USE OF A SINCLAIR SPECTRUM FOR SHAPE ANALYSIS

Susan Laflin

University of Birmingham

Abstract:

Last year I discussed the problems of input of profiles from the various digitising tablets available for the Spectrum. This year I shall describe the curve-fitting software applied to these profiles, the short-cuts necessary because of the slow speed of the BASIC interpreter and the problems of interfacing hardware or data files to the Pascal compiler, and the resulting restrictions of this software when applied to classification methods.

Introduction:

At the Computer Applications in Archaeology conference in 1985, I discussed the problems of input of profiles to the Spectrum. The first program (TRACE 1) of my set will allow input from the RD-Tracer and after suitable checking will allow the user to store the data in a file on microdrive. My tracer is old, has travelled widely and been used by many different people and I now find it is not very accurate. The same software would also work on newer models which are more accurate, although I don't know how long they would last. Users with a different configuration including one of the other digitising tablets or possibly a disc drive instead of microdrives would need to adapt this program.

The program TRACE 3 allows either digitised or fitted curves to be plotted on either the screen or an MCP40 plotter.

B-Spline Curves

The second program in the set (TRACE 2) gives a useful demonstration of interactive fitting of a B-spline curve to the set of digitised points. Since this is a technique widely used in Computer-Aided Design, a brief description of the method is appropriate at this stage. I shall restrict my discussion to the third order curve with n+1 control points, and to curves in two-dimensions although the same calculations can be carried out for x, y and z co-ordinates to define a curve in three-dimensional space. The values of x and y along the curve are calculated in terms of a parameter t which varies from 0 to tmax where tmax = n-k+2 = n-1 when the order k = 3. So a curve with 20 control points will have values of t varying from 0 to 18.

The method of calculation involves the use of a "knot-vector" which relates the values of t to particular control points and ensures that each value of x(t) and y(t) along the curve is calculated from the nearest k control points. To ensure that the curve passes through the first and last control points, the values of t = o and t = tmax must be repeated in the knot vector k times. So for 5 control points (n=4) we have tmax = 3 and the knot vector contains the values (0, 0, 0, 1, 2, 3, 3, 3). The values of the knot vector are referred to as v(i) where the index i varies from 0 to n+k or in this case from 0 to 7. The value of v(3) is 1 in this case.

For any given value of t, we calculate the values of the array N_{ik} (t) from the following recurrance relation.

$$N_{i1}(t) = 1 \text{ if } v(i) < t < v(i+1)$$
$$= 0 \text{ for all other values of i.}$$

Then to calculate succeeding columns of the array

$$N_{ij}(t) = \frac{(t-v(i))}{(v(i+j-1)-v(i))} N_{ij-1}(t) + \frac{(v(i+j)-t)}{(v(i+j)-v(i+1))} N_{i+1\ j-1}(t)$$

Finally when we have calculated all k columns of N, we evaluate the co-ordinates of the point P(t) from the expression

$$P(t) = \sum_{i=o}^{n} N_{ik}(t)Pi$$

The control points Pi are numbered P0, P1, P2,.....,Pn and this expression is evaluated for the x-coordinates and for the y-coordinates. For any given value of t, only three of the values of N_{i3} (t) will be non-zero and so the nearest three control points will give the values of any x(t) and y(t).

With the above example, let us look at the value of x when t = 0.4.

knot vector

v(i)	i	$N_{i1}(0.4)$	$N_{i2}(0.4)$	$N_{i3}(0.4)$
0	0	0	0	0.36
0	1	0	0.6	0.56
0	2	1	0.4	0.08
1	3	0	0	0
2	4	0	0	0
3	5	0	0	
3	6	0		
3	7			

So x(0.4) = 0.36.xo + 0.56.x1 + 0.08.x2 + 0.x3 + 0.x4 and we have a similar expression for y(0.4).

To plot the B-spline curve, we need to evaluate x(t) for values of t from t = o to t = tmax and plot these out. The program TRACE 2 plots these in steps of 0.2 and user then has the chance of moving one or more control points and re-plotting the curve until the two profiles lie on top of each other. This gives a useful demonstration that it is possible to find a B-spline which coincides with the digitised data but is very slow. The next requirement is an automatic method of calculating the required control points.

