The last arm of the data polygon is rotated; the nodes remain the same in each method. Data for the top figure: $\{x, y\} = \{(0, -0.15), (9.2, -0.15), (10, 0), (10, 0.5831)\}$. In the middle figure, the last point is changed to (10.5, 0.3). In the bottom figure, the last point is (9.5, 0.3). (a) Uniform, (b) chord length, (c) centripetal

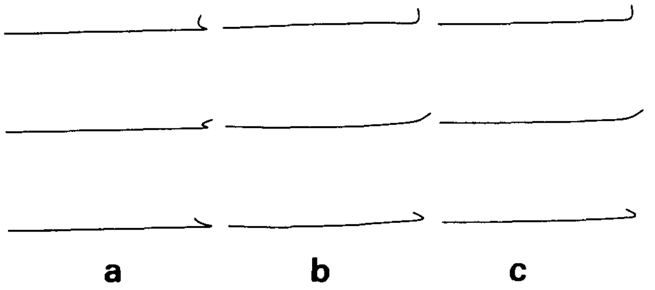

Figure 5. Same data as in Figure 4, but with natural cubic splines. (a) Uniform, (b) chord length, (c) centripetal

But here the problem is of a different nature: here the aim is to *create* a curve through a given set of points, and it is not clear why one should strive for the arc length parametrization, nor is it clear that the suggested iterations should even converge. If the curve is regarded as the trajectory of a particle, chord length parametrization implies approximately uniform speed of traversal. Does this somehow ensure 'fair' shape?

Imagine someone driving on a curved highway. It becomes quite obvious that the driver does not try to maintain an approximately uniform speed. Anticipating a sharp turn in the road, the driver would significantly slow down, because the centripetal force necessary to keep the driver in a natural posture or to keep the car from skidding becomes too great if the speed is not reduced. A little closer to the problem described here is the picture of the driver choosing his travel $P(t)$ on an open field to pass successively through a given set of points, somewhat like a slalom course. (By 'travel' or 'trajectory', the function $P(t)$ is emphasized, not just the locus of the positions.) The considerations are essentially the same. In a way, 'fair' or 'pleasing' is identified with the comfort of the driver during the travel.

The acceleration of the car moving according to $P(t)$ is (by chain rule)

$$\frac{d^2P}{dt^2} = \frac{d^2s}{dt^2}\frac{dP}{ds} + \left(\frac{ds}{dt}\right)^2 \frac{d^2P}{ds^2}$$
$$= \frac{d^2s}{dt^2} T + \left(\frac{ds}{dt}\right)^2 \kappa N$$

where s, T, N and κ denote respectively the arc length, unit tangent, unit normal and curvature at $P(t)$. It is assumed that the driver is not reckless; that is, that there is no sudden change in speed, so that the tangential acceleration d^2s/dt^2 may be ignored. Thus the comfort of the driver or the safety of the car depends on having the normal or centripetal acceleration

$$\kappa(ds/dt)^2$$

varying gently within certain bounds.

The aim here is to obtain an *a priori* estimate for a suitable choice of the arrival time t_i for P_i. One reason that the role of d^2s/dt^2 is discounted is that otherwise a simple choice seems hopeless. Since the curve does not yet exist, some simplifying assumptions are necessary. All quantities will now be replaced by their average values. For the arc between P_{i-1} and P_i, the average speed is roughly

$$\frac{ds}{dt} \approx \frac{|P_i - P_{i-1}|}{t_i - t_{i-1}}$$

Moreover, assuming no inflections, the average curvature over the arc is the change in tangent directions divided by the arc length (recall $\kappa N = dT/ds$). Thus

$$\kappa \approx \frac{\delta_i}{|P_i - P_{i-1}|} \tag{4}$$

where δ_i is the angular change from P_{i-1} to P_i. Finally, it is *postulated* that, for the comfort of the driver, the centripetal force over this arc should be kept roughly proportional to this angular change,

$$\kappa(ds/dt)^2 \approx \delta_i/c^2 \tag{5}$$

This last assumption is made mainly for the simplification below. Note that there is no simple quantitative criterion as to the 'optimal' size of the centripetal force to begin with. Admittedly assumption (5) is rather arbitrary and open to question. Still, it does not seem unreasonable: for a given $|P_i - P_{i-1}|$, the larger the δ_i the more centripetal force the driver expects to tolerate. Putting these together,

$$\frac{\delta_i}{|P_i - P_{i-1}|}\left(\frac{|P_i - P_{i-1}|}{t_i - t_{i-1}}\right)^2 \approx \delta_i/c^2$$

