Antialiasing of Curves by Discrete Pre-filtering

A.E. Fabris† and A.R. Forrest*
Universidade de São Paulo University of East Anglia

ABSTRACT

Pre-filtering is generally considered the ideal approach to anti-aliasing but is difficult to perform exactly for complex geometries such as curves or for arbitrary choice of filters. We present a discrete pre-filtering technique for anti-aliasing Bézier curves using arbitrary filters which is numerically and geometrically robust and whose accuracy is controllable.

CR Categories and Subject Descriptors: I.3.3 [Computer Graphics]: Picture/Image Generation — antialiasing
Additional Keywords: Bézier curves, pre-filtering

1 INTRODUCTION

The causes of aliasing in computer generated imagery are well known and may be broken down into two components: aliasing due to inadequate sampling of the ideal image and aliasing due to incorrect or inadequate reconstruction of the displayed image from the samples. Many techniques have been proposed but there is no ideal or universal solution and compromises have to be made. As Mitchell and Netravali [29] point out, graphics has generally avoided the issues of reconstruction but with increasing use of non-c.r.t. displays it is now clear that anti-aliasing should take into account the reconstruction properties of the display device [22]. In this paper we describe an anti-aliasing technique for curves which not only avoids the geometric and numerical problems found in previous techniques but also permits the use of arbitrary filters thus enabling rendering to be tuned to specific output devices. This is achieved by combining the point containment approach of Corthout and Pol [6,7,8] with a pre-filtering technique akin to but more general than Gupta and Sproull [18]

2 PRE- AND POST-FILTERING

A naïve view of aliasing ascribes the problem to treating a pixel as a point rather than an area. Following this view, aliasing can be avoided by area sampling rather than point sampling: we compute the fractions of a pixel covered by components of the *ideal image* lying within the pixel and use these fractions to weight the fragment colours, accumulating the sum of fragment contributions into the pixel value. Signal processing, on the other hand, shows that the problem lies in attempting to reconstruct an image which contains spatial frequencies higher than the maximum spatial frequency which the output device can display: we cannot correctly reconstruct an image which contains frequencies greater than or equal to half the sampling frequency. The solution is therefore to remove high frequency components before display by filtering. Area sampling can be shown to be a simple form of filtering—box filtering, but signal processing proves that better filters are possible.

There are two possible methods of filtering the ideal image: pre-filtering and post-filtering. In post-filtering the ideal image is point sampled at a higher rate than can be displayed, with multiple samples per pixel, and the samples are then numerically filtered by a discrete digital filter. Area sampling may be approximated numerically by averaging a regularly spaced super-sampling of a pixel. More sophisticated digital filters such as the Bartlett filter [10] give better results. However, regular super-sampling may still give rise to aliasing through missing important fragments. To overcome this, super-samples may be distributed in some stochastic manner [4] and reconstructed by a more complicated process, but the downside is the introduction of noise into the image. This may be considered more acceptable than jaggies although still highly visible.

The advantages of post-filtering lie largely in the simplicity of the process. In some applications, such as ray tracing, super-sampling followed by post-filtering is the only reasonable solution to aliasing. The disadvantages are the greatly increased costs of computation and the failure to eliminate aliasing completely.

If a precise geometric description of the ideal image is available, say analytic or piecewise analytic descriptions, then it is theoretically possible to filter the ideal image analytically to remove high frequency spatial components before sampling and then to point sample the bandwidth limited ideal image to produce alias-free images. For example, for simple geometries such as straight lines, we can pre-filter the geometry by a box filter by deriving expressions for pixel area coverage: sampling is then reduced to substitution of pixel coordinates in the expressions.

The advantages of pre-filtering are that once the ideal image has been analytically pre-filtered, a single sample per pixel suffices. The expectation is that if we can properly pre-filter the ideal image, then we will generate high quality displayed images. The main disadvantages heretofore have been the restricted range of geometries that could be pre-filtered and the restricted range of filters which could be employed. Pre-filtering will however remain a technique more appropriate to high quality two-dimensional images containing lines, regions and text rather than images of three-dimensional scenes.

3 PRIOR WORK ON PRE-FILTERING

Pitteway and Watkinson [33] describe the incorporation of area sampling in the Bresenham line algorithm, but do not handle line ends properly. Gupta and Sproull [18] developed an antialiased

† Instituto de Matématica e Estatistica, Caixa Postal 66281, CEP 05315-970, São Paulo - SP - Brazil, aef@ime.usp.br
* School of Information Systems, Norwich NR4 7TJ, U.K. forrest@sys.uea.ac.uk

version of the Bresenham line algorithm in which, at least notionally, the ideal line is convolved with a circularly symmetric filter. In their implementation a lookup table is constructed which contains the fractions of the volume of the conical filter intercepted by the ideal line, indexed by the perpendicular distance of the pixel centre from the centre of the line. This table can be generated from a simply derived analytic expression. The circular symmetry enables a single one-dimensional table to cover lines at all possible angles. Analytic expressions for line ends, however, are difficult to derive and Gupta-Sproull resort to less precise two-dimensional tables. Whilst circular symmetry leads to a fast incremental algorithm, the conical filter does not give rise to a flat constant signal for constant sample values, thus exhibiting what Mitchell and Netravali call sample-frequency ripple. Forrest gives examples of this for various filter radii [16].

Feibush, Levoy and Cook [14] describe antialiasing of polygons in which polygons are first split into triangles; the volume of the filter intersected by each triangle is used as a weight. In effect this amounts to a discrete approximation to the convolution integral applied to polygon fragments. In one implementation a two-dimensional lookup table stores triangle-cone intersections. If the chosen filter is not circularly symmetric, the lookup table is considerably more complex, being four-dimensional. Abram, Westover and Whitted [1] develop a more complex approach in which polygon-filter convolution is classified into several different cases, each of which employs a discrete approximation to the convolution integral.

Duff's polygon scan conversion by exact convolution [11] numerically integrates the convolution integral to antialias simple polygons. Filters are more general than in other methods but are limited to bivariate polynomials which can be used to approximate sinc and other filters.

