Rasterization of Nonparametric Curves

John D. Hobby

Abstract

We examine a class of algorithms for rasterizing algebraic curves based on an implicit form that can be evaluated cheaply in integer arithmetic using finite differences. These algorithms run fast and produce "optimal" digital output, where previously known algorithms have had serious limitations. We extend previous work on conic sections to the cubic and higher order curves, and we solve an important undersampling problem.

Catagories and Subject Descripters: I.3.3 [**Computer Graphics**]: Picture/Image Generation—Display Algorithms

General Terms: Algorithms, Verification

Additional Key Words and Phrases Scan conversion, algebraic curves, rasterization

Rasterization of Nonparametric Curves

John D. Hobby

1. Introduction

A basic problem in raster graphics and computer typesetting is how to convert from spline curves to discrete data appropriate for raster devices. Typically, graphical objects are represented internally via collections of curves from some fixed family, and the discrete representation is an array of pixels. There are many different versions of the rasterization problem, but all of them involve taking a given curve and determining its position relative to nearby pixels. For this reason, many algorithms can be adapted to a wide range of rasterization problems and it is best to take as general an approach as possible.

We introduce a "canonical" definition of rasterization to serve as a common denominator for the different versions of the rasterization problem. The canonical rasterization is appropriate for spline-bounded regions and can also be applied to curve drawing applications. The precise definition is given shortly, but it is based on the idea that the rasterization of a region is the set of pixels whose centers lie in the region. The canonical rasterization of a curve C is essentially a dividing line that determines which pixel centers lie to the left of C and which lie to the right. All algorithms discussed here are stated in terms of canonical rasterization, but with simple program transformations, the algorithms can equally well be regarded as "optimal" curve-drawing algorithms: all that is required is to work with a shifted copy of C as explained in [7].

The basic ideas behind "optimal" curve-drawing are due to Freeman who first proposed a precise definition for the set of pixels to be computed when rasterizing a curve [2, 3]. Rasterized curves computed according to a precise rule such as Freeman's tend to be superior in terms of smoothness and uniformity of width. They are sometimes referred to as "optimal" because they minimize the maximum deviation between the input and the rasterized output.

Because of the previously mentioned program transformations, any curve drawing algorithm that satisfies appropriate optimality conditions can be viewed as a solution to the canonical rasterization problem. The optimality conditions severely restrict the class of available algorithms, but Knuth gives one such algorithm for parametric cubic curves [9, Part 19]. The reason for the sparsity of appropriate algorithms may be because the parametric form is best suited for generating points on the curve, while the optimality conditions require testing specific points such as pixel centers to see which side of the curve they lie on.

One curve representation that does lend itself to checking pixel centers against the curve is the implicit form $F(x, y) = 0$. This leads to a well-known but often carelessly used class of algorithms that generalize some work by Pitteway [11]. The basic idea is that if F is a polynomial of low degree, it can be evaluated cheaply in integer arithmetic using a system of finite differences. This paper shows how to evaluate F at a sequence of adjacent pixel centers and use the results to determine the rasterization. One major advantage of this approach is inner loop speed: for cubic curves, six additions suffice to evaluate F and update the finite differences.

For comparison, Knuth's algorithm works by repeatedly subdividing the curve, and does at least 29 arithmetic operations per subdivision. The number of subdivisions depends on the slope of the curve and other parameters, but there are typically about as many subdivision steps as pixel centers examined by the implicit algorithms. The overall inner-loop speed is quite reasonable but not as good as with the implicit algorithm for cubic curves.

Of course Knuth's algorithm is not directly comparable to the implicit method, since Knuth uses parametric polynomial cubics while the cubic version of the implicit algorithm uses the broader class of algebraic cubic curves. Cubic curves are often given in parametric form, in which case they have to be converted to implicit form before the implicit method can be applied. Since the conversion process entails a significant amount of overhead and can cause numerical problems, the implicit method is best suited to problems where the implicit form is readily available or needed for other reasons. On the other hand, Pratt has found it practical to convert conic sections from parametric to implicit form and then use the implicit form for rasterization [12].

A good discussion of the relationship between the implicit and parametric forms and how to convert between them can be found in Sederberg's thesis [13]. Implicitization algorithms for cubic curves appear in Patterson [10] and Sederberg et. al. [14]. See [6] for an even more practical version that avoids the need for exact rational arithmetic.

This work appeared in preliminary form in [4]. It addresses two problems that have plagued previous work on implicit rasterization algorithms: (1) Previous work has been limited to conic sections, while many applications require curves of degree three or higher. (2) The common practice of simply testing the sign of $F(x, y)$ at pixel centers is susceptible to undersampling problems that have been successfully addressed by Pratt for conic sections [12]. These ideas are dealt with in this paper as follows: Section 2 introduces the basic algorithm; Section 3 deals with problem (2); Section 4 explains how to initialize the finite differences; and Section 5 gives some concluding remarks.

2. The Basic Algorithm

The algorithm is based on an implicit form $F(x, y) = 0$ for a curve C, where F is a polynomial of moderate degree in x and y. The object is to find the canonical rasterization of C. As we shall see shortly, this is essentially a dividing line that determines which pixel centers lie on one side of the curve and which lie on the other side. The necessary information can be obtained by generating a sequence of pixel centers (m_1, n_1), (m_2, n_2), $\ldots$ and testing them by evaluating $F(m_p, n_p)$ for each p.

Assume that F is integer-valued for integer x and y, and choose the coordinate system so that pixel centers have integer coordinates and adjacent pixels are one unit apart. Then the necessary evaluations of $F(m_p, n_p)$ can be accomplished via a system of finite differences as follows: if F has total degree d, we maintain $\binom{d+1}{2}$ integer-valued registers f_{00}, f_{10}, f_{01}, f_{20}, f_{11}, f_{02}, $\ldots$ where we always have $f_{00} = F(m_p, n_p)$ for some pixel center (m_p, n_p).

Since the finite differences vary as we go from one (m_p, n_p) to the next, it is convenient to introduce polynomials F_{kl}, and maintain the invariant that

$$f_{kl} = F_{kl}(m_p, n_p) \text{ for } 0 \leq k + l \leq d \tag{1}$$

when we are ready to evaluate $F(m_p, n_p)$. The invariant is easily maintained if we let $F_{00} = F$ be the implicit polynomial for C and define F_{kl} recursively so that

$$F_{k+1,l}(x, y) = F_{kl}(x + 1, y) - F_{kl}(x, y), \tag{2}$$
$$F_{k,l+1}(x, y) = F_{k,l}(x, y + 1) - F_{kl}(x, y), \tag{3}$$

for $k, l \geq 0$. When (1) holds we say that the difference registers are *valid for the point* (m_p, n_p).