If $\delta_i \neq 0$, the following estimate is obtained:

$$t_i - t_{i-1} = c|P_i - P_{i-1}|^{1/2}$$

which will be kept in use even when $\delta_i = 0$. Normalizing, a parametrization by the cumulative *square root* of the chord lengths is reached: $t_0 = 0$, and

$$t_i - t_{i-1} = \frac{|P_i - P_{i-1}|^{1/2}}{\sum_{j=1}^{n}|P_j - P_{j-1}|^{1/2}}, \quad 1 \leqslant i \leqslant n \tag{6}$$

With nearly equally spaced points, all three parametrizations (2), (3) and (6) are about the same. In all cases (except one) tested, where the data points are unevenly spaced, the square root method (6) invariably gives much better results than methods (2) or (3); and this, in the final analysis, is the only justification for the various drastic simplifications made above. The one *exception* is when all the data points are distributed monotonically along a straight line, in which case clearly the chord length method (3) gives a uniform sweeping of the line segment. The centripetal model is based on the curving of the road; if the road is straight, there is no need to vary speed. If assignment (6) were to be used in this situation, then with certain distributions of points, the resulting curve, even though on a straight line, would have a small 'doubling-back' or 'overshoot' effect near the end points, as will happen even more so with method (2). (It can be shown that such an effect, however, cannot occur if P(t) is quadratic.) This point could perhaps be dismissed by an appeal again to the psychology of the designer. If a single straight line segment is intended, it suffices to specify the two end points, or if for some reason the designer wants to input several points, most likely the points would be chosen more or less spaced evenly, in which case all three methods perform similarly. In case the data polygon is not a line but contains several consecutive collinear or nearly collinear segments, test examples show that the resulting curves are still very satisfactory. See Figures 4, 5, 7 and 8.

The method described here still does not fully take into account the geometry of the data polygon. It ignores the turnings of the successive arms of the polygon (due to the assumption (5)). However, test examples are quite satisfactory. See Figures 4 and 5, where one arm is rotated while all arm lengths are fixed.

EXAMPLES AND COMMENTS

The considerations in the previous section do not depend on the curve being in the class of polynomials or splines. In order for equation (1) to be solvable, P(t) can be a 'generalized polynomial' spanned by a system of $n+1$ functions *unisolvent* over [0, 1]. But only polynomials and splines seem to be of current interest in CAD.

The case of interpolation by splines does not exactly fit into the pattern of equation (1), and so perhaps needs some explanation. Generally the *knots* of the spline need not be the same as the *nodes* of interpolation, and for a system like equation (1) to be solvable, there is an interplay between nodes and knots involved (Schoenberg–Whitney). Here, for *odd* degree splines, the knots will simply be chosen to coincide with the nodes. Specifically, with d the degree, the knots are $\xi_{-d} = \cdots = \xi_0 = t_0$,

$$\xi_i = t_i, \quad 1 \leqslant i \leqslant n-1$$

and $\xi_n = \cdots = \xi_{n+d} = t_n$. The dimension of such a spline space is $n+d$, so in addition to the $n+1$ interpolating conditions (1), an extra $d-1$ conditions are needed, which may conveniently be assigned symmetrically, half at each end. It is well known that the resulting system is solvable, with various choices of the extra conditions. For the examples here, only *cubic* splines are considered ($d=3$), with the end conditions chosen as the 'natural' conditions

$$\mathrm{P}''(0) = 0, \qquad \mathrm{P}''(1) = 0 \tag{7}$$

For *even* degree splines, the interior knots may be chosen as the midpoints of the nodes. That is, the knots are $\xi_{-d} = \cdots = \xi_0 = t_0$,

$$\xi_i = (t_{i-1} + t_i)/2, \qquad 1 \leqslant i \leqslant n$$

and $\xi_{n+1} = \cdots = \xi_{n+d+1} = t_n$. The spline space now has dimension $n+d+1$, so that again $d/2$ conditions may be assigned symmetrically at each end (see de Boor[2], p. 76).

