The Xerox Alto Font Design System

Patrick Baudelaire

Guest Editor's Note

This article is from a talk given at Stanford University in 1983 at a seminar for the Association Typographique Internationale (ATypI). It describes pioneering digital font software developed at the Xerox Palo Alto Research Center in 1974. Built for prototype personal workstations, the software uses mathematical curves called "splines" to define the outlines of letter shapes that are converted to bitmaps (pixel mosaics) for use on computer screens and digital printers. This spline-bitmap model is used today for the screens of nearly all computers, smart phones, ebooks, and other text displays. Previously unpublished, the manuscript appears here as digital font archaeology – a glimpse of concepts from four decades ago that became the technology of much that we read today. We are grateful to Patrick Baudelaire for permission to publish it as he wrote it in 1985 and to the Cary Graphic Arts Collection of Rochester Institute of Technology for providing scans of the original manuscript and images in its collection.

Charles Bigelow

Keywords

font design, typography, letterform, software design, splines, bitmaps

Introduction

The Xerox Alto font design system was designed and built around 1974, at Xerox PARC's Computer Science Laboratory, initially as an experimental tool for helping in the production of digital typefonts for a growing number of prototype xerographic laser printers and display devices with very diverse resolution characteristics. At that time, it would have been presumptuous to imagine that the same system, only slightly adapted, would be in regular use, some ten years later, as a font digitization system for the Xerox printer product line.

Despite its daily usefulness, the system today [1985] shows sure signs of obsolescence in its performance and many of its implementation details, in both software and hardware. By current standards, however, it still performs quite well, and this endurance is a tribute to the quality of the overall design by Robert Sproull and to the remarkably advanced concepts incorporated in the Alto computer.

I will not attempt in this paper to present a comprehensive and detailed description of the operation of the system, which is fully described in a user's manual [1]. I prefer to give simply an overview of the system, trying to bring a critical eye on the design and the implementation of the software, a point of view now facilitated by time and distance. Before going further, it is important to stress that the system discussed here is an internal system, not a Xerox product, and that all the opinions expressed here are strictly those of the author and only engage himself.

General principles and overall organization

In the Alto system, as in several other systems [2], font production is basically a two-step process, resting on the geometrical representation of a character in outline form.

The process is illustrated in Figure 1. It is implemented by two main computer programs. FRED is an interactive graphics program used for defining the geometric outline of characters as mathematical spline curves. PREPRESS is used for creating automatically the appropriate digitized character in the form of a matrix of dots, for a given point size and a given printer (a process called scan-conversion). It is also used for touching up by hand the resulting dot matrix, also called a bitmap. (The DRAW program showed in Figure 1 is marginal to the process.)

The whole software was programmed for the Alto experimental workstation, a strikingly original personal computer, designed and built at Xerox PARC [3]. Its general size, performance and configuration (including a high resolution graphics display) made it well suited for interactive graphical interactions and friendly user interfaces relying on visual feedback. It has been the forerunner in a well-followed design trend of high-powered computers which has led to such recent products as the Apollo and Sun computers, the Xerox Star, Apple Lisa office workstations, and the Apple Macintosh personal computer.

With the exception of a television camera for scanning type artwork and its associated computer interface. the equipment configuration (portrayed in *Figure 2*) is the standard desk size Alto station in its more recent extended memory version (Alto II), featuring:

standard keyboard;

the now popular table top pointing device called the mouse ;

one or two disk drives with removable cartridges:

a connection to the Ethernet local network. allowing direct communication with printers and file storage servers:

a black and white graphics display. with a resolution of 600 by 800 dot.