Figure 1 gives routines for computing registers valid at $(m_p + 1, n_p)$ or at $(m_p, n_p + 1)$ given registers valid at (m_p, n_p). In an actual implementation, d would be a small constant, so the each procedure would reduce to $\binom{d+1}{2}$ assignment statements. Thus there are three additions per pixel center examined when $d = 2$, and six additions per pixel center when $d = 3$.

2.1. Canonical Rasterization

Before giving the rasterization routine, we need a precise definition for the canonical rasterization that is supposed to be computed. As can be seen from Figure 2a, the canonical rasterization

procedure $move\_right$;	**procedure** $move\_up$;
for $i \leftarrow 0, 1, \ldots, d-1$	**for** $i \leftarrow 0, 1, \ldots, d-1$
for $j \leftarrow 0, 1, \ldots, d-i-1$	**for** $j \leftarrow 0, 1, \ldots, d-i-1$
$f_{ij} \leftarrow f_{ij} + f_{i+1,j}$;	$f_{ij} \leftarrow f_{ij} + f_{i,j+1}$;

Figure 1: Routines for updating the difference registers when (m_{p+1}, n_{p+1}) is $(m_p{+}1, n_p)$ or $(m_p, n_p{+}1)$.

of a curve C is essentially an approximation to C that is restricted to a square grid appropriate for delimiting sets of pixels. Thus the canonical rasterization of the closed curve shown in the figure bounds the set of pixels whose centers lie in the region bounded by C. (A formal statement and proof of this property appears in [7].)

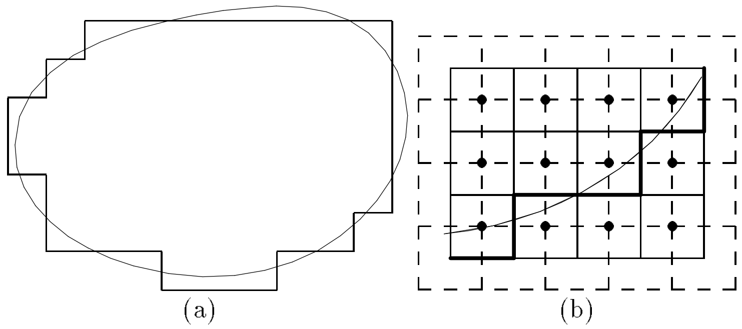


Figure 2: (a) A closed curve and its canonical rasterization. (b) A curve and its canonical rasterization with pixel centers indicated by heavy dots, pixel squares delimited by solid lines, and squares $\Psi(m,n)$ delimited by dashed lines. The heavy line is the canonical rasterization.

Let us assign integer coordinates to pixel centers and model pixels abstractly as unit squares. The canonical rasterization is basically an approximation to the given curve restricted to the boundaries of pixel squares. It is defined in terms of a function

$$\rho(x,y) = \left(\lceil x \rceil - \tfrac{1}{2}, \lfloor y \rfloor + \tfrac{1}{2}\right)$$

that maps points to the nearest pixel *corner*. Since pixel centers are equidistant from four pixel corners, there is a tie in that case and the function is designed to map pixel centers to the upper-left corner. (This tie-breaking rule necessitates the asymmetrical treatment of x and y in the definition of ρ.)

By considering the curve C as an ordered list of points rather than just a point set, we can apply ρ to each point on C and obtain an ordered list of pixel corners

$$(m_0 - \tfrac{1}{2}, n_0 - \tfrac{1}{2}),\ (m_1 - \tfrac{1}{2}, n_1 - \tfrac{1}{2}),\ (m_2 - \tfrac{1}{2}, n_2 - \tfrac{1}{2}),\ \ldots, (m_k - \tfrac{1}{2}, n_k - \tfrac{1}{2}). \tag{4}$$

If for each $i < k$, the difference vector $(m_{i+1}, n_{i+1}) - (m_i, n_i)$ is either $(0, \pm 1)$ or $(\pm 1, 0)$, the canonical rasterization of C is the polygonal line joining each point $(m_i - \frac{1}{2}, n_i - \frac{1}{2})$ in order.

The heavy line that follows pixel edges in Figure 2b is an example of a canonical rasterization. Since the set of all points (x,y) such that $\rho(x,y) = (m - \frac{1}{2}, n - \frac{1}{2})$ is a square of the form

$$\Psi(m,n) = \{(x,y) \mid m-1 < x \le m \text{ and } n-1 \le y < n\}$$

as illustrated in Figure 3, the canonical rasterization depends on the sequence of squares $\Psi(m,n)$ that C passes through.

A consequence of the treatment of the boundary of $Psi(m,n)$ illustrated in Figure 3 is that the plane is divided into square regions $\Psi(m,n)$, and where four of these regions meet at a corner,

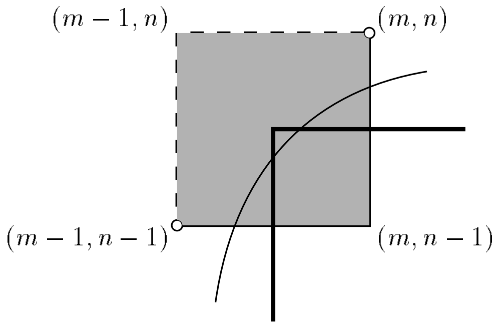


Figure 3: The region $\Psi(m,n)$ with a portion of C and the canonical rasterization passing through it. Portions of the boundary included in $\Psi(m,n)$ are shown as solid lines and other parts of the boundary are indicated by dashed lines. Corners of the region are labelled with their coordinates.

that corner is assigned to the square that is immediately above and to the left. Thus a curve C of positive slope cannot pass from $\Psi(m,n)$ to $\Psi(m+1,n+1)$ without passing through one of the orthogonally adjacent squares $\Psi(m+1,n)$ or $\Psi(m,n+1)$. In other words, the function $\rho(x,y)$ that generates (4) as (x,y) moves along C cannot jump directly $(m-\frac{1}{2}, n-half)$ to $(m+\frac{1}{2}, n+half)$.

The orthogonal adjacency of successive $\rho(x,y)$ values can only break down when C, with non-positive slope, passes through a pixel center. Then we can have $(m_{i+1}, n_{i+1}) - (m_i, n_i) = \pm(1,-1)$, in which case we disambiguate by inserting

$$\left(\min(m_i, m_{i+1}) - \tfrac{1}{2},\ \min(n_i, n_{i+1} - \tfrac{1}{2})\right)$$

after $(m_i - \frac{1}{2}, n_i - \frac{1}{2})$. (See [7] for more details.)