The cubic interpolating splines in this paper are computed using CUBSPL in de Boor[2], p. 57. (Cubic polynomials can also be obtained by the same routine, using the 'not-a-knot' end conditions instead of conditions (7).) The interpolating polynomials are computed using the Newton form, the coefficients of which are divided

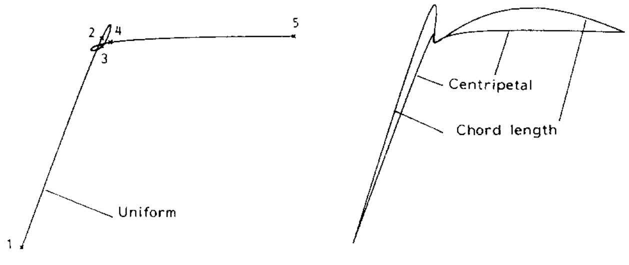

Figure 6. Natural cubic splines through 5 points: $\{x, y\} = \{(0, 0), (10, 25), (10, 24), (11, 24.5), (33, 25)\}$

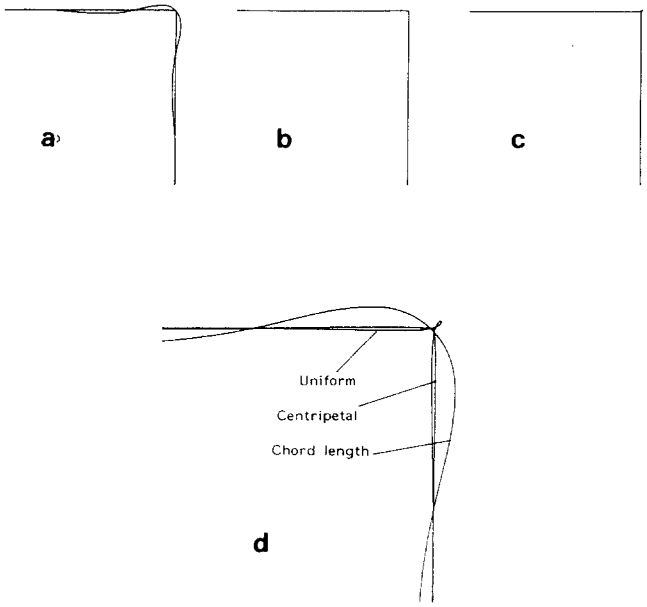

Figure 7. Natural cubic splines. Data: $\{x, y\} = \{(0, 41), (10, 41), (31, 41), (40.8, 41), (41, 40.8), (41, 31), (41, 10), (41, 0)\}$. *(a) shows the chord length parametrization and the centripetal model superimposed; (b) the centripetal model; (c) the uniform parametrization; (d) shows the different behaviours near the corner*

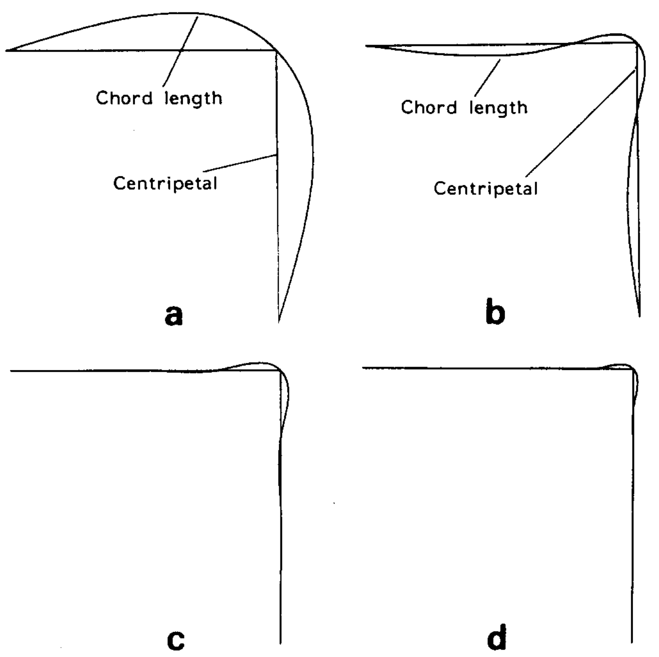

Figure 8. Natural cubic splines by chord lengths and by centripetal parametrization. In the former method, data points must be increased to decrease the undulations; while by the latter method, all cases remain about the same. The data are (a): $\{x, y\} = \{(0, 41), (40.95, 41), (41, 40.95), (41, 0)\}$*; (b): as for (a) plus (31, 41) and (41, 31); (c): as for (b) plus (10, 41), (21, 41), (41, 21) and (41, 10); (d): as for (c) plus (35, 41) and (41, 35)*

differences computed recursively (see de Boor[2], p. 24, for example). It is also a simple matter to rewrite the linear systems to solve for the Bézier or B-spline coefficients ('control points').