McCool [27] describes a method whereby an image consisting of Gouraud-shaded triangles can be represented by simplex splines; these can then be convolved with a box spline filter to form a set of prism splines representing the filtered image. This permits analytic filtering by filters which can be constructed from box spline basis functions, a special case being tensor product B-splines. Any further filtering for reconstruction is performed by digital post-processing.

4 PRIOR WORK ON ANTIALIASING CURVES

Aliased scan conversion of curves is still a topic for further research: whilst there exist efficient algorithms for circles and ellipses (although long thin ellipses still need special care), more general parametric, explicit or implicit curves generally require careful attention to both geometric and numerical detail in order to provide robust and efficient algorithms. In many cases the approach taken is to reduce the curve to a piecewise linear approximation which can then be antialiased; avoiding any visual evidence of polygonisation requires care.

Lien, Shantz and Pratt [26] develop an adaptive forward difference method for rendering curves and briefly mention a simple adaptation to their algorithm which enables an admittedly rough approximation to area sampling to be made. In effect the curve is approximated by short straight line segments. Klassen [23] emphasises the geometric problems found in rendering curves, particularly loops, cusps, and crossings, and goes on to develop a more robust approach than [26]. A curve is split into monotonic sections (with respect to the x or y axes) then adaptive forward differencing is used to divide the curve into short straight line segments which may then be antialiased by the Gupta Sproull algorithm [18] or by any other filter which can be accessed from a lookup table in a similar manner. The transitions from x major to y-major line segments (and vice versa) need special attention to avoid "nicks" in the output. Klassen pays particular attention to numerical detail.

Field [15] describes a fast incremental method for antialiasing circles and ellipses based on a predictor-corrector method to compute filter values. The filter employed is an approximation to area sampling. Pitteway and Banissi [32] describe an integer algorithm for rendering antialiased ellipses which employs an approximation to area sampling sufficient for a two bit per pixel system, this giving a marked improvement in the rendering of fonts composed of conic segments.

Prior methods are seen to be restricted in terms of filters, employing either box or conical filters which are known to be poor. Curves, apart from circles and ellipses, are approximated by line segments rather than being antialiased directly as curves, and geometric special cases need careful treatment.

5 DISCRETE PRE-FILTERING

Consider the problem of pre-filtering an image as a three-dimensional problem in which the ideal image is a function I(x,y) of the continuous spatial variables x and y, I representing the intensity (or colour) of the ideal image. The cross section of a line in a plane perpendicular to the line centre has a rectangular profile. Looking at the problem in this way, we can antialias by area sampling simply by convolving the intensity profile of the line with a box filter. Figure 1 shows several intensity profiles of three widths of lines (dark rectangles) with the corresponding box filtered profiles superimposed in gray. Area sampling is reduced to point sampling the convolved intensity profile. Figure 2 shows a three-dimensional view of a typical section of a 1.5 pixel-wide line after box filtering (the ruled surface end caps are not drawn).

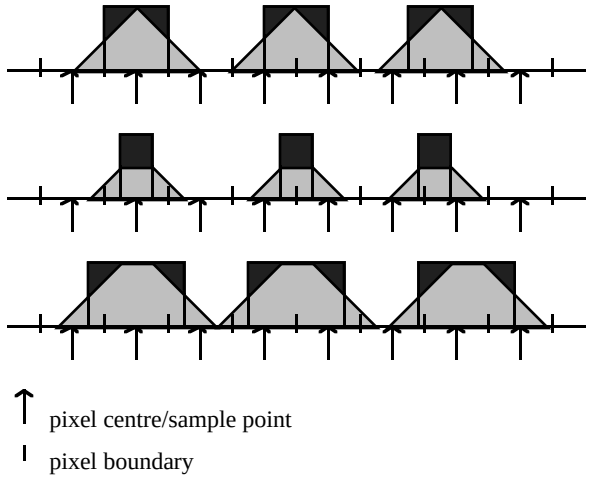


Figure 1

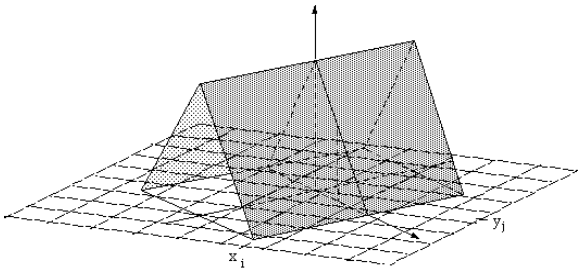


Figure 2

Lines and curves may be generated by continuous sweep of a brush along the mathematical centre of the line or curve [17]. Guibas et al. [19] refer to this process as convolving the curve with a brush, but in this paper we shall use the term *dilation* to denote the effect of brushing in order to avoid confusion with convolving a geometric object with a filter. Antialiased lines and curves of arbitrary width may similarly be generated by brushing with an ideal brush which has been convolved with some appropriate form of filter, resulting in intensity profiles of more complicated form than that shown in Figure 2 but of similar nature. Crow [9] describes line and character generation by an assembly of overlapping bump functions positioned to sub-pixel accuracy. The Quantel Paintbox painting system developed somewhat later uses a similar technique in which hand-drawn curves are painted as overlapping imprints of sub-pixel positioned cylindrical brushes convolved with an apparently ad hoc filter [34]. Whitted strokes curves with a brush of arbitrary footprint and intensity profile, using a Z-buffer technique to overwrite only those pixels where current position of the brush indexes a greater intensity than in previous writes [36]. In our approach however, we do not approximate the curve by a series of overlapping blobs or a series of straight line segments but sample the convolved ideal image by an approximate method whose accuracy is controllable both spatially and in terms of intensity [7,8,12,13].