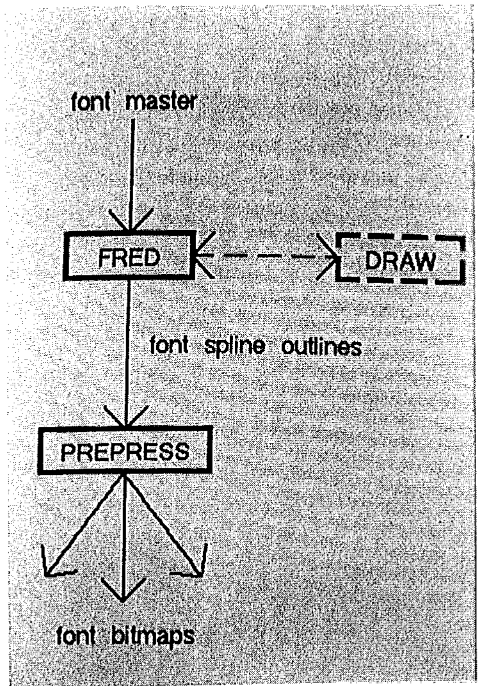

Figure 1.

The two-step process for producing digital typefonts.

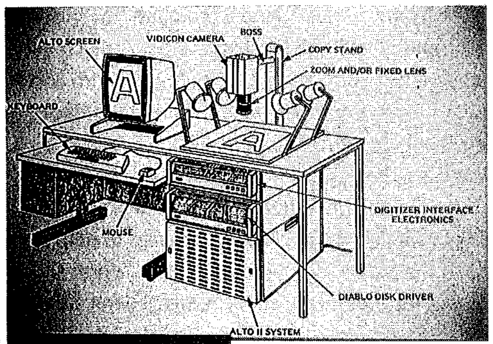

Figure 2.

The Alto workstation equipped with a television camera.

Overview of FRED

FRED is primarily an interactive graphics editor for creating and modifying 'images' composed of straight lines and spline curves. In fact, its graphics editing functions are quite general and are not specialized for handling letterform outlines. They include commands for defining, deleting, moving, copying, transforming by symmetry, and redefining lines, curves and pieces of curves. These operations are activated by selection from the main menu and are executed interactively by pointing with the mouse in the drawing area (*Figure 3*).

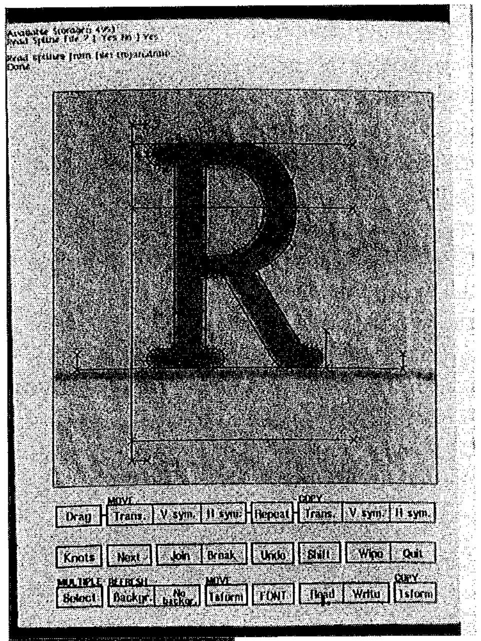

Figure 3.

FRED main menu for constructing a spline outline on a background television image of a character. interpolating.

In addition, there exist a small number of functions that are more specific to the font design application. First, a television image can be displayed inside the drawing area, the line and curve image being superimposed on this background *Figure 3*). This is used to facilitate replication of character shape by drawing directly on top of a suitably enlarged image of the letterform artwork. This is an alternate solution to tracing the artwork with a stylus on a digitizing tablet. One advantage of the television image method is the constant visual feedback it provides throughout the graphics editing process. On the other hand, the set up requires careful optical and electronic adjustments (the Alto system has provisions for positioning fiducial marks to insure proper relative scaling of all characters shapes within the same set).

There is also a set of commands for specifying various typographical attributes of the character shape: its 'bounding box' (defining width and baseline), its 'code' (such as: R), and its typeface (for instance : Times Roman italic).