Some examples are now discussed, the data for which are listed in the captions of the figures. At least in the first ten figures, the exact numerical data are not important; only their relative positions have significance, since the interpolation procedures considered here, being 'global', are invariant under (that is, commute with) similarity transformations (translations, rotations, reflections, dilations) of the data points, as such transformations clearly do not affect the choice of nodes under discussion. (It is also invariant under reversal of the data sequence; however, except for the uniform parametrization, it is not invariant under different scalings of the x and y components.)

In most of these examples, a small data set has purposely been used. This is to bring out the effects of various parametrizations more emphatically, and to accommodate polynomial examples (which should not be used with high degrees). The figures confirm our intuition: where there is an abrupt change in the length of successive arms of the data polygon, overshoot tends to occur with the uniform parametrization, due to large speed over the longer arm, resulting at times in loops or cusps; while with the chord length parametrization, the curve tends to wander, over the longer arm, far from the data polygon, due to long sojourn time. The centripetal model achieves a suitable balance.

It is also seen that in the polynomial case, even the centripetal model is not entirely satisfactory. An easy modification is the *general exponent* method: $t_0 = 0$ and

$$t_i - t_{i-1} = \frac{|P_i - P_{i-1}|^e}{\sum_{j=1}^{n} |P_j - P_{j-1}|^e}, \quad 1 \leqslant i \leqslant n \tag{8}$$

which reduces to method (2) with $e = 0$, and to method (3) with $e = 1$. As expected, decreasing the value of e causes the resulting curve to tend in shape to that of the uniform parametrization, while with increasing e the curve tends to that of the chord length parametrization. Thus in Figure 1, a more satisfactory curve results with $e = 0.65$, while in Figure 3, $e = 0.37$ seems better in that between data points 1 and 2 the curve is straighter. Note that $\mathbf{t} = (t_1, \ldots, t_{n-1})$, as a function of e, $0 \leqslant e \leqslant 1$, is a curve in the simplex $\{\mathbf{t} : 0 < t_1 < \cdots < t_{n-1} < 1\}$, and it is quite conceivable that better choices of $\mathbf{t}$ may be found off this curve. For example, in place of a single e, different values e_i might be determined over different arms in assignment (8). (Such a scheme would, however, destroy the invariance with respect to dilations.) In Figure 3(c), the result of $t_1 = 0.65$, $t_2 = 0.76$ does not seem obtainable by simply varying e in assignment (8). Such an experimentation (choosing various values of t_i) is clearly laborious and not practical unless n is small. In any case, the centripetal or the general exponent method provides fairly good results and can be used as a guide to further varying the nodes if necessary. Finally, it is reassuring to note from the figures that in the *spline* case, the centripetal models are already very satisfactory, and generally we should find no need to vary the exponent.

In all examples, for each method of parametrization, the natural cubic splines are always better shaped than the polynomial curves for the same data points. This is to be expected: splines should always be preferred over single polynomials. It is somewhat unexpected that in many test cases, with small n, the splines obtained with chord length parametrization cannot even compete with the corresponding single polynomials obtained with the centripetal method. For large n, of course, single polynomials will generally be undesirable, no matter what the method of parametrization.

In Figures 7 and 8, the data points lie on a right-angled corner with the intention that a slightly rounded corner should be obtained. The chord length method leads to undulations about the data polygon that persist, though with diminishing amplitudes, as the number of data points is increased. The centripetal models give the desired shape with no perceptible undulations in all cases. Thus the present method has the clear practical advantage that generally, for an intended shape, a smaller number of data points will suffice.

This type of example has been considered by Marin (see Figure 6 in Marin[8]). Marin proposes to vary the nodes to minimize the functional

$$\int_0^1 |P''(t)|^2 dt \tag{9}$$

The 'optimal' nodes are computed numerically. However, the resulting curve still exhibits significant undulations (about half the amplitude of that of the interpolant obtained with the chord length method). This seems to indicate, as far as shape is concerned, that the minimization of the integral (9) may not be a proper criterion. What then should be a more appropriate objective function? No simple satisfactory answer seems apparent. Perhaps, in view of the discussion in the previous section, an attempt may be made to minimize the *total variation* of the centripetal force. No work has been done on this. (It should be mentioned that Marin[8] makes a distinction between 'geometric applications' and applications to robot manipulators. For the latter, minimizing integral (9) would seem natural.)