Error Estimation

In order to decide when we have control points which give a good fit, we have to have some means of estimating the distance between the digitised points and the B-spline curve.

The values x(t) and y(t) of points on the curve are calculated in terms of the parameter t which takes values from 0 to 18 as we move along the B-spline curve of order 3 with 20 control points. To compare these with the digitised points x(j) and y(j), we need some means of assigning a value of t(j) to each digitised point. Then the control points can be moved in both x and y directions until the expression

$$E = \sum_{j=1}^{N} e(j) \quad \text{is as small as possible}$$

$$\text{where } e(j) = \left| x(j) - x(t(j)) \right| + \left| y(t) - y(t(j)) \right|$$

In this study, I have used two methods of calculating t(j). The first one uses the numbering of the points and scales these integers so that the range 1 to N corresponds to the range 0 to 18. i.e. $t(j) = (t-1) + 18/N$

The second one uses the idea of arc length along the digitised curve. Assume the points have been recorded so that the curve is approximated by linear segments joining successive points along the curve. The length of the line joining (x(j-1), (y(j-1)) to x(j), y(j)) is given by

$$d(j) = \sqrt{(x(j) - x(j-1))^2 + (y(j) - y(j-1))^2}$$

and the arc length measured along the curve as far as point n is

$$S(j) = \sum_{j=1}^{n} d(j)$$

Hence smax, the total length of the curve is

$$smax = \sum_{j=1}^{N} d(j)$$

and $t(j) = d(j) .\ 18/smax$

thus scaling this arc length to the range 0 to 18.

These two choices of t(j) lead to different types of fitting in the curve. In the first case, the spacing of the digitised points along the curve will influence the values of t and hence the grouping of the control points along the curve. If the digitised points are close together, then this area of the curve will account for a larger variation in t and hence will have the control points grouped more closely. It allows certain areas to have very much more detail than others. However, it does make it easy for the same curve digitised by two different people to yeild different sets of control points.

The second method ensures that the control points are evenly spaced along the curve. This ensures that the resulting set of control points is unique and independant of the person digitising the curve and any variations in the spacing of the points. It will not give a good fit if one small section of the curve is very much more complex than the rest e.g. the rim of a complete profile of a pot.

Curve Fitting Method

In deciding to deduce an automatic method of fitting the B-spline, I have based my approach on the minimax method. For this method, I examine the whole error curve and then adjust the positions of the control points to reduce the maximum errors of the curve. This will of course increase the errors at other positions within the curve, and the control points will have to be moved back to reduce these errors. It may not be possible to achieve zero error with a given number of control points, but it will be possible to find a "best fit" which balances out these errors and gives the smallest possible value of maximum error.

In applying this method on the Spectrum, I found my main problem was speed of calculation. When using the Basic interpreter as supplied with the Spectrum, large number-crunching jobs are very slow. This problem would have been much less acute had I been able to use the Hisoft Pascal Compiler with the input from the RD-Tracer, but I still haven't solved that interfacing problem. Consequently I had to cut corners on my curve fitting method and the resulting programs (TRACE 4a and TRACE 4b) will usually converge to a solution, but will not give the strict minimax best fit and may not be unique for some sets of data. Figure 1 shows the data on which they were compared and figure 2 gives an indication of the error curves from the final fit in each case.

The methods TRACE 4a and TRACE 4b differ only in the method used to calculate t(j) - for TRACE 4a it is calculated from j and for TRACE 4b it is calculated from the arc length s(j).

For these methods, we note that the factor multiplying Pi is $N_{3i}(t)$ and by observing the variation of $N_{3i}(t)$ with respect to t, we observe that Pi has maximum effect at t = 0.6, for i = 2,3,......17 Pi has maximum effect at t = i - 1.5 and P18 has maximum effect at t = 17.4. Accordingly, for each control point I find the value of j for which t(j) is closest to this maximum value of t. We might have the case that for j = 5, t(j) = 0.58 which is closest to the value 0.6 - then P1 is moved until the calculated value x(0.58), y(0.58) are as close as possible to the digitised points x(5), y(5). By only evaluating the array $N_{3i}(0.58)$ for all the movement of the control point P1, I reduce the amount of calculation and hence the time taken. It takes several iterations, first varying the x-coordinate of P1, then the y-coordinate and then the x-coordinate again and so on until the minimum error is found. This means that repeating this process for each of the values of Pi from i=1 to i=18 still takes about 20 minutes on the Spectrum.