6 POINT CONTAINMENT AND ANTIALIASING

The conventional approach to rendering would be to scan convert the convolved version of the curve but this as we have remarked earlier leads to numerical and geometric problems. Instead we chose to use the *point containment algorithm* developed by Corthout et al. to implement a pre-filtering algorithm along the lines of but more general than Gupta-Sproull [18]. Extension of their method to handle curves would involve computing the closest distance of a curve centre line to a pixel centre, a complex geometric calculation obviated by our approach. Brooks [3] quoting Poulton points out that if current hardware trends continue, the number of pixels per primitive rendered by hardware will approach unity, and in such circumstances we might as well compute pixels directly from the underlying geometry rather than first approximating the geometry by polygons or line segments. The point containment approach is an example of this strategy, generating pixels directly from Bézier curves and thus avoiding the difficulties in curve rendering tackled by Klassen [23] and Lien et al. [26]. In [5] Corthout and Jonkers describe an algorithm for determining the containment of a point within a region bounded by discrete Bézier curves. This is extended in [6] to encompass discrete rational Bézier curves and in [7,8] to support dilation and erosion of discrete Bézier curves and regions bounded by discrete Bézier curve by brushes which may be regions bounded by discrete Bézier curves. Extensions to the algorithm include the ability to transform the brush whilst stroking.

Corthout and Pol's thesis [8] contains full details of the mathematical theory of discrete Bézier curves, a version of the Jordan curve theorem for regions bounded by discrete curves, a formal development of the point containment algorithm and a description of its implementation in dedicated silicon, the Pharos chip. Recognising that at some stage we have to make the transition from the continuous world to the discrete, Corthout and Pol define discrete Bézier curves in a discrete model space which is of higher precision than device space. Discretisation is readily performed in a controlled fashion using a simple discrete integer space rather than floating point. [8] contains details of the overall integer precision required for rendering on a chosen device together with proofs of robustness and accuracy.

For each pixel in the image the point containment algorithm has to determine whether the pixel centre lies within the dilated curve. Rather than testing all pixels in the image, an obvious optimisation restricts testing to pixels lying within the bounding box of the curve dilated by the bounding box of the brush. Further optimisations of increasing complexity are described in [8]. Determining whether a pixel centre lies within the region defined by the dilated curve is non-trivial but fortunately the dual problem of determining whether the curve intersects the region defined by dilation of the pixel centre by the brush is rather simpler, Figure 3. Corthout and Pol describe an efficient recursive algorithm using integer arithmetic in [8].

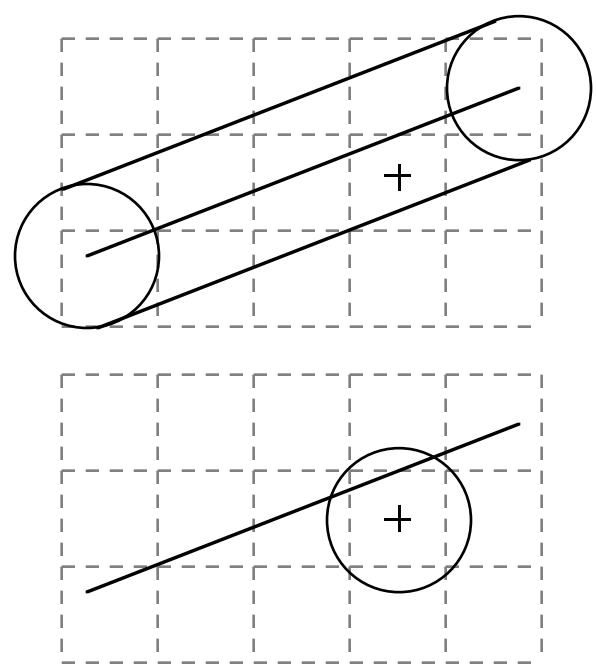


Figure 3

A discrete Bézier curve can be computed by subdivision on an integer grid of specified resolution. Figure 4 shows a typical 8-connected discrete curve computed on a grid with 4 times pixel resolution with a circular brush of radius 1.25 pixels centred on each point of the discrete curve to approximate the dilated curve. Pixels whose centre lie within the dilated curve are rendered by the point containment algorithm: inclusion of just one of the discrete curve points within a brush centred on a pixel will cause that pixel to be rendered, Figure 3. The accuracy of the discrete intersection test is a function of the sub-pixel resolution chosen for the discrete curve which reflects both the need to accommodate

rounding errors [8] and the higher spatial resolution dictated by antialiasing [25].

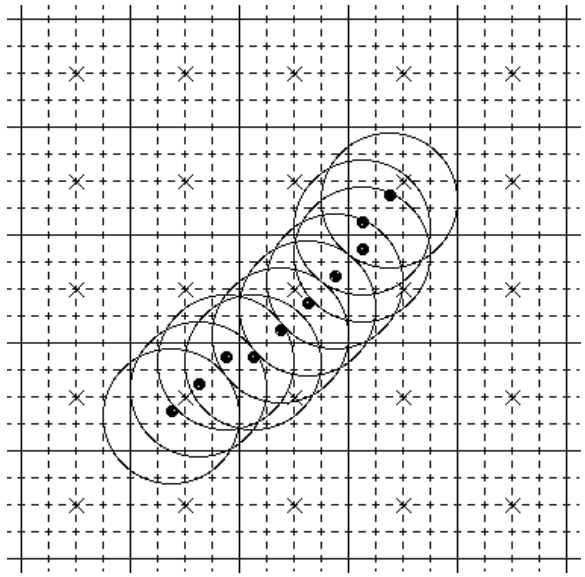


Figure 4

The algorithm relies on three predicates:

- `NotInDilatedBBox` — if the test pixel is outside the dilation of the bounding box of the discrete Bézier curve dilated by the bounding box of the brush, then `NOT_COVERED` is returned and the algorithm terminates.
- `At` — if the query point is covered by the brush placed at either end of the curve, `COVERED` is returned. For a circular or square brush this predicate is simply implemented, as in our initial system. More elaborate shapes are discussed in [8].
- `BaseCase` — if the curve is primitive, that is to say length of the curve is ≤1 measured in the discrete space selected, then `TRUE` is returned.