Finally, there are the expected filing functions for saving away or retrieving character shapes. The recommended practice is to group characters of the same typeface in a single or a small number of files to help the bookkeeping for an inescapably fast growing database of fonts, a non-satisfying solution to an often underestimated problem. In this general area of database handling (somewhat side-stepped, since we relied almost entirely on the effective but very simple filing functions of the Alto) one early design decision turned out to be very wise: data files of character shapes, created or manipulated with FRED, are recorded as text files (rather than binary data files) and can thus be printed as well as read and corrected with any text editor. Although the exercise of reading or text editing such a file is certainly tedious, this form made it possible on a number of occasions to recover from difficult situations

The graphics editing functions were designed to be unconstrained by the nature of the application. They deal with geometrical 'objects' such as lines and curves, in a fairly general fashion, without any requirements that they form, at a given moment, a 'meaningful' contour made of connected lines and curves. The same commands are used to create or modify lines and curves: a line is defined by pointing at its two end points: a spline curve is drawn by pointing at a number of control points that define the path followed by the curve (in fact. a line is the simple case of a 'curve' defined by only two points). This somewhat loose handling of the graphical elements making up character outlines offers a degree of generality in the construction process which has some advantages. It allows, for instance, the use of pieces of a character to build other characters (e.g., serifs or strokes), or the utilization of graphical elements as guiding tools or 'scaffolding' in the construction process (marking up the baseline, x-height. cap height. descender height, em width of a typeface, as shown in *Figure 3*).

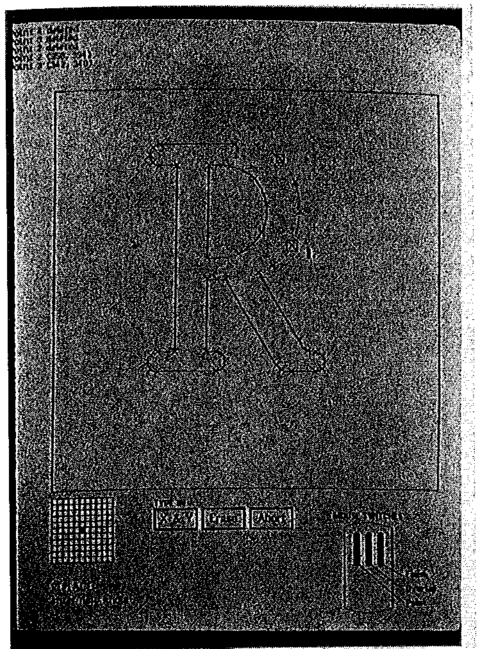

Figure 4.

FRED spline editing menu showing the modification of a character outline.

Another attempt at generality turned out to be, retrospectively, a poor choice. Deleting or modifying a line or curve (by removing, adding or changing control points) is implemented as a single unique command, by which a contiguous set of control points on a curve (including all the cases from one single point to all the points of the curve) is replaced by an arbitrary set of new control points (possibly empty). Although one can easily convince oneself that this method actually provides for all deletion or modification operations, the principle remains somewhat abstract and its application is not always the most efficient.

For instance, it is probably more natural to provide an elementary operation by which a control point is simply 'grabbed' with the mouse and moved to a new position. Similarly, adding a new control point to a curve would be done simply by pointing at the position of this new point, under the assumption that a new point is likely to be in the vicinity of the curve (and not at some arbitrary distance) and therefore that the insertion of this point at the proper place in the sequence of control points is not ambiguous.

FRED does not provide any scaling operation, nor any means of slanting, expanding or transforming geometric shapes in some automated fashion, which was an oversight. Some of these functions are possible with PREPRESS, as part of the scan conversion process (see below), but they would sometimes be useful at the outline design stage. Another graphics editing program, DRAW [5], built primarily for preparing graphical illustrations and providing some geometric transformations, is sometimes used for this purpose in conjunction with FRED (*Figure 1* and *Figure 5*). This can be qualified, at best, as an afterthought.

