Fat Curves – A title which is much too long to serve as running head

Chris Yao,[1†] Anton Notherone[2] and Jeremy Rokne[1]

[1] Department of Computer Science, The University of Calgary, Calgary, Alberta, Canada
[2] Another Department to illustrate the use in papers from authors with different affiliations

Abstract
Fat curves in two-dimensional Euclidean space are discussed. Previous work on fat curves is reviewed and a new definition is given for a fat curve having a smooth axis. The joining of two fat curves is discussed and a technique for scan-converting fat curves is presented.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Line and Curve Generation

1. Introduction

In computer graphics it is frequently desirable to represent a curve or the outline of an object as "thick" or a "fat" curve. This is particularly true for high-resolution graphics devices where curves represented by a chain of single pixels are too faint or where for aesthetic reasons the curve should have a fixed non-zero width. The definition of such objects pose particular geometric problems. Implementing the objects in terms of raster graphics similarly introduces further problems in the scan-conversion process.

If we look to standard references on algorithms for computer graphics such as Pavlides[1] then we do not find the fat line or curve concept. In fact there are only two fundamental concepts considered by Pavlides[1]: a thin curve with an orientation-dependent average width ranging from $1/\sqrt{2}$ pixels to 1 pixel and a full region. Fat curves are subsumed under the full region concept and no consideration is given to the particular problems encountered in dealing with them.

We want here only show a possible citation, such as the typical citation of the Foley et al. book [2] or a well known paper on ray tracing of volumetric dataset [5]. Fat lines are discussed in Bresenham under the concept of *Widelines* and he poses a number of questions relating to this concept. A fundamental question is how the fat lines are terminated and what the assumptions are when such lines are joined at decreasing angles. To quote Bresenham:

> Is *Wideline* a consistent concept, or is it a poorly specified and incompletely defined attempt to set up an implicit but fuzzily understood reference model of areas in contrast to lines? What is the shape of wideline ends? Is line width a geometric property in our original modeling coordinate space, or is such thickness only a picture-rendering cosmetic attribute akin to pseudo-pen size in final raster space? How should projective transformations affect *Widelines*? If width is a geometric attribute, what is the implied boundary definition?

In this paper we discuss some of the problems posed by Bresenham and we suggest solutions both in the underlying geometric setting and in the raster plane. We first give a precise definition of a fat line or curve as a continuous geometric object. Then, using this definition, we develop new algorithms implementing scan-conversion for such curves.

In the next section we survey previous definitions of fat lines concluding with the specific problems that we attempt to solve in this paper. In Section 4 we consider the analytic definition of smooth fat curves and we verify some simple properties. The problem of joining fat curves is then dealt with and we introduce the concept of a piece-wise smooth

† On leave from Suzhou Institute of Silk Textile Technology, P.R. China Since July 1988.

fat curve. Finally we give a method for the scan-conversion of the fat curves we defined in the previous sections.

2. Previous Work

Fat lines are discussed in several recent papers, but here we cite papers that have nothing in common with the *fat line* topic [5,4,6,7]. The concept is also used by several advanced workstations (see for example the IRIS User's Guide) and by typesetting systems such as PostScript.

Perhaps the most relevant discussion is found in the paper by Posch-Fellner.[3] They discuss an algorithm called options for double precision arithmetic. The impact of using single precision arithmetic is demonstrated in Table 1. Even when compiled with the double precision option, the program by Douglas produces results which deviate significantly from those produced by others. The formula used to calculate the squares of offset values presented in Table 1 is as follows:

$$\begin{aligned}
\lambda &= \frac{(x1*(x1-x2-x3)+x2*x3+y1*(y1-y2-y3)+y2)}{(x2-x1)^2+(y2-y1)^2} \\
x &= x1+\lambda*(x2-x1) \\
y &= y1+\lambda*(y2-y1) \\
\text{dis} &= (x3-x)^2+(y3-y)^2
\end{aligned}$$

In a recent debate on the accuracy of floating point calculations, Huggins stated that the arbitrary-precision arithmetic language 'bc' could be used to obtain precise results. We used this UNIX utility to calculate offset values for points C and D. On the VAX 8200, SEQUENT SYMMETRY and SUN 3/60, bc returned identical values for these points:

C: 28143.490838958534 D: 28143.49083895834

Forrest (p. 721) pointed out the well known fact that floating point calculations are still very much machine dependent. Machine dependency exposed further problems, which could be treated as problems of implementation but which are arguably more conceptual in nature as explained in the following sections.