6.1 Standard Stroking

```
Discrete Curve C;
Brush B;
Pixel P;
{for(each pixel P in the image)
    if(Dilate(C,B,P)==COVERED) then Render(P);
}

Dilate(C,B,P)
/* returns COVERED if P is covered by C+B */
DiscreteCurve C
Brush B
Pixel P
{   if(Stroke(C-P,B)==COVERED) then
        return(COVERED)
    else return(NOT_COVERED);
}
```

```
Stroke(C,B)
/* detects whether the origin is covered */
/* by C+B */
DiscreteCurve C;
Brush B;
{   n=Length(C);
    if(NotInDilatedBBox(C,B))
return(NOT_COVERED);
/* test start point of curve */
    if(At(c[0],B)) return(COVERED);
/* test end point of curve */
    if(At(C[n],B)) return(COVERED);
    if(BaseCase(C)) return(NOT_COVERED);
    left=LeftHalf(C);
    right=RightHalf(C);
    if(Stroke(left,B)==COVERED)
        return(COVERED);
    if(Stroke(right,B)==COVERED)
        return(COVERED);
    return(NOT_COVERED);
}
```

`Dilate` translates the test pixel and the discrete curve so that the test pixel is centred at the origin. `Stroke` recursively splits the discrete curve into left and right halves when necessary using repeated halving of the polygon sides in a conventional manner, Figure 5. Control vertices are held at higher integer precision than the discrete space to allow for any accuracy loss in subdivision down to the `BaseCase`. Note that more than one point on the discrete curve may lie within a pixel but it suffices to find only one.

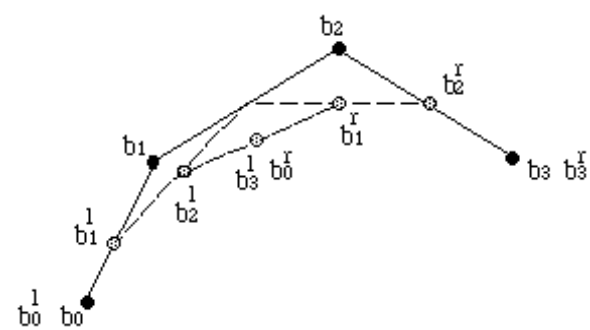


Figure 5

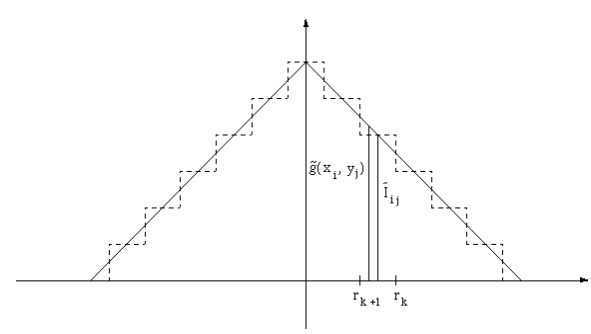


Figure 6

In our antialiasing version, we first convolve the brush with an antialiasing filter. The convolved brush is then approximated in a piecewise constant manner by a stack of nested brushes, as shown in Figure 6 for a box filter. The discrete Bézier curve is then

dilated by the stack of brushes and the appropriate gray level (or antialiasing fraction) is determined by finding the smallest brush in the stack which is intersected by the discrete curve. A simple implementation of the algorithm uses the procedure `Stroke` as described above:

6.2 Antialiased Stroking

```
DiscreteCurve C;
StackOfBrushes SB;
Pixel P;
Index H;
/* H is an index to a brush in the brush */
/* stack */
/* Height(H) is the height of the brush */
/* in the stack and hence the fraction of */
/* the curve colour used for rendering */
for(each pixel P in the image)
{   if(AntialiasedDilate(C,SB,P,H))==COVERED
        then Render P with Height(H);
}

AntialiasedDilate(C,SB,P,H)
/* returns COVERED and H if P is covered*/
/*  by C+SB(H) */
DiscreteCurve C;
StackOfBrushes SB;
Pixel P;
Index H;
{   n=number of brushes in stack SB
    H=0
    repeat with i=1 to n {
        if((Stroke(C-P,SB(H))==COVERED)
            then H=i;
        else exit repeat;
    }
    if(H==0) then return(NOT_COVERED);
    else return(COVERED);
}
```

In this version the brush stack is tested in order from bottom to top. Other strategies could be used: for example, test the largest brush for the `NOT_COVERED` case (early exit), then test the smallest brush. If this returns `COVERED`, we are done, otherwise we test the middle brush in the brush stack to determine whether the highest containing brush is larger or smaller than the middle brush, and so on, i.e. by interval halving. Details of the method employed for generation of the brush stack by optimal discrete stepwise approximation of the intensity profile of the filtered brush to a specified accuracy are given in [12,13]. On-demand generation of sections of the brush profile rather than using a pre-computed set of brush slices would enable a root finding procedure to determine the antialiasing fraction to arbitrary precision by interval halving—we have not yet implemented this approach.

7 RESULTS

Grayscale illustrations were computed at up to 6x6 sub-pixel resolution and viewed on an Apple Macintosh with a colour monitor using the “special gamma” setting. For reproduction, the images were saved as PostScript files and printed on a Hewlett-Packard LaserJet 4 Plus printer at 600 dpi using the “calibrated colour/grayscale” option in the standard print dialogue. Images viewed on the screen are rather smoother as a consequence of the blurring effect of the gaussian-type reconstruction of the CRT. Differences between filters are less noticeable than in the printed versions. Laser printing allowed us more control over grayscale than would have been possible using photography. The illustrations are best viewed from approximately 0.5 metres using a strong incandescent bulb for lighting.

Plate 1 shows seven Bézier curves, parabolae and cubics, which are drawn aliased using the point containment algorithm (note the algorithm does not generate Bresenham lines). The curves are chosen to demonstrate particular geometric features such as near-vertical and near-horizontal portions, inflexion, low and high curvature, a cusp and its bordering configurations (a small loop and a curve with two inflexion points), and finally a large loop. In Plates 2-5 we use a circular brush convolved with a variety of filters to create stacks of circular brushes. Plate 2 shows the seven curves rendered as one pixel wide curves using a discrete version of the Mitchell-Netravali filter [29]. In Plate 3 we demonstrate the rendering of a low curvature parabola using a variety of filters. On balance the Mitchell-Netravali filter proved the best compromise between sharpness and smoothness or lack of braiding. The poor performance of box filtering is obvious. Plate 4 illustrates the ability of the algorithm to render curves with a variety of thicknesses and also the correct handling of a tight loop which is progressively filled in as the curve thickness increases. Plate 5 demonstrates the effect of gray level quantisation using a box filter and a truncated-sinc filter.