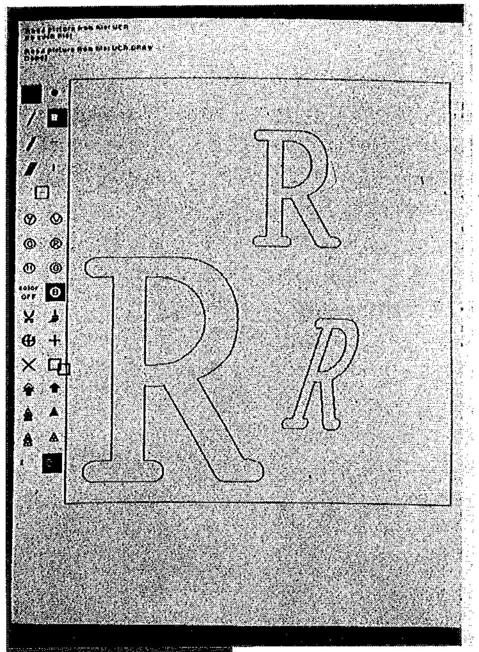

Figure 5.

The DRAW graphics editor used occasionally for geometrical shape transformation.

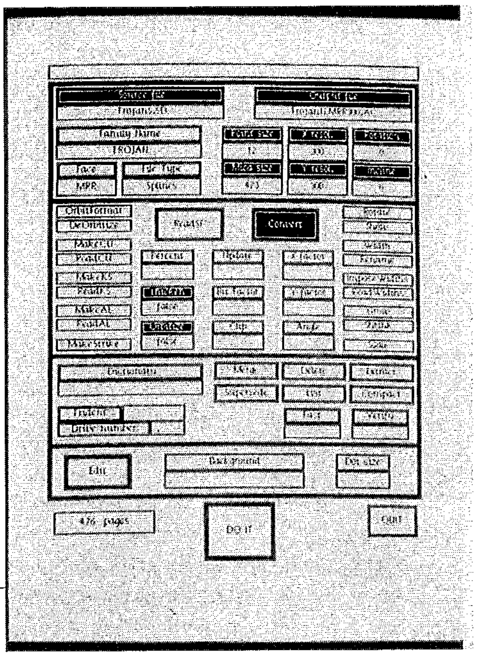

Figure 6.

PREPRESS main menu. set-up for a convert operation to produce a 12 point font bitmap for a 300 dot per inch printer.

Overview of PREPRESS

The second step of the font production process is to generate digitized characters from the spline outlines. This is done by PREPRESS. The program provides two major functions: scan-converting outlines to bitmap and editing bitmaps. It also has a number of bookkeeping functions which are necessary for handling the font database. All the operations are invoked through menu interaction (*Figure 6*). At first glance. the menu appears imposing, but it is actually quite simple to use.

The production of bitmap fonts is an automated process: this is the convert operation (*Figure 6*). It requires, to be fully specified, three parameters. The first two are the point size of the font and the resolution of the display or printing device for which this font is being generated. This determines the actual size of the character bitmap. For more generality, the resolution in the two directions of the matrix can be different. A third parameter, the rotation factor, defines the relationship between the scanning direction of the device and the writing direction for the font, which determines the particular ordering of dots within the bitmap. It allows the creation of digitized fonts that run horizontally, vertically or at any angle on the display or printed page.

Some geometric transformations are possible during scan-conversion. Slanted characters (poor man's italics) are produced with the incline option. Pseudo-expanded or condensed characters are obtained by application of a scaling factor in the horizontal direction.

PREPRESS also provides a bitmap editor for creating or modifying a bitmap through direct screen interaction. This is used generally for touching up scan-converted characters. Experience showed that, given the straightforward conversion algorithm used in PREPRESS, character bitmaps need to be hand edited if their height is less than 40 to 50 dots (a 10 or 12 point font on a 300 dot per inch printer).

For instance, Figure 7 shows the uncorrected scan-conversion results for a 12 point serif font (Trojan, a typewriter font from a Diablo printer daisy wheel). It exhibits classical scan-conversion artifacts particularly visible on the serifs. To help in the dot editing process, another bitmap character can be displayed as a background. This allows the user to compare different versions of the character (*Figure 8*). More interestingly, one can use a bitmap of much higher resolution (thus free of scan-conversion artifacts) as a guiding pattern (*Figure 9*).