As an aside, note that it need not be surprising that minimization of integral (9) does not afford a good criterion here. The strain energy of a thin elastic beam is the integral along the curve of the square of the curvature. If the beam is given by a function $y = y(x)$, then under *small deflections*, the energy is well approximated by

$$\int (y'')^2 \, dx$$

so the consideration of integral (9) seems natural. However, a little reflection will show that, for *parametric* curves, integral (9) can no longer be considered as a good approximation of the strain energy, small deflection or not. (Consider, for instance, a straight line segment $x(t) = t(2-t)$, $y(t) = 0$, $0 \leqslant t \leqslant 1$.) More discussions on energy and related matters are contained in another article[13].

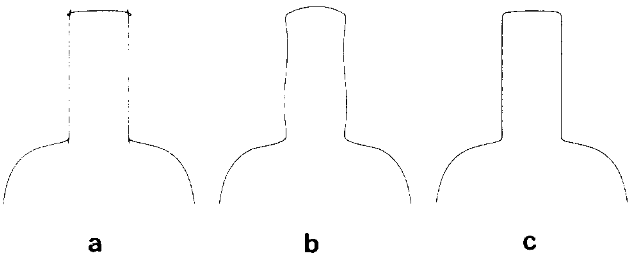

Figure 9. Natural cubic splines through 18 points: $\{x, y\} = \{(0, 0), (1.34, 5), (5, 8.66), (10, 10), (10.6, 10.4), (10.7, 12), (10.7, 28.6), (10.8, 30.2), (11.4, 30.6), (19.6, 30.6), (20.2, 30.2), (20.3, 28.6), (20.3, 12), (20.4, 10.4), (21, 10), (26, 8.66), (29.66, 5), (31, 0)\}$. Note (a) contains several loops; (b) might even be preferred on artistic grounds, but the shape is not the one intended. (a) Uniform, (b) chord length, (c) centripetal

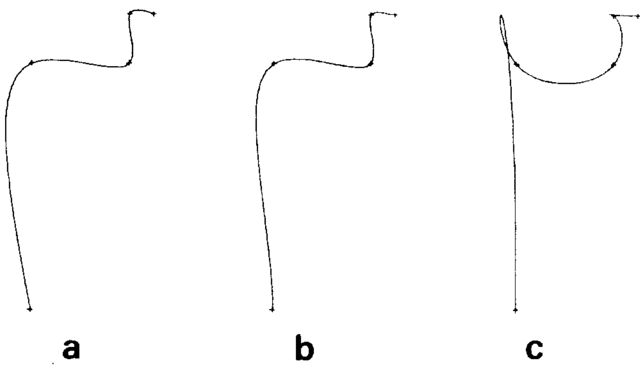

Figure 10. Effects of end conditions. The splines are cubic, with centripetal parametrization; data ponts are: $\{x, y\} = \{(0, 0), (0, 10), (4, 10), (4, 12), (5, 12)\}$. In (b) and (c), the tangent direction at (0, 0) is (0, 1), and that at (5, 12) is (1, 0). The magnitudes of the derivatives in (b) are those suggested by formula (11) in the text, whereas those in (c) are 5 times as large. (a) Natural boundary conditions

Instead of the 'natural' end conditions (7), the first derivatives may be assigned at the ends ('complete' cubic spline interpolation). Generally, however, the designer will only be able to assign the tangent directions. Thus, for instance, the first equation of conditions (7) will be replaced by

$$P'(0) = \alpha T$$

where T is a unit vector. The question arises as to what appropriate magnitude α should be assigned. Consistent with the heuristic, it may be suggested that the magnitude should be roughly the average speed over the first segment; thus,

$$\alpha = \frac{|P_1 - P_0|}{t_1} \tag{10}$$

Let $d_i = |P_i - P_{i-1}|$ and $\rho = \Sigma_1^n d_i^{1/2}$. With the centripetal method, α is

$$\alpha = d_1^{1/2}\rho \tag{11}$$

Different parametrizations of course give different expressions for equation (11). For instance, with uniform parametrization, $\alpha = nd_1$. It goes without saying that this is merely a guide; however, a size differing from formula (11) by about an order of magnitude will generally show ill effects (see Figure 10).