8 COMMENTS AND APPLICATIONS

Our technique follows the philosophy of the Corthout-Pol point containment technique [5,6,7,8] in which the problems of discretisation are acknowledged and tackled by casting the problem to be solved as a discrete integer problem from the outset rather than attempting to accommodate all the problems of numerical accuracy which follow from floating point discretisation, geometric approximation, or a less rigorous approach to discretisation later in the rendering process. Using their theory of discrete Bézier curves we are able to avoid the numerical problems which typically arise in computational geometry, especially with special cases. The Corthout-Pol point containment methods are provably accurate and robust. We gain the ability to use discrete approximations of any antialiasing filter. In [13] we describe how stacks of brushes can be generated to approximate the convolved brush to any required accuracy. We pay a price in terms of efficiency. At first sight, point containment is quadratic in terms of pixel resolution but Corthout and Pol [8] have shown that by exploiting area coherence and the convex hull properties of discrete Bézier curves, their method can be reduced to quasi-linear. Brooks’ remarks cited earlier [3] are apposite. Furthermore, the method lends itself to hardware implementation: the Pharos chip fabricated by Philips implements the point containment algorithm in full and could be used serially or in a variety of parallel configurations for antialiasing. We make no claims for the efficiency of a software implementation.

Solution of the cases where the curve has more than one intersection with a pixel, for example where a curve crosses itself, need further discussion. There is a temptation to think of the problem in terms of area sampling but this is inappropriate as the convolved ideal image is point sampled: the issue is how convolved strokes should be merged prior to sampling. If we consider the case of aliased curve brushing, the ideal image is a union of the ideal strokes and hence it suffices to determine

whether any one portion of the curve lies within the brush. By extension, if we consider one of the slices of a filtered brush, then we need to find containment in the union of the corresponding level set and hence the containment of at most one fragment of curve in the brush slice to determine whether that gray level has been reached within the pixel. Thus the approach we have taken is to detect the smallest (highest) brush intersected, computing a MAX value over all the curve's points within the pixel. There are no apparent problems with the looping curves in Plates 2 and 4. Where two fragments of curve run side by side without overlapping and together cover a complete pixel, the algorithm gives an incorrect rendering, requiring global knowledge of the curve configuration within the pixel rather than serial exploration, but this is expensive to implement. The problem is related to the bulge elimination problem discussed by Bloomenthal in the context of generating implicit branching surfaces by convolution of skeletons [2] and the solution may lie there or in investigation of level curves [28, 20].

Since our method is particularly suited to handling intricate details such as cusps and serifs in a robust fashion, an obvious application is in the generation of grayscale fonts from outline masters, at the requisite multiple sub-pixel positions if hardware is used, or pre-scanned and cached using an approach similar to that described by Naiman [30,31]. Hersch et al. [21] argue that for readability it is better to scan convert fonts using perceptually based tuning of the font outlines taking into account typographical features rather than to employ better filters, but their antialiasing is by post-filtering (simple averaging) of high resolution bitmaps (see also comments by Corthout and Pol [7]). Their approach ensures that similar typographical elements are, as far as possible, scan converted into identical groups of pixels rather than into varying sets of pixels which depend on sub-pixel positioning. Warnock, on the other hand, generated quite readable grayscale sub-pixel positioned fonts at small point sizes without perceptual tuning [35]. Our approach permits the use of sophisticated filters and does not require any geometric adjustment of font outlines before straightforward scan conversion. Whether text antialiased using our approach will prove to be as readable as Hersch's perceptually tuned fonts is a matter for future experiment. It is simple to modify our algorithm to allow the brush size to vary continuously whilst stroking curves [7] without the complication of computing offsets from the curve centreline [24]—a property which may be useful for generating Chinese and other brushed characters.

The implementation described in this paper covers only the antialiasing of curves as strokes and assumes black curves drawn over a white background permitting write-only image generation. The technique can of course be used in a simple manner for straight lines and polygons as well as parametric and rational spline curves. Future implementations will include antialiasing of region boundaries and the use of read-modify-write (lerping). More complex brushes such as orientable brushes and brushes defined by closed sequences of Bézier curves need investigation.

ACKNOWLEDGEMENTS

We wish to acknowledge the Advanced Technology Group, Apple Computer, Cupertino, for the provision of computing equipment and software through Apple Computer UK. The work of A.E. Fabris, on leave from the Instituto de Matématica e Estatistica, University of São Paulo (São Paulo, Brazil), was partially supported by a grant from the Conselho Nacional de Desenvolvimento CientÍfico e Tecnológico (CNPq). The initial inspiration for this work is due to Marc Corthout and Evert-Jan Pol.

REFERENCES

[1] G.D. Abram, L. Westover, L. and J.T. Whitted. Efficient Alias-Free Rendering Using Bit-Masks and Look-up Tables. In Brian A. Barsky, editor, *Computer Graphics, (SIGGRAPH 85 Conference Proceedings)*, volume 19, pages 53-60. ACM SIGGRAPH, July 1985. ISBN 0-89791-166-0.

[2] J.I. Bloomenthal. Bulge Elimination in Convolution Surfaces. *Computer Graphics Forum*, 16(1): 1-11, January 1997. ISSN 0167-7055.

[3] F.P. Brooks Jr. Springing into the Fifth Decade of Computer Graphics—Where We've Been and Where We're Going! In Holly Rushmeier, editor, *SIGGRAPH 96 Conference Proceedings*, Annual Conference Series, page 513. ACM SIGGRAPH, Addison Wesley, August 1996. ISBN 0-89791-746-4.

[4] R.L. Cook. Stochastic Sampling in Computer Graphics. *ACM Transactions on Graphics* 5(1): 51-72, January 1986. ISSN 0730-0301.

[5] M.E.A. Corthout and H.B.M. Jonkers. A New Point Containment Algorithm for B_Regions in the Discrete Plane. In R.A. Earnshaw, editor, *Theoretical Foundations of Computer Graphics and CAD*, NATO Advanced Study Institute Series F, F40, pages 297-306. Springer-Verlag, 1988. ISBN 0-387-19506-8.