This is not a minimax fit and the results for TRACE 4b for this particular profile show large errors in the centre of the range which corresponds to the detail of the run and indicates that equally spaced control points do not give enough variation in this area. This would probably be more successful when applied to other shapes such as profiles of axes.

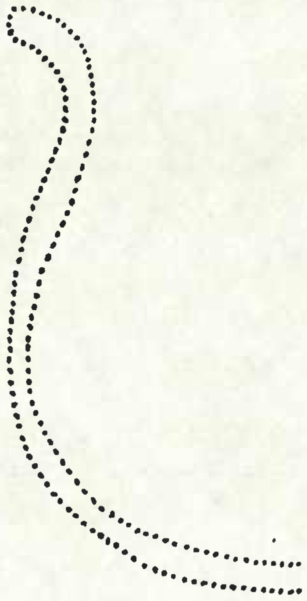
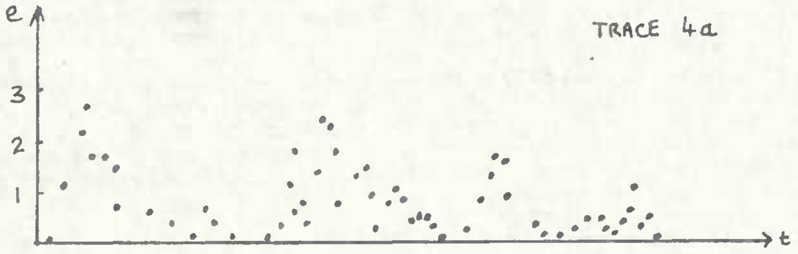