At times a designer may want to impose tangent directions at other data points. (Of course, the interpolation conditions (1) will be suitably amended. Note for polynomials, degrees will be raised; for cubic splines, only C^1 curves may be expected.) Suppose $P'(t_i) = \alpha_i T_i$, $0 < i < n$, where T_i is the assigned unit vector. It seems reasonable to assign to α_i the average speed over the adjacent segments $P_{i-1}P_i$ and P_iP_{i+1}. Thus

$$\alpha_i = \frac{d_i + d_{i+1}}{t_{i+1} - t_{i-1}}$$

which, with the centripetal parametrization, can be written as the average

$$\alpha_i = \frac{d_i^{1/2}}{d_i^{1/2} + d_{i+1}^{1/2}}(d_i^{1/2}\rho) + \frac{d_{i+1}^{1/2}}{d_i^{1/2} + d_{i+1}^{1/2}}(d_{i+1}^{1/2}\rho)$$

No test has been made, however, regarding this.

FURTHER REMARKS

Some obvious disclaimers should perhaps be noted. First, the emphasis is on *parametric* rather than *function* interpolation, and hence the interpolants need not preserve 'single-valuedness' (that is, if $P_i = (x_i, y_i)$, with x_i increasing, and if the interpolant is $(x(t), y(t))$, $y(t)$ need not be a *function* of $x(t)$). In fact, the only rotation-invariant interpolation method which preserves 'single-valuedness' must result in the data polygon as the interpolant; see Brodlie[3], p. 34.

Second, any definite qualitative sense of 'shape preservation' is not claimed here other than that the interpolant conforms well to the data polygon. In recent years there have appeared a number of sophisticated 'shape preserving' *function* interpolation schemes. By this is meant either monotonicity preserving or convexity preserving, or both. Generally they obtain estimates of the derivatives at the data points and use them in the construction of the interpolants. As such, they belong to what is called 'local' methods, in contrast to the 'global' methods considered here.

Still, it may not be entirely out of place to compare a few of the test results obtained with some of the function interpolation methods with the results obtained by simple node choices. Figure 11 shows results for the Akima data tested in Fritsch and Carlson[11]. The centripetal model seems quite passable, but the result with e = 0.25 appears as good as that obtained by the monotonicity preserving method of Fritsch and Carlson[11]. Note the chord length result in Figure 11(b) is not 'single-valued'. Nor is it in Figure 12(b), where the data is from Brodlie[3]. In Figure 12(c), between data points 4 and 5, the curve clearly fails to be monotone; not much improvement can be obtained by varying e. However, the centripetal model still compares quite well with some earlier local methods given in Brodlie[3] (e.g. the 'osculatory method' and the method of Ellis–McLain; a result by J. Butland in the same survey is, however, much better).

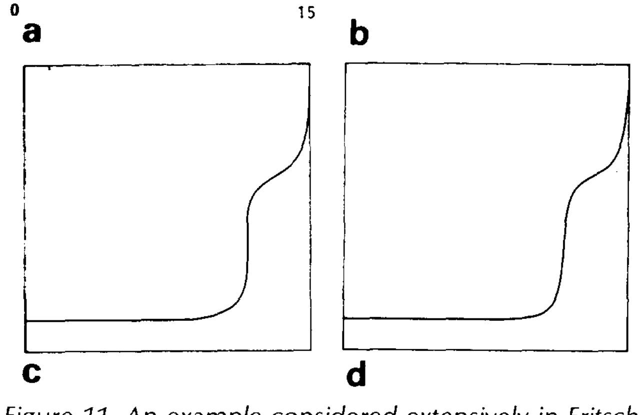

Figure 11. An example considered extensively in Fritsch and Carlson[11]*: {x, y} = {(0, 10), (2, 10), (3, 10), (5, 10), (6, 10), (8, 10), (9, 10.5), (11, 15), (12, 50), (14, 60), (15, 85)}. This figure shows results obtained by parametric interpolation with natural splines, under various node choices. Display scales are different in* x *and in* y. *(a) Uniform, (b) chord length, (c) centripetal, (d)* e *= 0.25*

A common criticism often levelled against spline interpolations is the tendency to produce spurious inflections. Often this is due to the insistence upon *function* interpolation, where the nodes are dictated as the abscissas of the data points. In Figure 13, the results for a data set used in Brodlie[3] are shown, where all three node choices produce equally good curves.