[6] M.E.A. Corthout and E.-J.D. Pol. A Point Containment Algorithm for Regions in the Discrete Plane Outlined by Rational Bézier Curves. In J. André & R.D. Hersch, editors, *Raster Imaging and Digital Typography*, pages 169-179. Cambridge University Press, 1989. ISBN 0-521-37490-1.

[7] M.E.A. Corthout and E.-J.D. Pol. Supporting Outline Font Rendering in Dedicated Silicon: the PHAROS Chip. In R.A. Morris and J. André, editors, *Raster Imaging and Digital Typography II*, pages 177-189. Cambridge University Press, 1991. ISBN 0-521-41764-3.

[8] M.E.A. Corthout and E.-J.D. Pol. *Point Containment and the PHAROS Chip*. Joint Ph.D. Thesis, University of Leiden, The Netherlands, March 1992.

[9] F.C. Crow. The Use of Grayscale for Improved Raster Display of Vectors and Characters. In Richard L. Phillips, editor, *Computer Graphics, (SIGGRAPH 78 Conference Proceedings)*, volume 12, pages 1-5. ACM SIGGRAPH, August 1978.

[10] F.C. Crow. A Comparison of Antialiasing Techniques. *IEEE Computer Graphics and Applications*, 1(1): 40-48, January 1981. ISSN 0272-1716.

[11] T.D.S. Duff. Polygon Scan Conversion by Exact Convolution. In J. André & R.D. Hersch, editors, *Raster Imaging and Digital Typography*, pages 154-168. Cambridge University Press, 1989. ISBN 0-521-37490-1.

[12] A.E. Fabris. *Robust Anti-aliasing of Curves.* Ph.D. Thesis, University of East Anglia, U.K., November 1995.

[13] A.E. Fabris and A.R. Forrest. *Robust Anti-aliasing of Curves.* December 1996, submitted for publication.

[14] E.A. Feibush, M. Levoy and R.L. Cook. Synthetic Texturing Using Digital Filters. In James J. Thomas, editor, *Computer Graphics, (SIGGRAPH 80 Conference Proceedings)*, volume 14, pages 294-301. ACM SIGGRAPH, July 1980. ISBN 0-89791-1021-4.

[15] D.A. Field. Algorithms for Drawing Anti-aliased Circles and Ellipses. *Computer Vision Graphics and Image Processing*, 33(1): 1-15, January 1986. ISSN 0734-189X.

[16] A.R. Forrest. Antialiasing in Practice. In R.A. Earnshaw, editor, *Fundamental Algorithms for Computer Graphics*,

NATO Advanced Study Institute Series F, F17, pages 113-134. Springer-Verlag, 1985. ISBN 0-387-13920-6.

[17] P.K. Ghosh. A Mathematical Model for Shape Description using Minkowski Operators. Computer Vision, Graphics, and Image Processing, 44(3): 239-269, December 1988. ISSN 0734-189X.

[18] S. Gupta and R.F. Sproull. Filtering Edges for Gray-Scale Displays. In Henry Fuchs, editor, *Computer Graphics, (SIGGRAPH 81 Conference Proceedings)*, volume 15, pages 1-5. ACM SIGGRAPH, July 1981. ISBN 0-89791-045-1.

[19] L.J. Guibas, L.H. Ramshaw and J. Stolfi. A Kinetic Framework for Computational Geometry. In *Proceedings of the IEEE 1983 24th Annual Symposium on the Foundations of Computer Science*, pages 100-111. IEEE Computer Society Press, 1983.

[20] W. Heidrich, M.D. McCool and J. Stevens. Interactive Maximum Projection Volume Rendering. In G.M. Nielson and D. Silver, editors, *IEEE Visualization '95*, pages 11-18, CP-3. IEEE Computer Society Press, 1995. ISBN 0-8186-7187-4.

[21] R.D. Hersch, C. Bétrisey, J. Bur and A. Gürtler. Perceptually Tuned Generation of Grayscale Fonts. *IEEE Computer Graphics and Applications*, 15(6): 78-89, November 1995. ISSN 0272-1716.

[22] R.V. Klassen. *Device Dependent Image Construction for Computer Graphics.* Ph.D. Thesis, University of Waterloo, Waterloo, Ontario, July 1989.

[23] R.V. Klassen. Drawing Antialiased Cubic Spline Curves. *ACM Transactions on Graphics* 10(1): 92-108, January 1991. ISSN 0730-0301.

[24] R.V. Klassen. Variable Width Splines: a Possible Font Representation? *Electronic Publishing—Origination, Dissemination and Design (Special Issue, Proceedings of RIDT'94)*, 6(3): 183-194, September 1994. ISSN 0894-3982.

[25] W.J. Leler. Human Vision, Anti-Aliasing, and the Cheap 4000 Line Display. In James J. Thomas, editor, *Computer Graphics, (SIGGRAPH 80 Conference Proceedings)*, volume 14, pages 308-313. ACM SIGGRAPH, July 1980. ISBN 0-89791-021-4.

[26] S.-L. Lien, M. Shantz and V.R. Pratt. Adaptive Forward Differencing for Rendering Curves and Surfaces. In Maureen C. Stone, editor, *Computer Graphics, (SIGGRAPH 87 Conference Proceedings)*, volume 21, pages 111-118. ACM SIGGRAPH, July 1987. ISBN 0-89791-227-6.

[27] M.D. McCool. Analytic Antialiasing with Prism Splines. In Robert Cook, editor, *SIGGRAPH 95 Conference Proceedings*, Annual Conference Series, pages 429-436. ACM SIGGRAPH, Addison Wesley, August 1995. ISBN 0-89791-701-4.

[28] M.D. McCool. Private communication, September 1996.

[29] D.P. Mitchell and A.N. Netravali. Reconstruction Filters in Computer Graphics. In John Dill, editor, *Computer Graphics, (SIGGRAPH 88 Conference Proceedings)*, volume 22, pages 221-228. ACM SIGGRAPH, August 1988. ISBN 0-89791-275-6.

[30] A.C. Naiman. Grayscale Character Generator and Method. *United States Patent* 4,851,825, 25 July 1989.