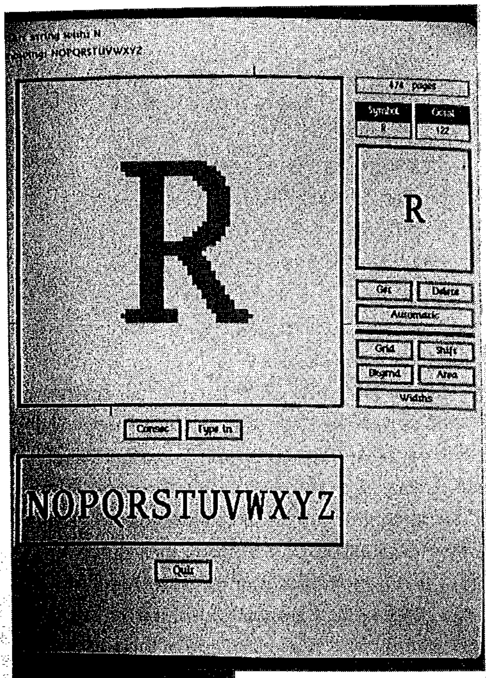

Figure 7.

The PREPRESS bitmap editor showing the uncorrected result of the bitmap conversion (12 point, 300 dot per inch).

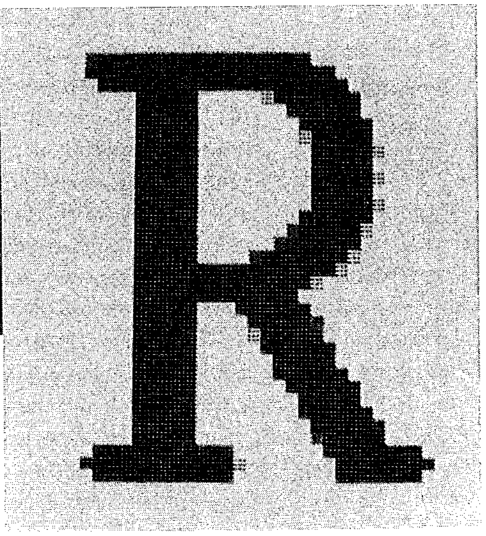

Figure 8.

Superimposed views of the corrected and uncorrected bitmaps: medium gray dots have been removed, light gray dots have been added, dark gray dots are untouched.

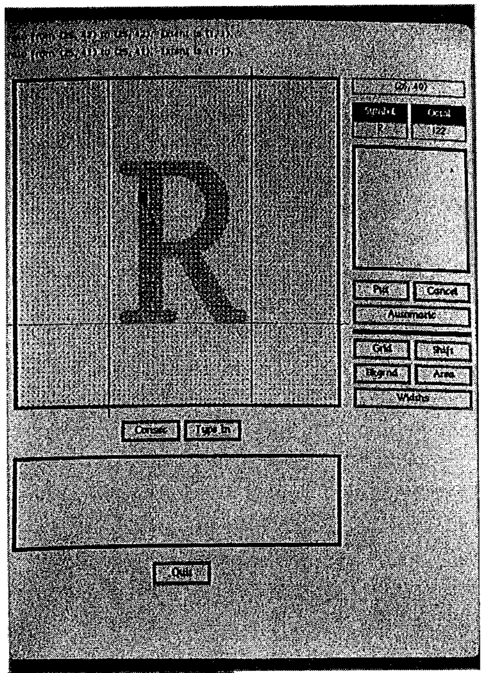

Figure 9.

Using the PREPRESS editor to create a low-resolution bitmap character, with a higher resolution bitmap as a background.

Finally, PREPRESS is used for the rather complex bookkeeping task involved in managing font bitmap databases containing numerous typefaces in many sizes for a variety of printing machines and display devices, as well as the corresponding character width tables that must be used by document formatters for proper typesetting. As is often the case with computer software, the whole bookkeeping chore is amplified by required compatibility with past devices and programs. As a result, PREPRESS is equipped to handle on the order of a dozen different file formats to represent all sorts of font related information, most of which is unfortunately not hidden from the font system's user.