Hartley and Judd[6] suggested a different method of spline interpolation. The *knots* are chosen in a special way; the *nodes* are then taken as the Schoenberg averages of these knots. No extra end conditions are required. However, in the very special case where the number of data points equals the order of the spline, it is easily seen that the method actually reduces to interpolation by a *single* polynomial with *uniform* parametrization.

After completion of this paper, the author was informed that the parametrization by the square root of the chord lengths has already been touched upon in Hosaka and Kimura[12] with, however, no attempt at a justification. Specifically, Section 3 of that paper deals with interpolation by parametric cubic splines, couched in terms of Bézier polynomials. If $P_i(t)$, $0 \leqslant i \leqslant n-1$, are cubic polynomials with $P_i(0) = P_i$, $P_i(1) = P_{i+1}$, the assignment of the factors k_i, $0 \leqslant i \leqslant n-2$, in

$$P_i'(1) = k_i P_{i+1}'(0)$$

is equivalent to the assignment of the nodes t_i, $1 \leqslant i \leqslant n-1$. (Thus the chord length parametrization corresponds to $k_i = |P_{i+1} - P_i| / |P_{i+2} - P_{i+1}|$, etc.) Part of that section seems devoted to a justification of the *chord length* method, since they prove that under this assignment, two different approaches lead to the same set of equations for the spline coefficients. However, in a brief remark (with an example), it is stated that 'frequently', the square root method 'gives more

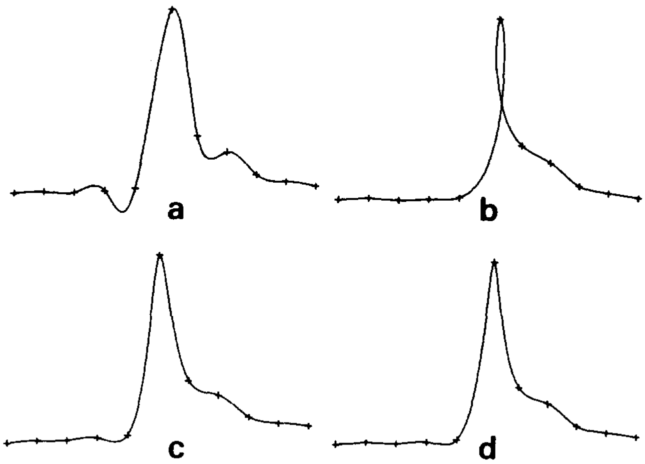

Figure 12. Parametric interpolation with natural cubic splines, for a case considered in Brodlie[3]*: {x, y} = {(0, 1), (1, 1.1), (2, 1.1), (3, 1.2), (4, 1.3), (5, 7.2), (6, 3.1), (7, 2.6), (8, 1.9), (9, 1.7), (10, 1.6)}. (a) Uniform, (b) chord length, (c) centripetal, (d)* e *= 0.6*

natural curves' than does the chord length method. (These comments also appear in some of their later publications.)

In conclusion, engineers in industry often complain about control points not being on the curve; of course they are not. Bézier and B-spline curves are particular representations of polynomial and piecewise polynomial curves, the control points being the coefficients in these representations. These representations are particularly nice due to their numerical stability, and the control points reflect significantly the geometry of the curve. However, if a curve is required to satisfy certain discrete data (positions, tangent directions, etc.), moving the control points about is hardly a feasible way to achieve this. (It is the overemphasis, in a number of publications, of this type of tweaking or 'control', that is the cause of the complaint.) To meet such discrete constraints, interpolation is required, thereby computing the requisite control points. For parametric curves, this requires the specification of a set of interpolation nodes out of infinitely many possible choices; and it is this problem that is of concern here.

ACKNOWLEDGEMENT

The author wishes to thank Gunther Schuller for his interest in this work and his help in facilitating the graphics displays. The name 'centripetal model' is also suggested by him. The author is also indebted to George Allen who drew his attention to the work of Hosaka and Kimura, during the SIAM meeting in Albany, 1987, where this paper was presented. Comments by Les Piegl also result in some improvement of presentation.