[31] A.C. Naiman and A. Fournier. Rectangular Convolution for Fast Filtering of Characters. In Maureen C. Stone, editor, *Computer Graphics, (SIGGRAPH 87 Conference Proceedings)*, volume 21, pages 233-242. ACM SIGGRAPH, July 1987. ISBN 0-89791-227-6.

[32] M.L.V. Pitteway and E. Banissi. Soft Edging Fonts. In *Proceedings of Computer Graphics 87*, pages 133-154. Online Publications, 1987.

[33] M.L.V. Pitteway and D. Watkinson. Bresenham's Algorithm with Grey Scale. *Communications of the ACM*, 23(11): 625-626, November 1980. ISSN 0001-0782.

[34] I.C. Walker. Video Image Creation. *U.K. Patent* GB 2,089,625B, published 25 September 1985.

[35] J.E. Warnock. The Display of Characters Using Gray Level Sample Arrays. In James J. Thomas, editor, *Computer Graphics, (SIGGRAPH 80 Conference Proceedings)*, volume 14, pages 302-307. ACM SIGGRAPH, July 1980. ISBN 0-89791-021-4.

[36] J.T. Whitted. Anti-Aliased Line Drawing Using Brush Extrusion. In Peter Tanner, editor, *Computer Graphics, (SIGGRAPH 83 Conference Proceedings)*, volume 17, pages 151-156. ACM SIGGRAPH, July 1983. ISBN 0-89791-109-1.