2.2. The Rasterization Routine

When actually computing the rasterization, it is convenient to break up the curve $F(x,y)=0$ by finding vertical and horizontal extrema. For simplicity, we deal only with the case where x and y are nondecreasing on C and C is a *first quadrant curve.* (Other quadrants can be handled by rotation and slight changes in the tie-breaking rules).

Given a first quadrant curve C that begins at (X_a, Y_a) and ends at (X_b, Y_b), we initialize $(m,n) \leftarrow \rho(X_a, Y_a) + (\frac{1}{2}, \frac{1}{2})$ and then examine $F(m,n)$ to determine whether C enters $\Psi(m+1,n)$ or $\Psi(m,n+1)$ when it leaves $\Psi(m,n)$. The algorithm in Figure 4 uses this idea to compute the canonical rasterization of C. Thus the output is not a sequence of pixels but rather the vertices of a polygonal line that passes between pixels. It is a relatively simple matter to use this output to determine the pixels whose centers lie on the "interior" side of C. Alternatively, the output can be used with the techniques of [5] to find a rasterized version of C one or more pixels wide.

The statements "**if** (m,n) is above C" and "Make the registers valid for (m,n)" in Figure 4 are discussed in Sections 3 and 4.

Depending on how the output is to be used, the subtractions in the output statement can probably be avoided in practice. Additionally, one comparison could be saved in the inner loop by only testing m against m' when m is updated and similarly reducing the comparisons between n and n'.

3. The Undersampling Problem

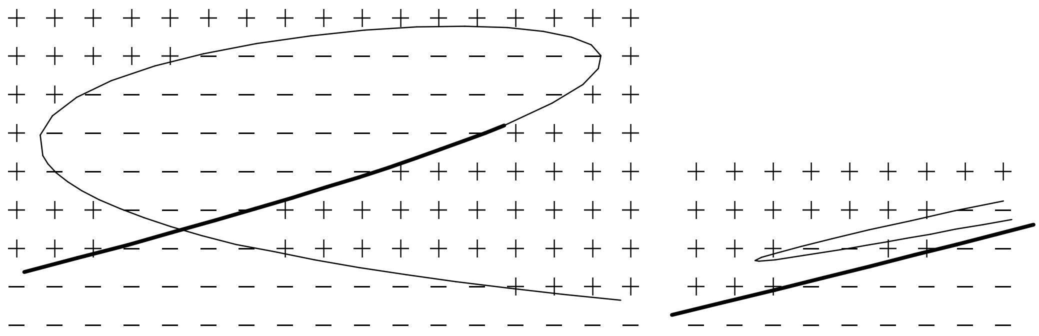
The undersampling problem is that a curve $F(x,y)=0$ cannot necessarily be located reliably by sampling the sign of $F(m,n)$ at integer m and n. This was recognized by Pitteway in 1967, but it was only recently solved by Pratt [12] and Van Aken and Novak [15] for the special case where F

procedure *quadrant1*;
$(m, n) \leftarrow \rho(X_a, Y_a) + (\frac{1}{2}, \frac{1}{2})$;
$(m', n') \leftarrow \rho(X_b, Y_b) + (\frac{1}{2}, \frac{1}{2})$;
Make the registers valid for (m, n);
while $(m \neq m'$ and $n \neq n')$
  { output (m, n);
    if (m, n) is above C
    then { $m \leftarrow m + 1$; *move_right* }
    else { $n \leftarrow n + 1$; *move_up* }
  }
output $(m - \frac{1}{2}, n - \frac{1}{2})$;
if $m \neq m'$ **then** output $(m + 1, n')$, $(m + 2, n')$, $\ldots$, (m', n')
else if $n \neq n'$ **then** output $(m', n + 1)$, $(m', n + 2)$, $\ldots$, (m', n');

Figure 4: A procedure for rasterizing a first quadrant curve starting at (X_a, Y_a) and ending at (X_b, Y_b).

has total degree 2. The problem arises in the implementation of the test "**if** (m, n) is above C." It is convenient to test the sign of $F(m, n)$ (by testing the sign of f_{00}), but as Figures 5a and 5b show, this is not always sufficient. Each figure shows a cubic curve C together with sign information for a corresponding polynomial function F. The curved lines give the zeros of F, and the thick parts locate the original curve C. The sign of F at pixel centers near C is given by $+$ and $-$ signs centered on the appropriate coordinates.

When rasterizing the curve shown in Figure 5a, the function F is initially positive above the curve and negative below it, but the relationship is reversed after the cross-over point. It would be especially difficult to implement the aboveness test near the crossing point where the sign change disappears. In any case, the situation is even more hopeless in Figure 5b where sampling the sign of F at pixel centers gives little or no indication of the potential confusion.



(a) (b)

Figure 5: Cubic curves $F(x, y) = 0$ with the sign of $F(x, y)$ for pixel centers (x, y).

In his paper [12] on conic sections, Pratt mentions that confusion can be avoided by testing the sign of $\frac{\partial F}{\partial y}$ in addition to the sign of F. To generalize this to the case where F has total degree $d > 2$, we simply use the registers f_{02}, f_{03}, f_{04}, $\ldots$ to obtain higher order derivatives. Assuming that the difference registers are valid for a pixel center (m_p, n_p), the partial derivatives determine the function $F|_{x=m_p}$ obtained by restricting F to the line $x = m_p$. The test required in *quadrant1* reduces to comparing n_p to the roots of $F|_{x=m_p}$, given the derivatives at n_p.

Letting $\phi(y) = F|_{x=m_p}$, we evaluate the sequence of derivatives $\phi(n_p)$, $\phi'(n_p)$, $\phi''(n_p)$, $\ldots$, $\phi^{(d)}(n_p)$, and let $r_d(\phi, n_p)$ be the number of sign reversals omitting any zeros in the sequence. If roots(ϕ, y)

is the number of roots of ϕ that are greater than y, Budan's theorem states that

$$\text{roots}(\phi, y) - \text{roots}(\phi, y') \le r_d(\phi, y) - r_d(\phi, y') \quad \text{when } y' \ge y.$$

(See Borofsky [1]). Thus we can test $\text{roots}(\phi, n_p)$ against any desired threshold by testing $r_d(\phi, n_p)$ against a (possibly different) threshold.

Since one of the roots of $\phi(y)$ corresponds to the point where the desired curve intersects $x = m_p$, a test of $\text{roots}(\phi, n_p)$ against an appropriate threshold determines whether the y in question is greater than n_p. If not, then (m_p, n_p) is on or above the curve.

To determine whether (m_p, n_p) is strictly above the curve, let $\theta(y) = \phi(-y)$ and use $\tilde{r}_d(\phi, n_p) = r_d(\theta, -n_p)$ to test $\text{roots}(\theta, -n_p)$. Thus $\tilde{r}_d(\phi, n_p)$ is the number of sign reversals after eliding zeros in the sequence $\phi(n_p)$, $-\phi'(n_p)$, $\phi''(n_p)$, ..., $(-1)^d\phi^{(d)}(n_p)$.