Performance improvements

The Alto font design system was probably one of the earliest fully operational system using spline outlines. It benefited from the nicely integrated (and at that time, unusual) design of the Alto computer, allowing a direct and fine-tuned control of the displayed image from within the application program FRED and PREPRESS. The resulting manmachine interface is, by average standards, reasonably friendly without being very innovative. It relies heavily on direct pointing with the mouse, menus, visual feedback and graphical display of information. However, the overall interface design is still rather conventional: this is one of the areas where the age of the software shows up. It is clear that the user interface could be greatly improved and simplified by following more modern design concepts such as demonstrated in systems such as the Xerox Star and Apple Lisa.

But the general history of this software, with its unforeseen transition from laboratory to production site, did not allow for a cost effective redesign. Moreover, the initial memory limitation on the Alto of 64

kilobytes (for programs, data and the 600 x 800 displayed image) placed some severe restrictions on the design that were not significantly alleviated in the following extended memory versions of the machine, barring a significant reimplementation. As a comparison, current graphical workstations routinely have from 8 to 16 times more memory (512 to 1024 kilobytes).

Modern machines also have finer and faster graphics. Displays of 800 x 1200 dots are now common. The character display area used by FRED is only 500 x 500, without zooming capabilities, the remaining areas of the screen being used for menus. This limits the resolution of the original character outline from which digitized fonts are produced to 500 coordinate units. As pointed out by Charles Bigelow [4,6], this turns out to be below the usual resolution for high quality artwork. On larger modern displays, one could now easily achieve a resolution of 1000 coordinate units, sufficient for representing font masters with accuracy.

By and large the Alto provides the right architecture for this type of application. Its integrated raster-type graphics screen enables the display of an appropriate variety of images: medium quality text, line and curve drawings, and digitized images in bitmap form. Performance improvements would follow naturally from current progress in hardware design.

A word about splines

Lest the non-mathematician reader be left with the somewhat optimistic impression that fitting type outlines by mathematical spline curves is a fully mastered concept, I wish to conclude this overview by a few general remarks about the mathematics of spline fitting.

A spline curve is actually made of pieces of curves that are connected end to end to construct a seemingly smooth single curve. In fact, it is this construction scheme which allows spline curves to approximate arbitrary shapes quite well. However the scheme is not exempt from constraints. In particular, the quality of approximation by splines can be expressed in terms of three desirable properties.

(a) Curvature continuity.

The smoothness of a spline curve is achieved by the mathematical requirement that the curve pieces that compose the spline not only be end to end connected, but also have the same slope at their junctions. This minimum property of tangential continuity insures a smooth appearance to the overall spline curve. However it will not produce a uniform variation of the curvature across the junctions of the pieces, an imperfection which the eye can be quite sensitive to. A common example of this effect is the case of a straight line segment (which has no curvature) connecting tangentially to a circular arc (which has constant curvature): the sudden jump of curvature, from zero to some fixed value, may be viewed, in certain applications, as aesthetically

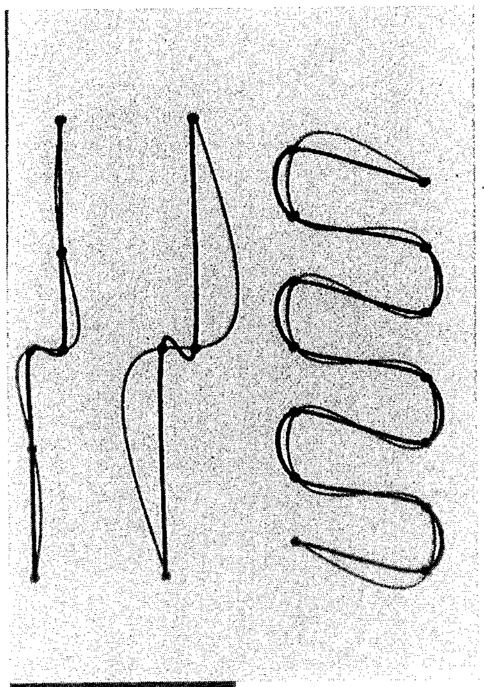

Figure 10A

Different methods for computing spline curves: (A) interpolating, (B) non-interpolating.