Figure 1



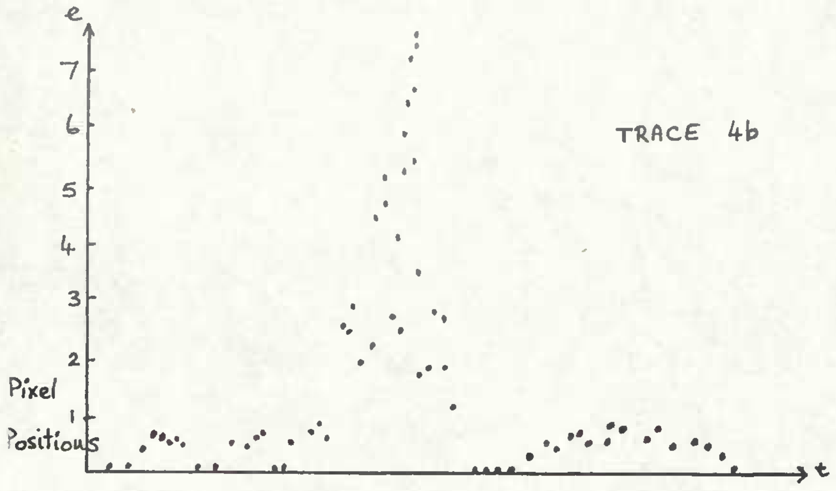


Figure 2

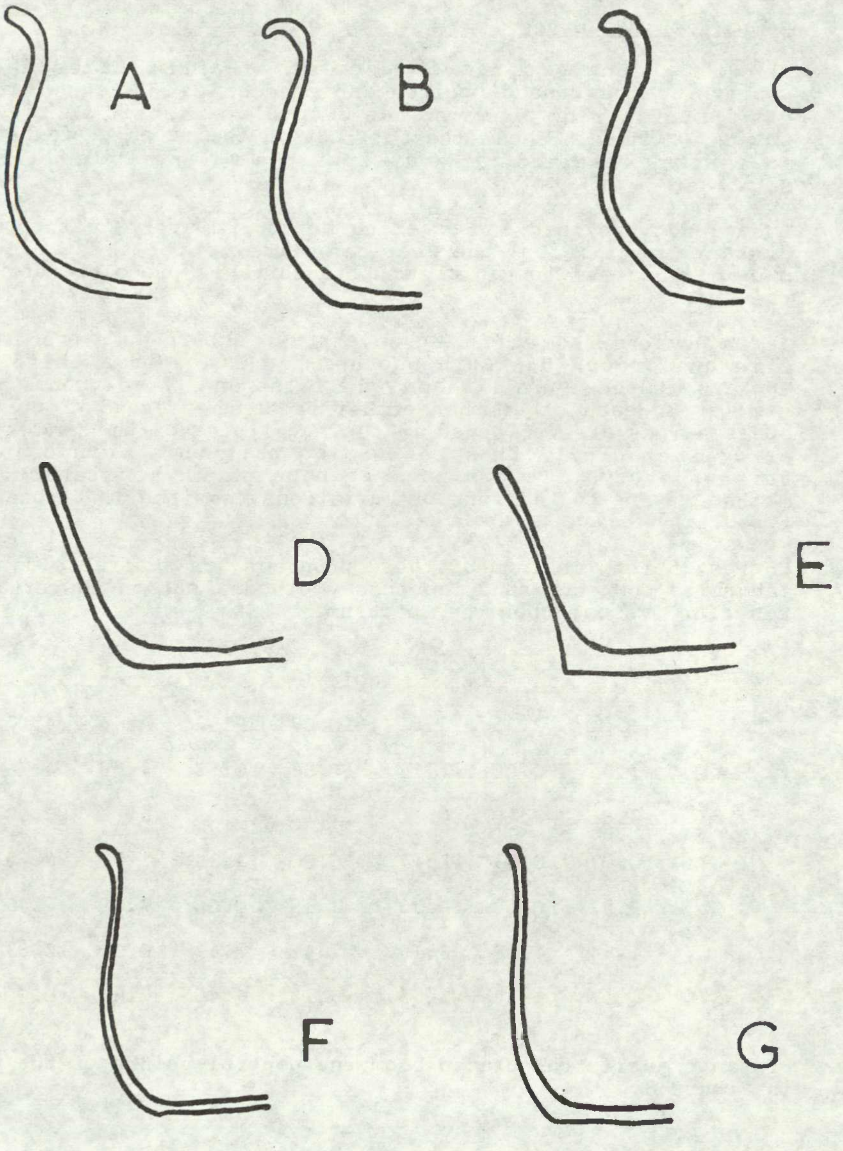


Figure 3.

Comparison of Curves

The remaining program in the set, TRACE6, takes the files containing the control points and produces a distance matrix for these profiles by applying the City-Block Metric to the control points. Figure 3 shows the outlines of the seven profiles used as an initial test and table 1 shows the values obtained for this data.

I have not provided a clustering method to apply to the resulting distance matrix, but the user can either enter these values to a package such as Clustan or write a suitable piece of software for the Spectrum.

It is obvious from the diagram that we should have profiles A,B & C in one group, D & E in another and F & G in a third. Also looking at the overall shape, the first and third groups are more similar to each other than either is to the second group. These observations are reflected in the magnitudes of the numbers in the array in table 1, thus confirming that this method justifies further study. It has not yet been shown how sensitive this method may be to the sort of variations required in archaeological typography.

Copies of the listings of these programs are available on request. I hope to continue studying these methods, but on faster and more powerful systems than the Spectrum.

Table 1

0.00	2.65	3.01	10.07	9.52	4.64	4.17
2.65	0.00	2.10	10.93	10.12	4.78	4.78
3.01	2.10	0.00	11.14	10.33	6.16	5.68
10.07	10.93	11.14	0.00	3.39	7.45	6.78
9.52	10.12	10.33	3.39	0.00	7.49	6.45
4.64	4.78	6.16	7.45	7.49	0.00	2.83
4.17	4.78	5.68	6.78	6.45	2.83	0.00

Distance matrix calculated from the control-points of the profiles in Figure 3.