The aboveness test depends on knowing the threshold $k = \tilde{r}_d(\phi, y_p)$, where $\phi = F|_{x=m_p}$ and (m_p, y_p) is on the desired curve C. To determine whether $n_p > y_p$ without computing y_p, we test if $\tilde{r}_d(\phi, n_p) > k$:

$$\begin{aligned}
n_p > y_p &\Longrightarrow 0 < \text{roots}(\theta, -n_p) - \text{roots}(\theta, -y_p) \le r_d(\theta, -n_p) - r_d(\theta, -y_p) = \tilde{r}_d(\phi, n_p) - k \\
n_p \le y_p &\Longrightarrow 0 \le \text{roots}(\theta, -y_p) - \text{roots}(\theta, -n_p) \le r_d(\theta, -y_p) - r_d(\theta, -n_p) = k - \tilde{r}_d(\phi, n_p)
\end{aligned}$$

3.1. Implementing the Aboveness Test

In order to evaluate $\tilde{r}_d(F|_{x=m_p}, n_p)$ for pixel centers (m_p, n_p), the *quadrant1* procedure must be able to compute the partial derivatives of F given only the difference registers f_{00}, f_{01}, f_{02}, ..., f_{0d} valid for (m_p, n_p).

We can obtain an expression for $F(m_p + \bar{x}, n_p + \bar{y})$ in terms of the difference registers by using a simple inductive argument based on the formulas (2) and (3) from Section 2 to verify for nonnegative integers $\bar{x}$ and $\bar{y}$, that

$$F_{kl}(m_p + \bar{x}, n_p + \bar{y}) = \sum_{\substack{i \ge k \\ j \ge l \\ i+j \le d}} \binom{\bar{x}}{i-k}\binom{\bar{y}}{j-l} F_{ij}(m_p, n_p), \qquad \text{for } k + l \le d. \tag{5}$$

Since

$$\binom{\bar{x}}{i-k} = \frac{\bar{x}(\bar{x}-1)(\bar{x}-2)\cdots(\bar{x}+1+k-i)}{(i-k)!} \quad \text{and} \quad \binom{\bar{y}}{j-l} = \frac{\bar{y}(\bar{y}-1)(\bar{y}-2)\cdots(\bar{y}+1+l-j)}{(j-l)!}$$

can be viewed as polynomials in $\bar{x}$ and $\bar{y}$, (5) can be viewed as a polynomial equation that holds for all real $\bar{x}$ and $\bar{y}$.

Setting k, l and $\bar{x}$ to zero and noting that $F_{ij}(m_p, n_p) = f_{ij}$ when the registers are valid for (m_p, n_p), we obtain

$$F(m_p, n_p + \bar{y}) = \sum_{i+j \le d} \binom{0}{i}\binom{\bar{y}}{j} f_{ij} = \sum_{0 \le j \le d} \binom{\bar{y}}{j} f_{0j}.$$

The next step is to substitute

$$\binom{\bar{y}}{j} = \sum_{0 \le i' \le j} (-1)^{j-i'} \left[{j \atop i'}\right] \frac{\bar{y}^{i'}}{j!},$$

where $\left[{j \atop i'}\right]$ denotes the Stirling number of the first kind as given by Knuth [8]:

$$\left[{j \atop i'}\right] = (j-1)\left[{j-1 \atop i'}\right] + \left[{j-1 \atop i'-1}\right]; \qquad \left[{0 \atop i'}\right] = \begin{cases} 1 & \text{if } i' = 0; \\ 0 & \text{otherwise.} \end{cases}$$

The result of the substitution is

$$F(m_p, n_p + \bar{y}) = \sum_{0 \le i' \le j \le d} (-1)^{j-i'} \begin{bmatrix} j \\ i' \end{bmatrix} \frac{\bar{y}^{i'}}{j!} f_{0j},$$

and differentiating q times with respect to $\bar{y}$ yields

$$\frac{\partial^q F(m_p, n_p)}{\partial y^q} = q! \sum_{q \le j \le d} \frac{(-1)^{j-q}}{j!} \begin{bmatrix} j \\ q \end{bmatrix} f_{0j}. \tag{6}$$

In other words $\frac{\partial^q F}{\partial y^q}$ is a linear combination of at most $d+1-q$ of the difference registers with simple rational coefficients.

In the important case $d = 3$, (6) becomes

$$\begin{aligned} F(m_p, n_p) &= f_{00}, \\ \frac{\partial F(m_p, n_p)}{\partial y} &= f_{01} - \tfrac{1}{2} f_{02} + \tfrac{1}{3} f_{03}, \\ \frac{\partial^2 F(m_p, n_p)}{\partial y^2} &= f_{02} - f_{03}, \\ \frac{\partial^3 F(m_p, n_p)}{\partial y^3} &= f_{03}. \end{aligned}$$

Thus $\frac{\partial^2 F}{\partial y^2} > 0$ when $f_{02} > f_{03}$ and $\frac{\partial F}{\partial y} > 0$ when $f_{01} + \frac{2}{3} f_{03} > f_{02} - f_{01}$. Since the difference registers are integer-valued, $\frac{\partial F}{\partial y} > 0$ when $f_{01} + \lceil \frac{2}{3} f_{03} \rceil > f_{02} - f_{01}$, $\frac{\partial F}{\partial y} < 0$ when $f_{01} + \lfloor \frac{2}{3} f_{03} \rfloor < f_{02} - f_{01}$, and $\frac{\partial F}{\partial y} = 0$ only if f_{03} is divisible by 3.