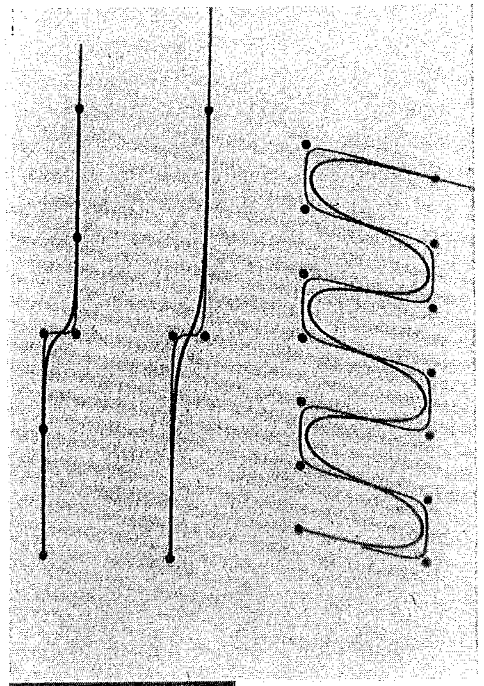

Figure 10B

undesirable. Therefore it is often imposed as an additional mathematical requirement to a spline fitting scheme that it provide curvature continuity.

(b) Interpolation.

The approximation of a shape by a spline curve is mathematically obtained by specifying a number of control points that will guide the spline along its path. However, one has to choose between two methods. Interpolating splines go through the guiding points, which must therefore be placed directly along the trajectory to be approximated. Non-interpolating splines go in the vicinity of the guiding points, which makes the relationship between the shape of the spline curve and the disposition of the control points somewhat harder to grasp. Although the latter scheme is used extensively in many computer aided design applications, it is often found to be more difficult to master by the designer and less desirable than interpolation.

(c) Locality.

The computation of a spline curve follows directly from the specification of its control points. At this stage, two types of computing methods are

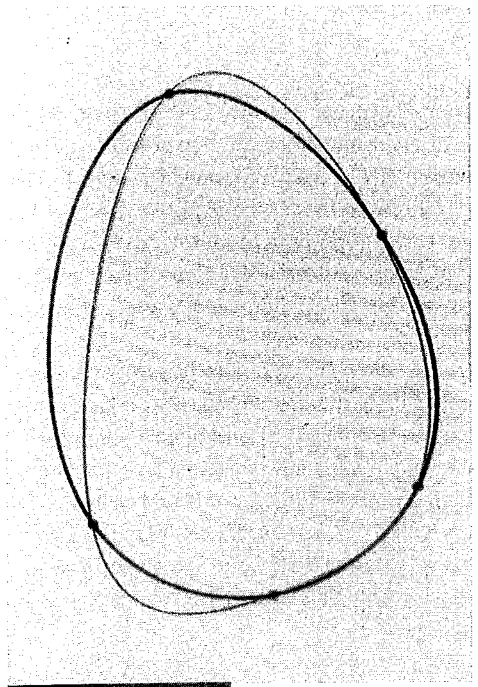

Figure 11.

Comparing the results of two interpolating spline-computing methods.

possible. Non-local splines are computed in one single process from the whole set of control points: therefore, their shape depends on all the control points. Although the influence of a particular control point on the shape of the curve decreases rapidly with the distance, this effect remains noticeable when a long flat stretch is followed by a sharp turn, producing annoying ripples that require iterated corrections of the control points. On the other hand, local splines are computed one piece at a time from a small number of neighboring control points. In addition to being simpler to calculate, they offer an easier control over the detailed shape of the spline.