2.1. Equidistant points from the anchor-floater line

The algorithm is based on the assumption that lines may be subdivided in an unambiguous manner using the maximum perpendicular offset. To our knowledge, the problem of two or more points being equidistant from the anchor-floater line has never been considered. Indeed, we only became conscious of this possibility when the same program yielded different results on ICL 3980 and SUN 3/60 computers. A sample problem is illustrated in Figure 5. Points C and D are equidistant from the anchor-floater line A–B. The inexact representation of floating point numbers results in C being selected on SUN workstations and D being selected on the ICL computer by the same program. With double precision arithmetic, the errors are negligible but are nevertheless sufficient to generate different results since published programs tend to use either a "greater than" or "less than" condition. GIMMS and the programs by Douglas and Wade select the first point from a set of identical offsets. White's program selects the last. The results therefore are variable and become dependent on the direction of digitising of lines. If, on the other hand, we select a point from this set at random, the procedure would become blatantly arbitrary. This problem poses other implications, which we will now examine in greater detail.

Figure 1: *Here is a sample figure.*

3. Our proposal in detail

This section describes in detail our proposal, as graphically shown also in Figure 1, with a non-sense text. Non-sense text follows text text text text text text text text text.

4. Digitising Errors

Like most cartographic algorithms, the Douglas–Peucker algorithm does not fully address the issue of digitising errors. When estimating truth values, it is usually assumed that the true line (in this case the analogue line) lies within the error band of the digitised line. This band is also known as the Perkal epsilon band. In his review on issues relating to the accuracy of spatial databases, Goodchild[5] indicated that researchers have proposed uniform, normal and even bimodal distributions of error across this band. This concept provides some basis for estimating the position of the true line at locations between digitised points. Here, we are merely concerned with the accuracy of digitised points. Whilst it is probable that operators digitise points along high curvatures more carefully than at intermediate positions, there is at present no sound basis for modelling the distribution of error along the line. As in the Circular Map Accuracy Standard, it is usual to assume a bivariate normal distribution of error when estimating the position of the true point. In the context of line simplification, absolute positional accuracy is less important than the relative position of points describing the shape of features along the line.

The DoE/SDD boundary data contain some gross digitising errors. For example, inlet X in Figure 2c does not feature on conventional Ordnance Survey 1: 50 000 maps of the area. The data are also not very accurate where coastlines are convoluted. Even if we ignore these and other gross errors, such as spikes, there will always be an element of random error in digitised data. It is reasonable to assume that points digitised from 1: 50 000 source material may only be accurate to within $+/-5$ metres. This algorithm does not lead to a substantial accumulation of rounding errors, hence the numerical errors discussed earlier tend to be very small compared with digitising errors.

For the purposes of our argument, it is unnecessary to undertake an exhaustive evaluation of the consequences Douglas and Peucker have treated overhangs and closed loops as different problems, and have used different methods to cope with each case.

4.1. Numerical Problems

The FORTRAN programs by Douglas, White, and Wade use single precision REALS when computing offsets (see results in Table 1). Whilst double precision accuracy may be attained through the use of compiler options, we are unsure whether previous research has been based on programs compiled in this manner. Wade's program was so compiled for use in our previous evaluations. Forrest stated that Ramshaw (1982) had to adopt carefully tuned double and single precision floating point arithmetic to compute the intersection of line segments whose end points were defined as integers. Forrest exclaimed "This is an object lesson to us all: constructing geometric objects defined on a grid of points, requiring ten bits for representation can lead to double precision floating point arithmetic!".

Most evaluative studies do not cite the co-ordinates in use. We do not know whether the published test lines were in original digitiser co-ordinates or whether they had been converted to geographic references. British National Grid co-ordinates for the administrative boundaries of England, Scotland and Wales (digitised by the Department of Environment (DoE) and Scottish Development Department (SDD)) are input to one metre accuracy and require seven decimal digits for representation if we include the northern islands of Scotland. At the South West Universities Regional Computer Centre these co-ordinates have been rounded to 10 metre resolution; even this requires six decimal digits. Seamless cartographic files at continental and global scales use much larger ranges of geographic co-ordinates.