We can now test the sign of F, $\frac{\partial F}{\partial y}$, $\frac{\partial^2 F}{\partial y^2}$, and $\frac{\partial^3 F}{\partial y^3}$ and implement the aboveness test when $d = 3$, assuming without loss of generality that F has been chosen so that the constant f_{03} is nonnegative. First consider the case when f_{03} is positive and not divisible by three so that $\frac{\partial F}{\partial y}$ is never zero. Then the test "**if** (m_p, n_p) is above C" can be implemented as follows when $\lfloor \frac{2}{3} f_{03} \rfloor$ and $\lceil \frac{2}{3} f_{03} \rceil$ are precomputed:

$$\begin{aligned} \tilde{r}_3(F|_{x=m_p}, n_p) > 2 &\iff f_{00} > 0 \ \text{ and } \ \lceil \tfrac{2}{3} f_{03} \rceil + f_{01} > f_{02} - f_{01} \ \text{ and } \ f_{02} > f_{03}, \\ \tilde{r}_3(F|_{x=m_p}, n_p) > 1 &\iff \text{if } \ \lceil \tfrac{2}{3} f_{03} \rceil + f_{01} > f_{02} - f_{01} \ \text{ then } \ f_{02} > f_{03} \text{ else } \ f_{00} < 0, \\ \tilde{r}_3(F|_{x=m_p}, n_p) > 0 &\iff f_{00} > 0 \ \text{ or } \ \lfloor \tfrac{2}{3} f_{03} \rfloor + f_{01} < f_{02} - f_{01} \ \text{ or } \ f_{02} > f_{03}. \end{aligned}$$

For instance the third line says that there are sign reversals in the sequence F, $-\frac{\partial F}{\partial y}$, $\frac{\partial^2 F}{\partial y^2}$, and $-\frac{\partial^3 F}{\partial y^3}$ whenever $F > 0$, $-\frac{\partial F}{\partial y} > 0$, or $\frac{\partial^2 F}{\partial y^2} > 0$.

The same tests work when f_{03} is a positive multiple of three, except that

$$\begin{aligned} \tilde{r}_3(F|_{x=m_p}, n_p) > 1 \iff &\text{if } A = B \text{ then } f_{00} < 0 \text{ and } f_{02} > f_{03} \\ &\text{else if } A > B \text{ then } f_{02} > f_{03} \text{ else } f_{00} < 0, \end{aligned}$$

where $A = \frac{2}{3} f_{03} + f_{01}$ and $B = f_{02} - f_{01}$. In any case, the aboveness test for cubic curves requires two additions and three comparisons.

3.2. Finding Thresholds

Now that we know an efficient way of testing $\tilde{r}_d(F|_{x=m_p}, n_p)$ against an integer threshold, the remaining question is how to choose the threshold. Presumably we have difference registers that define the function F and we know the endpoints of the desired curve C. Of course some kind of additional information may be necessary in order to determine C unambiguously when the endpoints are given approximately or when they lie at places where $F(x, y) = 0$ crosses itself. Such crossings are particularly hard to deal with in the absence of additional information since it can be difficult to decide which branch of $F(x, y) = 0$ is desired.

Rather than go into great detail about how to locate crossings and decide which branch of $F(x, y) = 0$ is desired there, let us assume that any possible crossing points are given as part of the problem instance. Such crossings commonly occur when C has a parametric form from which the implicit form is derived. If this derivation is done as described in [6] and [14], the crossing point location is found as a side effect.

The problem is to find $\tilde{r}_d(F|_{x=m_p}, y)$, where (m_p, y) is on the curve C being rasterized. As a point (x, y) moves along C, the threshold $\tilde{r}_d(F|_{x=x}, y)$ can only change when one of the derivatives $\frac{\partial F}{\partial y}$, $\frac{\partial^2 F}{\partial y^2}$, $\ldots$, $\frac{\partial^{d-1} F}{\partial y^{d-1}}$ changes sign. Thus it suffices to find all such sign changes and use them to subdivide C so that each piece has its own threshold value.

When F has total degree $d = 2$, C is a conic section and $\frac{\partial F}{\partial y}$ has no zero crossings except at horizontal extrema. Thus if C is already subdivided as required by the *quadrant1* procedure, the threshold $\tilde{r}_d(F|_{x=x}, y)$ is constant for (x, y) on C and we need only evaluate it at the starting point.

In the cubic case where $d = 3$, zero crossings of $\frac{\partial F}{\partial y}$ on C occur at horizontal extrema and possibly at a point where $F(x, y) = 0$ crosses itself as in Figure 5a. Thus if the crossing point is given in advance, we need only find where C crosses the line $\frac{\partial^2 F}{\partial y^2} = 0$.

For cubics where $f_{03} \neq 0$, it is easier to find the zeros of $\frac{\partial F}{\partial y}$ on $\frac{\partial^2 F}{\partial y^2} = 0$ and use them to evaluate the range of x-coordinates R^+ where $\frac{\partial F}{\partial y} \geq 0$ on $\frac{\partial^2 F}{\partial y^2} = 0$. Assume without loss of generality that $f_{03} > 0$ so that $F|_{x=m_p}$ is monotone increasing in y when $m_p \in R^+$. Thus we need only test $\tilde{r}_3(F|_{x=m_p}, n_p)$ when $m_p \notin R^+$. We need not find zero crossings of $\frac{\partial^2 F}{\partial y^2}$ as (x, y) moves along C because such zero crossings do not affect $\tilde{r}_3(F|_{x=x}, y)$ for $x \notin R^+$ since $\frac{\partial F}{\partial y}$ and $\frac{\partial^3 F}{\partial y^3}$ have opposite signs when $\frac{\partial^2 F}{\partial y^2} = 0$.

Figure 6 shows a cubic curve where R^+ is the interval $[x_1, x_2]$ and the crossing point occurs at $x = x_3$. These three *critical x-coordinates* determine four different forms of the test "(m_p, n_p) is above C:"

$$\begin{cases} \tilde{r}_3(F|_{x=m_p}, n_p) > 2 & \text{when } m_p \leq x_1, \\ F(m_p, n_p) > 0 & \text{when } x_1 \leq m_p \leq x_2, \\ \tilde{r}_3(F|_{x=m_p}, n_p) > 0 & \text{when } x_2 \leq m_p \leq x_3, \\ \tilde{r}_3(F|_{x=m_p}, n_p) > 1 & \text{when } x_3 \leq m_p. \end{cases}$$

To find threshold values and critical x-coordinates, we need expressions for the partial derivates in terms of difference registers valid at some point (m_p, n_p). Setting $k = l = 0$ and $d = 3$ in (5) and differentiating, we obtain

$$\frac{\partial F(m_p + \bar{x}, n_p + \bar{y})}{\partial x} = f_{10} + \bar{y} f_{11} + \binom{\bar{y}}{2} f_{12} + (\bar{x} - \tfrac{1}{2})(f_{20} + \bar{y} f_{21}) + (\tfrac{1}{2}\bar{x}^2 - \bar{x} + \tfrac{1}{3}) f_{30} \quad (7)$$

$$\frac{\partial F(m_p + \bar{x}, n_p + \bar{y})}{\partial y} = f_{01} + \bar{x} f_{11} + \binom{\bar{x}}{2} f_{21} + (\bar{y} - \tfrac{1}{2})(f_{02} + \bar{x} f_{12}) + (\tfrac{1}{2}\bar{y}^2 - \bar{y} + \tfrac{1}{3}) f_{03} \quad (8)$$

$$\frac{\partial^2 F(m_p + \bar{x}, n_p + \bar{y})}{\partial y^2} = f_{02} + \bar{x} f_{12} + (\bar{y} - 1) f_{03}. \quad (9)$$

To evaluate $\frac{\partial F}{\partial y}$ when $\frac{\partial^2 F}{\partial y^2} = 0$, substitute $\bar{y} = 1 - (f_{02} + \bar{x} f_{12})/f_{03}$ into (8):

$$(f_{01} + \tfrac{1}{2} f_{02} + \tfrac{1}{6} f_{03} - \tfrac{1}{2} f_{02}^2/f_{03}) + (f_{11} + \tfrac{1}{2} f_{12} - \tfrac{1}{2} f_{21} - f_{02} f_{12}/f_{03})\bar{x} + \tfrac{1}{2}(f_{21} - f_{12}^2/f_{03})\bar{x}^2. \quad (10)$$

Thus we obtain critical x-coordinates by adding m_p to each root of (10).