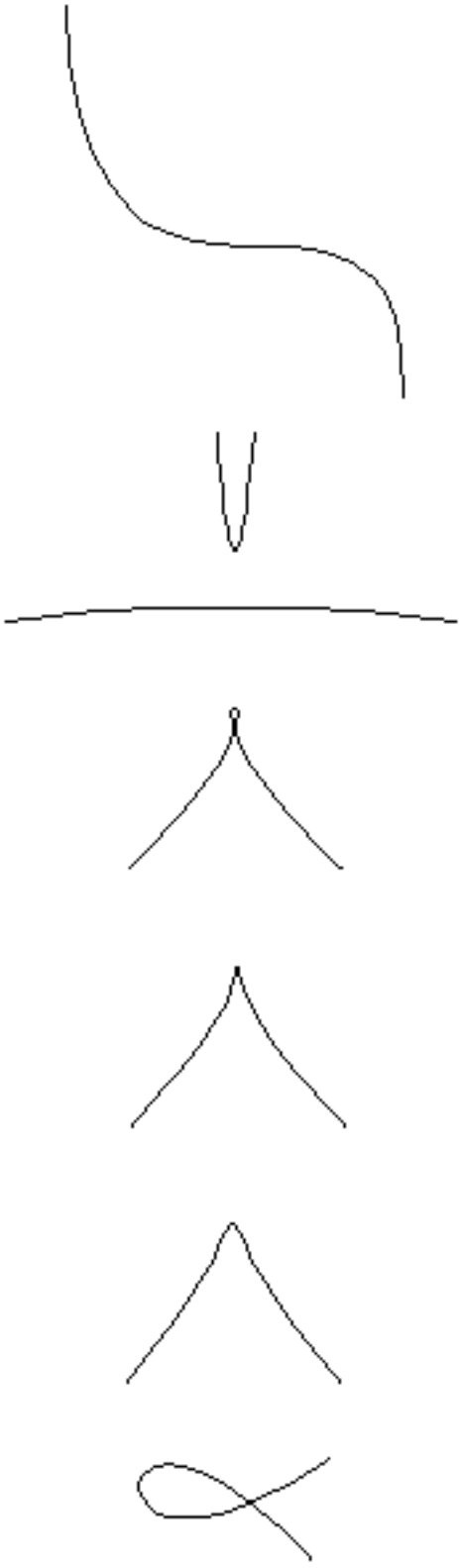


Plate 1: Aliased Test Curves

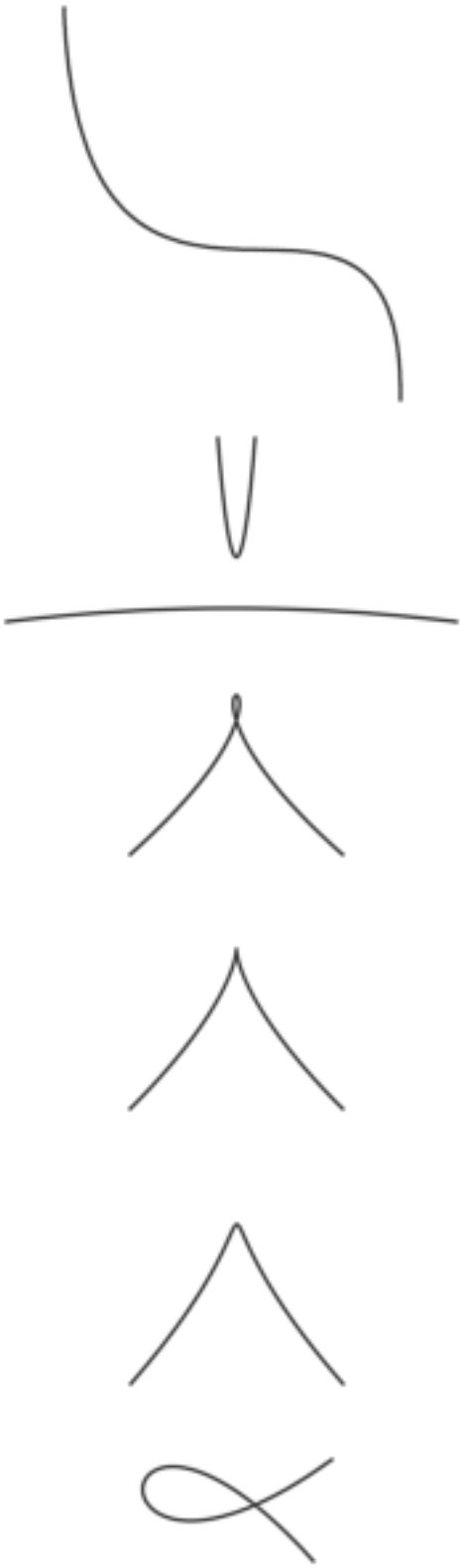


Plate 2: Anti-aliased Test Curves

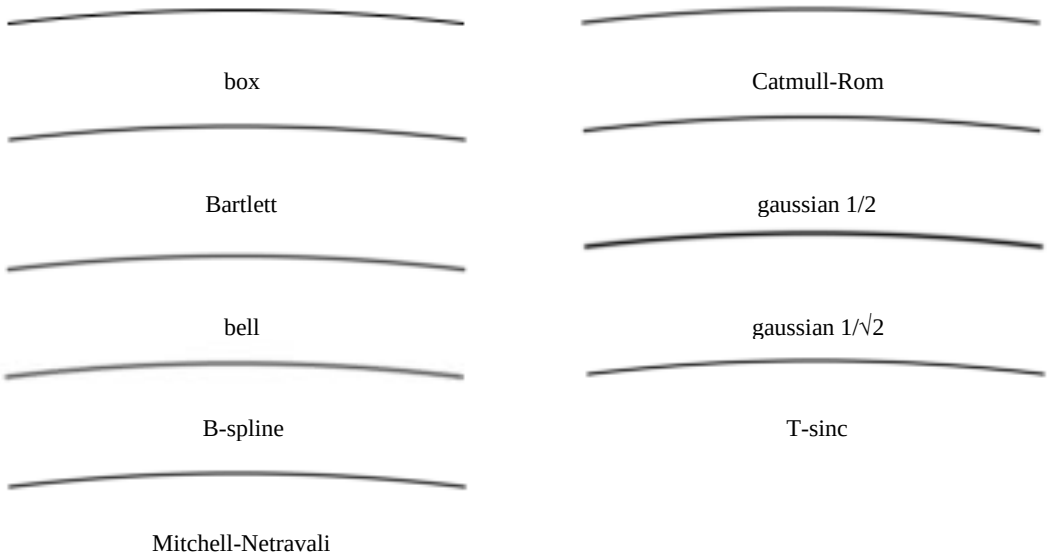


Plate 3: Parabola rendered with a variety of filters.

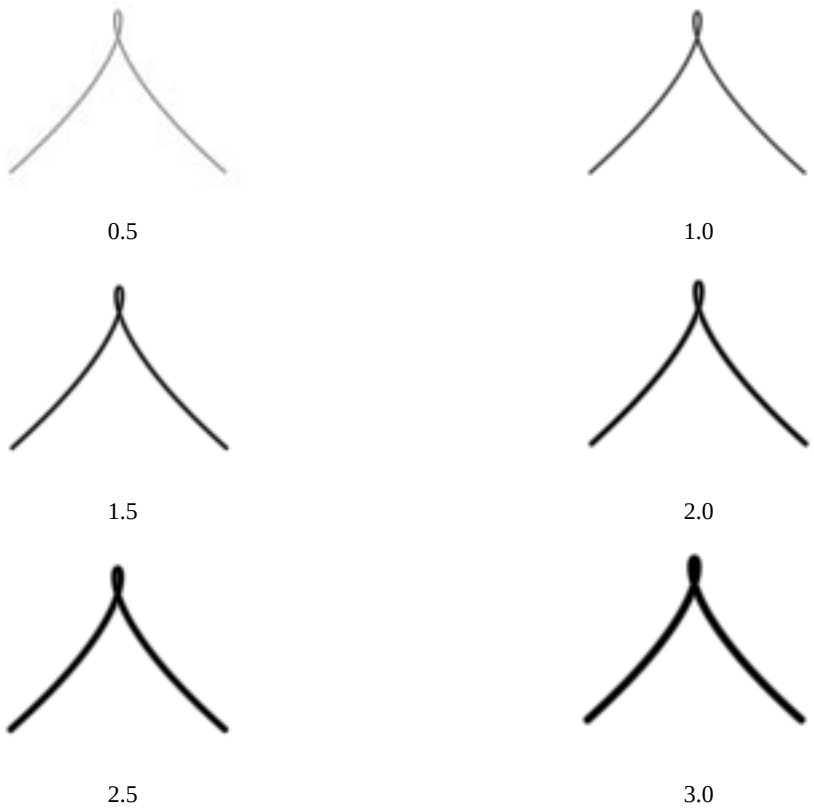
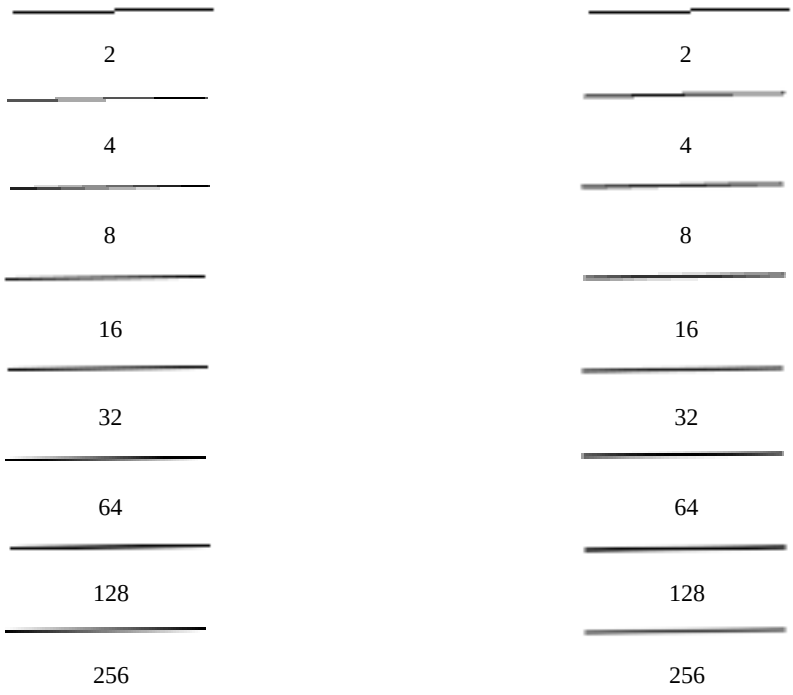


Plate 4: Varying curve width (in pixel units) for a cubic with a small loop near a cusp.



(a) box filter **(b) T-sinc filter**

Plate 5: Varying number of grey levels.