Machine	Points	Calculated squares of offset values	
		Single Precision	Double Precision
ICL 3980			
	(C)	28199.351562500	28143.490838958
	(D)	28171.789062500	28143.490838961
VAX 8200			
	(C)	28253.095703125	28143.490838958
	(D)	28165.806640625	28143.490838958
SEQUENT SYMMETRY			
	(C)	28145.100000000	28143.490838961
	(D)	28145.100000000	28143.490838961
SUN 3/60			
	(C)	28253.095703125	28143.490838961
	(D)	28165.806640625	28143.490838961

NOTES

Offsets of points C and D from the anchor-floor line A–B as calculated using Wade's program. Points A, B, C and D are shown in Figure 5. The British National Grid coordinates (in metres) of the points are as follows:

Point A	238040	(x1)	205470	(y1)	ANCHOR
Point B	237890	(x2)	205040	(y2)	FLOATER
Point C	237810	(x3)	205320	(y3)	
Point d	238120	(x3)	205190	(y3)	

Note that the above co-ordinates may be used in conjunction with the expression presented in section 3.2.2a to check the tabulated results.

Table 1: *The Precision of Calculations*

A limited number of papers actually described improved for new algorithms or methods for visualization[5, 6, 7]. This may be caused by the complexity of the environment in which a method is used; issues of system architecture, user interface, data handling, etc. must be dealt with before a new presentation technique can show its full advantage. But even so, we think the field can use more contributions of this type.

There was also a discussion session on the merits of animation and special effects (such as sound) to support visualization. For example, in the area of flow visualization, it is quite common to use animation, and techniques for video registration have been developed.

5. Issues in Visualization

Scientific visualization is an interdisciplinary field, which can only flourish when computer graphics experts cooperate with specialists from application areas, and providers of computing, visualization, and data management facilities. Therefore, it is essential that all of these viewpoints are represented in research projects and also in meetings such as this workshop. It is not enough that suitable display algorithms, data structures, or user interfaces be developed, but also that these be integrated in usable systems and evaluated by expert users. This complex environment, and the complex systems it requires, call for a common language between different parties involved, and therefore *a reference model*, or an abstract description summarizing the entire process of data visualization, is needed.

At the Delft workshop, an attempt was made to continue the meetings of sub-groups as started in Clamart[1], but it appeared that a useful description of sub-areas or sub-problems should be based on a stable conceptual framework. Except for the flow visualization group, the subgroup definitions were abandoned, and instead it was decided to concentrate on design of an initial reference model; a first attempt is currently being undertaken by Lesley Carpenter and Michel Grave. At the same time, the separate flow visualization subgroup (chaired by Hans-Georg Pagendarm) agreed to design a general model of the flow visualization process! In addition, arrangements were made for the exchange of test data sets for system evaluation, and the exchange of information on and experience with visualization software.

Special discussion sessions were held about the practice the "circle-brush" algorithm. In this algorithm a solid disk is assumed to move along a trajectory in R^2. This trajectory is then scan-converted into the raster plane. and experience of the Stardent AVS system, and about general evaluation methods for visualization software. There is an obvious need to share experience or even make a formal (comparative) evaluation of systems, but this is also hampered by lack of a common framework, and also by the continuing development of visualization systems.

Interactive visualization was also an interesting subject for discussion, which yielded a lively debate[1]. In a session about visualization facilities, it was suggested from experience that large research institutes might well have to employ specialized 'visualization experts', to bridge the gap between complex numerical simulations and sophisticated visualization facilities.

6. Results

This section only refers a table with some numerical results (see Table 1).
Non-sense text follows text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text texttext text text text text.

7. Conclusions

Here are conclusions and possible extensions. As shown by the results reported in Section 6 and in Figure **??** (see color plates), conclusions. Conclusions conclusions.

Acknowledgements

Introduce here, if you would....

References

1. Y. Le Lous, "Report on the First Eurographics Workshop on Visualization in Scientific Computing", *Computer Graphics Forum* **9**(4), pp. 371–372 (December 1990).
2. J. Foley, A. van Dam, S. Feiner, J. Hugues, and R. Phillips. *Introduction to Computer Graphics*. Addison Wesley, 1993.
3. K.Ch. Posch and D.W. Fellner. The Circle-Brush Algorithm. *Transactions on Graphics*, **18**(1):1–24, 1989.
4. T.A. Foley, H. Hagen, and G.M. Nielson. Visualizing and modeling unstructured data. *The Visual Computer*, (9):439–449, 1993.
5. M. Levoy. Efficient ray tracing of volume data. *ACM Transactions on Graphics*, **9**(3):245–261, July 1990.
6. T. Porter and T. Duff. Compositing digital images. *ACM Computer Graphics (Proc. of SIGGRAPH '84)*, **18**:253–259, 1984.
7. R. Ronfard and J. Rossignac. Full-range approximation of triangulated polyhedra. *Computer Graphics Forum (Eurographics'96 Proc.)*, **15**(3):67–76, 1996.