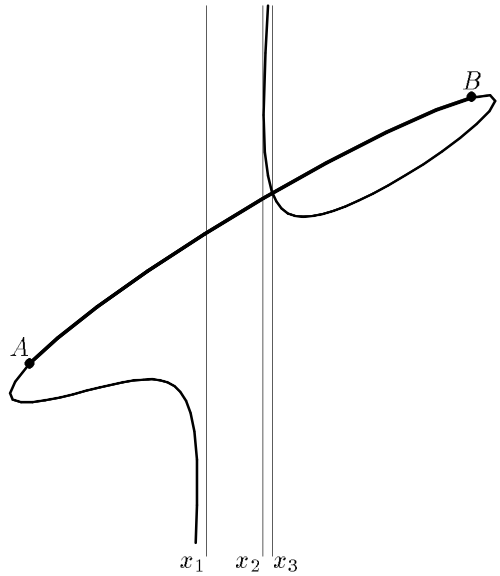


Figure 6: The zeros of $F(x, y) = -4x^3+13x^2y-12xy^2+y^3+30x^2-40xy-20y^2$ and the corresponding critical x coordinates. The first quadrant curve from A to B requires four different versions of the aboveness test.

Once we have the critical x-coordinates, we choose the appropriate form of the aboveness test and run *quadrant1* up to the first critical x-coordinate. Then we choose the form of the aboveness test to use up to the next x-coordinate and repeat. When starting in or entering a region where $x \in R^+$, the aboveness test becomes "$f_{00} > 0$." Otherwise it is "$\tilde{r}_3(F|_{x=m_p}, n_p) > k$," where we use (8) and (9) to evaluate $k = \tilde{r}_3(F|x = x_i, y_i)$ for some convenient (x_i, y_i) on the appropriate part of C assuming that $F = 0$ and $\frac{\partial^3 F}{\partial y^3} > 0$ at (x_i, y_i). We can always choose (x_i, y_i) to be the starting or ending point of C or where C crosses a boundary of a region where $x \in R^+$.

To make this more reliable when the slope is nearly vertical at (x_i, y_i), note that the gradient of F is perpendicular to C and therefore $\frac{\partial F}{\partial x}$ and $\frac{\partial F}{\partial y}$ always have opposite signs when C is a first quadrant curve. Hence we evaluate $\frac{\partial F}{\partial x}$ and $\frac{\partial F}{\partial y}$ at (x_i, y_i) and look at the sign of whichever has the greatest magnitude. If $\frac{\partial F}{\partial y}$ evaluates to zero, or if numerical errors cause a result of the wrong sign, the proper threshold may be obtained by using $-\frac{\partial F}{\partial x}$ in place of $\frac{\partial F}{\partial y}$ when evaluating $\tilde{r}_3(F|_{x=x_i}, y_i)$.

When C leaves a region where $x \in R^+$ at some critical x-coordinate x_i, it is awkward to obtain the corresponding y_i where $F_{x=x_i} = 0$, but we know à priori that $F = 0$, $\frac{\partial F}{\partial y} > 0$ and $\frac{\partial^3 F}{\partial y^3} > 0$ at (x_i, y_i). Thus the threshold $\tilde{r}_3(F|_{x=x_i}, y_i)$ is two or zero depending on whether $\frac{\partial^2 F(x_i, y_i)}{\partial y^2} > 0$. Since $\frac{\partial^2 F}{\partial y^2}|_{x=x_i}$ and $F|_{x=x_i}$ are both monotone increasing, $\frac{\partial^2 F}{\partial y^2}|_{x=x_i} > 0$ when $F|_{x=x_i} = 0$ if and only if $F|_{x=x_i} < 0$ when $\frac{\partial^2 F}{\partial y^2}|_{x=x_i} = 0$; i.e., if the registers are valid for (m_p, n_p), we test if

$$\sum_{i+j \le 3} \binom{\bar{x}}{i} \binom{\bar{y}}{j} f_{ij} < 0,$$

when $\bar{x} = x_i - m_p$ and $\bar{y} = 1 - (f_{02} + \bar{x} f_{12})/f_{03}$.

The overall process of finding threshold values for cubic curves consists of solving (10) and then evaluating some partial derivatives in order to find $\tilde{r}_3(F|_{x=x_i}, y_i)$ at the appropriate (x_i, y_i). In terms of total arithmetic operations, it takes about 17 to find the coefficients of (10) and a few more to solve it. In addition, about 50 operations are required to find each threshold value using the techniques given above. These are likely to be floating point operations that are more expensive than the nine additions and four comparisons per pixel center in the inner loop of *quadrant1*.

4. Initializing the Difference Registers

Just what is meant when the *quadrant1* procedure says "Make the registers valid for (m, n)?" We are given a polynomial of total degree d in x and y expanded about some point (X, Y) and we are to compute a new form expanded about (m, n) and rescale if necessary to obtain integer difference registers. In other words, we choose a scale factor λ and base the difference registers on

$$F(X + \bar{x},\, Y + \bar{y}) = \lambda \sum_{k+l \le d} c_{kl} \bar{x}^k \bar{y}^l = 0. \tag{11}$$

Thus

$$F(m + \bar{x}, n + \bar{y}) = \lambda \sum_{k+l \le d} c'_{kl} \bar{x}^k \bar{y}^l,$$

where

$$c'_{kl} = \sum_{i+j \le d-k-l} \binom{k+i}{k} \binom{l+j}{l} (m - X)^i (n - Y)^j c_{k+i,l+j}. \tag{12}$$

To relate $F(m + \bar{x}, n + \bar{y})$ to the difference registers valid for (m, n), we specialize (5) to

$$F(m + \bar{x}, n + \bar{y}) = \sum_{i+j \le d} \binom{\bar{x}}{i} \binom{\bar{y}}{j} f_{ij}. \tag{13}$$