The case for curvature continuity. interpolation and locality, would seem clear cut if it were not for the unfortunate fact of line that these three desirable properties cannot be satisfied mathematically all at once. As a result, there exist a variety of spline computing schemes that each exhibit at most two of the above properties. A few examples are illustrated in Figure 10. In addition, to complicate the matter even more, several spline computing schemes are subject to variations in which the resulting shape of the curve can be drastically modified by adjustment of various parameters that more or less model the 'tension,' 'stiffness' or 'resilience,' of the curve. For instance, Figure 11 illustrates two different ways of computing natural splines (a non local. interpolating scheme with curvature continuity). It shows how a non-symmetrical distribution of control points around an egg-shaped contour can yield either a nearly symmetric or a dissymmetric curve shape, according to the computation scheme.

FRED uses natural splines of the latter type. One could possibly claim that the first type would have been preferable, as it enforces more regularity and symmetry in shape design. At this point, however, one can safely say that there is probably no unique ideal spline method for character representation, although all existing systems appear to be using interpolating methods. It is certain that all methods require some practice on the part of the font designer in order to master the interactive shape editing process. The idiosyncrasies of each particular spline method, illustrated in Figures 10 and 11, can be in general overcome by the practice of defining a greater number of knots than would be strictly necessary. A better solution could be to do automatic spline fitting of digitized images, as recent work demonstrated [7], manual intervention being limited to adjustment of the resulting shape.

Conclusions

The Alto font system was designed with two main objectives in sight. The first one concerned the issue of a type font representation providing a good quality and a uniformity of appearance over a wide range of devices. This led naturally to the choice of spline outlines. The second objective concerned the production of digitized fonts: the goal was to be able to produce rapidly a large assortment of typefaces for many different experimental devices, by reproducing existing designs. The objective of designing new letterforms was never taken into consideration.

The availability of the Alto computer, with characteristics well matched to display oriented applications, permitted the design of a complete, interactive, self-contained font production system. The result turned out sufficiently efficient and user friendly to warrant an extended life beyond the experimental stage into a production oriented environment. The fact that the system is still being used today confirms its general usefulness and the soundness of its overall design. This should not, however, hide a number of shortcomings and limitations, as well as some obsolescent aspects of the implementation.

Acknowledgements

The general architecture of the Alto font production system, the principle of spline outlines of characters, was designed by Robert Sproull. within the scope of the PRESS printing system. The implementation was shared by Robert Sproull (PREPRESS) and the author (FRED). The original software was improved by Joe Maleson and adapted to a production environment by a team which included Chuck Hains. Ron Pellar. Paul Lam, Kerry La Prada and Ron Gechman.

The author is grateful to Ron Pellar and Judy Kaye for their help in preparing material for this paper.

References

1 R. Gechman. P. Baudelaire. R.F. Sproull et al. *XEROX font digitization software for vidicon based system, Operator Instruction Manual.* XEROX internal publication. February 1979.

2 L. Ruggles, Letterform Design Systems. Department of Computer Science, Stanford University, April 1983.

3 C.P. Thacker et al., ALTO : a personal computer, in : D. Siewiorek, C. Bell, and A. Newell (eds.), *Computer Structures : Principles and Examples*, McGraw-Hill, 1982.

4 C. Bigelow, "Technology and the Aesthetics of Type" *The Seybold Report*, Vol. 10, No. 24, August 1981.

5 C. Bigelow, "Technology and the Aesthetics of Type," *The Seybold Report*, Vol. 11, No. 11, February 1982.

6 P. Baudelaire. DRAW manual, in ALTO User's Handbook (B.W. Lampson et. al.), XEROX PARC, 1979.

7 M. Plass and M. Stone, "Curve-fitting with Piecewise Parametric Cubics", *SIGGRAPH '83 Conference Proceedings*, Computer Graphics, Vol. 17. No. 3, July 1983.

Author

Patrick Baudelaire is a retired computer scientist and research management executive. He conducted research at Xerox PARC on digital typography and on the practical design of digital fonts. Further in his career, he directed research laboratories at Digital Equipment Corporation and Thomson Multimedia.