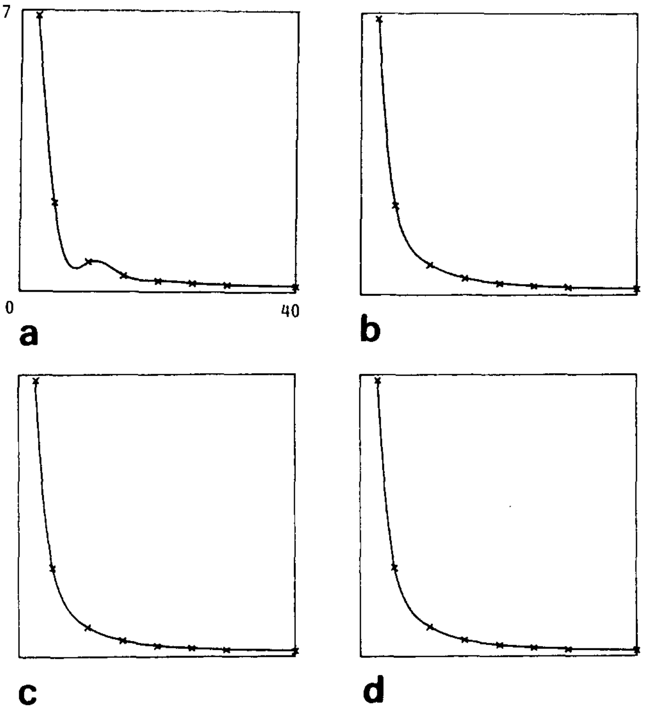

Figure 13. Natural cubic splines for a case considered in Brodlie[3]: $\{x, y\} = \{(2.5, 6.875), (5, 2.230), (10, 0.751), (15, 0.416), (20, 0.283), (25, 0.219), (30, 0.182), (40, 0.143)\}$. Display scales in x and y are different. (a) shows the result of function interpolation, i.e. the nodes are the abscissas (rescaled) of data points; (b) uniform; (c) chord length; and (d) centripetal.

REFERENCES

1 **Ahlberg, J H, Nilson, E N and Walsh, J L** *The theory of splines and their applications* Academic Press, New York, NY, USA (1967)

2 **de Boor, C** *A practical guide to splines* Springer-Verlag, Berlin, FRG (1978)

3 **Brodlie, K W** 'A review of methods for curve and function drawing' in **Brodlie, K W (ed.)** *Mathematical methods in computer graphics and design* Academic Press, London, UK (1980) pp 1–37

4 **Späth, H** *Spline algorithms for curves and surfaces* Utilitas Mathematica, Winnipeg, Canada (1974)

5 **Foley, T A** 'Interpolation with interval and point tension controls using cubic weighted ν-splines' *ACM Trans. Math. Softw.* Vol 13 (1987) pp 68–96

6 **Hartley, P J and Judd, C J** 'Parametrization and shape of B-spline curves for CAD' *Comput.-Aided Des.* Vol 12 No 5 (1980) pp 235–238

7 **Topfer, H-J** 'Models for smooth curve fitting' in *Conf. Proc.: Numerical methods of approximation theory* Oberwalfach, FRG (1981) pp 209–224

8 **Marin, S P** 'An approach to data parametrization in parametric cubic spline interpolation problems' *J. Approx. Theory* Vol 41 (1984) pp 64–86

9 **Fritsch, F** 'Energy comparisons of Wilson–Fowler splines with other interpolating splines' in **Farin, G E (ed.)** *Geometric modeling, algorithms and new trends* SIAM, Philadelphia, PA, USA (1987) pp 185–201

10 **Epstein, M P** 'On the influence of parametrization in parametric interpolation' *SIAM J. Numer. Anal.* Vol 13 No 2 (1976) pp 261–268

11 **Fritsch, F N and Carlson, R E** 'Monotone piecewise cubic interpolation' *SIAM J. Numer. Anal.* Vol 17 No 2 (1980) pp 238–246

12 **Hosaka, M and Kimura, K** 'Synthesis methods of curves and surfaces in interactive CAD' *IEEE Conf. Proc.: Interactive techniques in computer aided design* (Bologna, Italy) IEEE, New York, NY, USA (1978)

13 **Lee, E T Y** 'Energy, fairness, and a counterexample' *Comput.-Aided Des.* (to appear)