Then let

$$f_{ij} = \lambda \sum_{k+l \le d} \left\{ {k \atop i} \right\} \left\{ {l \atop j} \right\} i!\, j!\, c'_{kl} \quad \text{for } i + j \le d, \tag{14}$$

where $\left\{ {k \atop i} \right\}$ denotes the Stirling number of the second kind as given by Knuth [8]:

$$\left\{ {k \atop i} \right\} = i \left\{ {k-1 \atop i} \right\} + \left\{ {k-1 \atop i-1} \right\} \qquad \left\{ {0 \atop i} \right\} = \begin{cases} 1 & \text{if } i = 0; \\ 0 & \text{otherwise.} \end{cases}$$

We can verify (14) by substituting it into the right-hand side of (13), giving

$$\begin{aligned} \lambda \sum_{\substack{i+j \le d \\ k+l \le d}} \binom{\bar{x}}{i} \binom{\bar{y}}{j} \left\{ {k \atop i} \right\} \left\{ {l \atop j} \right\} i!\, j!\, c'_{kl} &= \lambda \sum_{k+l \le d} c'_{kl} \left(\sum_i \left\{ {k \atop i} \right\} \binom{x}{i} i! \right) \left(\sum_j \left\{ {l \atop j} \right\} \binom{y}{j} j! \right) \\ &= \lambda \sum_{k+l \le d} c'_{kl} x^k y^l \end{aligned}$$

as required.

For $d = 3$, (14) reduces to

$$\begin{array}{lll} & & f_{30} = 6\lambda c'_{30} \\ f_{00} = \lambda c'_{00} & f_{20} = \lambda(2c'_{20} + 6c'_{30}) & f_{21} = 2\lambda c'_{21} \\ f_{10} = \lambda(c'_{10} + c'_{20} + c'_{30}) & f_{11} = \lambda(c'_{11} + c'_{21} + c'_{12}) & f_{12} = 2\lambda c'_{12} \\ f_{01} = \lambda(c'_{01} + c'_{02} + c'_{03}) & f_{02} = \lambda(2c'_{02} + 6c'_{03}) & f_{03} = 6\lambda c'_{03}. \end{array}$$

To make the registers valid for (m, n), we use (12) to obtain the necessary values for c'_{kl} and plug these into (14).

Some additional issues arise if F is not given in a form that guarantees integer values for integer inputs. First of all, it is necessary to round off the right-hand side of (14) in order to get integer-valued difference registers. The difference registers then refer to a slightly different function $F'(x, y)$ and *quadrant1* computes the rasterization of $F'(x, y) = 0$. The functions F and F' will be quite similar near the initial point (m, n) but may diverge as (x, y) moves away from (m, n). Thus if the curve contains a point where the gradient of F is zero, the initial (m, n) should be as close to that point as possible. For a cubic curve where $F(x, y) = 0$ crosses itself, the gradient is zero there so it is a good idea to divide the curve in half at that point and process each part separately starting at the crossing point.

Choosing the Scale Factor

The purpose of the scale factor λ is to limit the errors produced by rounding the right-hand side of (14) while ensuring register values remain small enough to avoid integer overflow. If $F(x, y)$ has integer coefficients and is known to be small enough to avoid overflow, it is possible to set $\lambda = 1$ and skip the rest of this section. This section applies to those applications where λ must be set so that the register values will be as large as possible without danger of overflow. We give an upper bound that depends on λ for the quantities computed when the *quadrant1* procedure is used to rasterize a first-quadrant curve $F(x, y) = 0$ with starting and ending points (m, n) and (m', n'). (Alternative techniques that may yield better bounds when a parametric form of $F(x, y) = 0$ is known are discussed in [4].)

The algorithm examines pixel centers (m_i, n_i) where $m \le m_i < m'$ and $n \le n_i < n'$, generating registers valid for such points (m_i, n_i). When F has total degree d, the register values are $F_{kl}(m_i, n_i)$ for $0 \le k+l \le d$, where F_{kl} are the finite differences of F as defined in the introduction. Additionally, the aboveness test requires linear combinations of register values. For $d = 3$, these are $\frac{2}{3}F_{03}(m_i, n_i)+F_{01}(m_i, n_i)$ and $F_{02}(m_i, n_i) - F_{01}(m_i, n_i)$.

To get bounds on the above mentioned functions at the relevant pixel centers (m_i, n_i), it suffices to bound the functions on the rectangle $m \le x \le m' - 1$, $n \le y \le n' - 1$. The computations are somewhat simplified if we use the fact that $F_{kl}(m_i, n_i)$ is equal to $\frac{\partial^{k+l}F}{\partial x^k \partial y^l}$ evaluated at some point (x, y) where $m_i < x < m_i + k$ and $n_i < y < m_i + l$. (This follows by repeated application of the Mean Value Theorem.) Thus we can bound F_{kl} by finding the extremes of $\frac{\partial^{k+l}F}{\partial x^k \partial y^l}$ on the rectangle $m \le x \le m' + k - 1$, $n \le y \le n' + l - 1$.

In the special case $k = l = 0$, we can take advantage of the fact that we are trying to track zeros of $F(x, y)$; i.e., we need only consider $F(m_i, n_i)$ for pixel centers (m_i, n_i) where the curve $F(x, y) = 0$ crosses from $\Psi(m_i, n_i)$ to $\Psi(m_i{+}1, n_i)$ or $\Psi(m_i, n_i{+}1)$, where Ψ is as defined in Section 2.1. Thus there exists t in the interval $[0, 1]$ where either $F(m_i - t, n_i) = 0$ or $F(m_i, n_i - t) = 0$. Hence we can bound $|F(m_i, n_i)|$ by maximizing $\left|\frac{\partial F}{\partial x}\right|$ on the line segment $\{\,(m_i - t', n_i) \mid 0 \le t' \le 1\,\}$ and maximizing $\left|\frac{\partial F}{\partial x}\right|$ on $\{\,(m_i, n_i - t') \mid 0 \le t' \le 1\,\}$. In other words, we get bounds of $|F|$ for free by slightly extending the rectangle over which we maximize $\left|\frac{\partial F}{\partial x}\right|$ and $\left|\frac{\partial F}{\partial y}\right|$.

When F has total degree $d = 3$,

$$\begin{aligned} F_{02}(m_i, n_i) - F_{01}(m_i, n_i) &= F_{02}(m_i, n_i) - \big(F_{02}(m_i, n_i - 1) + F_{01}(m_i, n_i - 1)\big) \\ &= F_{03}(m_i, n_i - 1) - F_{01}(m_i, n_i - 1). \end{aligned}$$

Hence each of $|F_{01}|$, $|F_{03}|$, $\left|\frac{2}{3}F_{03} + F_{01}\right|$, and $|F_{02} - F_{01}|$ never gets larger than

$$\left|\frac{\partial^3 F}{\partial y^3}\right| + \max_{\substack{m \le x \le m'-1 \\ n-1 \le y \le n'}} \left|\frac{\partial F}{\partial y}\right|. \tag{15}$$

The overall bound on all quantities computed by *quadrant1* is the maximum of (15), $\left|\frac{\partial^3 F}{\partial x^3}\right|$, $\left|\frac{\partial^3 F}{\partial x^2 \partial y}\right|$, $\left|\frac{\partial^3 F}{\partial x \partial y^2}\right|$, and

$$\max\left(\max_{\substack{m-1 \le x \le m' \\ n \le y \le n'-1}} \left|\frac{\partial F}{\partial x}\right|, \max_{\substack{m \le x \le m'+1 \\ n \le y \le n'-1}} \left|\frac{\partial^2 F}{\partial x^2}\right|, \max_{\substack{m \le x \le m' \\ n \le y \le n'}} \left|\frac{\partial^2 F}{\partial x \partial y}\right|, \max_{\substack{m \le x \le m' \\ n \le y \le n'+1}} \left|\frac{\partial^2 F}{\partial y^2}\right| \right). \tag{16}$$

When F is defined via (11) with $d = 3$, each partial derivative of F is λ times a known polynomial. The third partials are independent of x and y; the second partials are linear functions of x and y that can easily be maximized over the required rectangles; and the first partials are quadratic functions that are only a little harder to maximize. Thus we can find $F_{\max}$ such that the maximum of (15), (16), and the third partials is $\lambda F_{\max}$. Hence if we allow integers of maximum magnitude M, we set $\lambda = M/F_{\max}$.

5. Conclusion

We have introduced general techniques for the rasterization of algebraic curves given in implicit form with particular attention to curves of degree three. The techniques can be applied to parametric curves by first converting them into implicit form as explained in [14] and [6]. The main reason for concentrating on cubic curves is that the degree-two case has already been well studied [12].

Since we compute the rasterization according to the definition given in Section 2.1, our results are mathematically precise. They are compatible with Freeman's rule for generating one-pixel-wide lines as well as the "pixels whose centers lie inside" rule for rasterizing spline-bounded regions. The properties of these rules and the resulting rasterized images have been well studied by works such as [15].

A major contribution of this work is a general solution to the undersampling problem, which in the cubic case involves two extra additions and two extra comparisons per pixel center examined by the *quadrant1* procedure. Of course there is also the overhead required to find critical x-coordinates and threshold values as explained in Section 3.2, but the actual operation counts suggest that this overhead is quite reasonable for cubics.

Alternatively, the overhead disappears if we regard the threshold values as part of the problem instance. This may be reasonable because it avoids certain problems that are especially troubling for curves of degree higher than cubic: The integer difference registers define a precise function $F(x, y)$, but it might not be clear without the thresholds which solutions to $F(x, y) = 0$ give the desired curve. After all, the given starting and ending points may be rational or floating-point approximations that do not lie precisely on $F(x, y) = 0$. In fact, the curve could begin at one crossing point and end at another.

The worst difficulties are avoided for cubic curves where it appears possible to derive reasonable thresholds from approximate data. When used with floating-point arithmetic, the techniques of Section 3.2 are intended to "do the right thing" except in degenerate cases where it makes very little difference. This is borne out by practical experience, but it would be difficult to prove.

The problem of approximate data is compounded when it is necessary to round the initial register values to integers. An implicit form with floating-point coefficients is inherently imprecise, but the errors may be magnified when we enforce the same standard of absolute accuracy on all registers regardless of the magnitude of their initial values.

One way to control such errors is to apply a special linear transformation that maps pixel centers to pixel centers. Such "unit determinant transformations" are discussed in [4]. They work by modifying the *quadrant1* procedure to use difference registers based on the transformed coordinate system while computing the canonical rasterization in the original coordinate system. This eases the problems caused by rounding initial register values.

In any case the initial register values and threshold choices determine a precisely defined curve, and the algorithm computes the canonical rasterization of this curve with no errors of any kind. Rounding the initial register values can cause the curve actually rasterized to differ from the desired curve, but the difference is usually fairly small; e.g., errors on the order of a few hundred thousandths of a pixel are typical for curves 100 pixels long computed with 32-bit arithmetic.

References

[1] S. Borofsky. *Elementary Theory of Equations.* Macmillan, New York, 1950.

[2] H. Freeman. On the encoding of arbitrary geometric configurations. *IRE Transactions on Electronic Computers*, EC-10(2):260–268, June 1961.

[3] H. Freeman and J. M. Glass. On the quantization of line-drawing data. *IEEE Transactions on Systems Science and Cybernetics*, 5(1):70–79, January 1969.

[4] John D. Hobby. Non-parametric digitization algorithms. Computing Science Technical Report no. 125, AT&T Bell Laboratories, Murray Hill, New Jersey, 1986.

[5] John D. Hobby. Rasterizing curves of constant width. *Journal of the ACM*, 36(2):209–229, April 1989.

[6] John D. Hobby. Numerically stable implicitization of cubic curves. *ACM Transactions on Graphics*, 10(3):255–296, July 1991.

[7] John Douglas Hobby. *Digitized Brush Trajectories.* PhD thesis, Dept. of Computer Science, Stanford University, 1985.

[8] D. E. Knuth. *The Art of Computer Programming*, volume 1. Addison Wesley, Reading, Massachusetts, 1973.

[9] D. E. Knuth. *METAFONT the Program.* Addison Wesley, Reading, Massachusetts, 1986. Volume D of *Computers and Typesetting.*

[10] R. R. Patterson. Parametric cubics as algebraic curves. *Computer-Aided Geometric Design*, 5(2):139–159, July 1988.

[11] M. L. V. Pitteway. Algorithm for drawing ellipses or hyperbolæ with a digital plotter. *Computer Journal*, 10(3):282–289, November 1967.

[12] Vaughan Pratt. Techniques for conic splines. *Computer Graphics*, 19(3):151–159, July 1985.

[13] T. W. Sederberg. *Implicit and Parametric Curves and Surfaces for Computer Aided Geometric Design.* PhD thesis, Dept. of Mechanical Engineering, Purdue University, 1983.

[14] T. W. Sederberg, D. C. Anderson, and R. N. Goldman. Implicitization, inversion, and intersection of planar rational cubic curves. *Computer Vision Graphics and Image Processing*, 31(1):89–102, July 1985.

[15] J. Van Aken and M. Novak. Curve-drawing algorithms for raster displays. *ACM Transactions on Graphics*, 4(2):147–169, April 1985.