ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/267267354

A Method of Deviation for Drawing Implicit Curves

Article · January 2011

CITATIONS
3

READS
406

2 authors, including:

Kumar Sankar Ray
Indian Statistical Institute
109 PUBLICATIONS **1,127** CITATIONS
SEE PROFILE

Some of the authors of this publication are also working on these related projects:

Project DNA Cryptography View project

All content following this page was uploaded by Kumar Sankar Ray on 23 April 2016.

The user has requested enhancement of the downloaded file.

A Method of Deviation for Drawing Implicit Curves

[1] Kumar S. Ray, [2] Bimal Kumar Ray

[1] *Electronics & Communication Sciences Unit, Indian Statistical Institute*
203 B.T. Road, Kolkata – 700108 India

[2] *School of Information Technology & Engineering, VIT University*
Vellore – 632 014 India

ksray@isical.ac.in, bimalkumarray@vit.ac.in

Abstract

In contrast to the mid point technique, which may not be successful in a region where two edges of a curve cross each other, this paper introduces a unified approach called Method of Deviation that draws digital curve transforming the implicit equation of analog curve into algorithm. The method involves finding the deviation of the location of the pixel from the location of the corresponding point on the analog curve. The method is applied to implicit curves of degree three (Folium of Descartes), four (Cassini Ovals) and on the curve y – sin(1/x) = 0. The curves drawn are always one pixel wide even at the singular points. For some of the curves (e.g. Folium of Descartes, Cassini Ovals) integer arithmetic is sufficient, but for some other curves (e.g. y – sin(1/x) = 0) floating point arithmetic is required.

Keywords: *Method of deviation, pixel deviation, implicit curve drawing.*

1. Introduction

Curve drawing is a fundamental tool for computer graphics. One needs an efficient algorithm for drawing curves especially for animated display. In absence of efficient algorithm the animated display will deviate from realistic view.

The most common algorithm for drawing curves is the midpoint technique introduced by Pittway [1] and elaborated by Van Aken [2] and van Aken and Novak [3] for drawing implicit curves. This algorithm draws pixel using a decision parameter, based on whether the midpoint of the next two possible pixels is inside or outside the curve. In case, the curve has gentle positive slope in the area where the pixels are to be drawn and if the last pixel drawn is (x_k, y_k) then the mid point technique finds the midpoint of $(x_k + 1, y_k)$ and $(x_k + 1, y_k + 1)$. If this midpoint is inside the curve then the next pixel to be drawn for the curve is $(x_k + 1, y_k)$, otherwise the next pixel to be drawn is $(x_k + 1, y_k + 1)$. For regions with slope other than gentle positive, similar decision criterion can be derived using symmetry. The decision for selecting the next pixel is based purely on the observation of the relative position of the midpoint with respect to the path of analog curve. The midpoint algorithm, not being sure of which of the pixels $(x_k + 1, y_k)$ or $(x_k + 1, y_k + 1)$ to turn on, takes the midpoint and test the same to find out whether it is inside or outside the curve. This approach deviates from the principle of analytic geometry which states that the equation of an analog curve is satisfied by every point that lies on the curve and is not satisfied by every other point that does not lie on the curve. One of the two points $(x_k + 1, y_k)$ and $(x_k + 1, y_k + 1)$ may satisfy the equation of the curve, but their midpoint may not. Moreover, the mid point method may not be successful in a region where two edges of a curve actually cross each other.

This paper takes the direct mathematical approach to determine the next pixel to be drawn for a curve. In case, the curve has gentle positive slope in the area where the pixels are to be drawn and if dx and dy be the step size then each of the two points $(x_k + dx, y_k)$ and $(x_k + dx, y_k + dy)$ is substituted in the equation of the curve. The pixel that takes the equation of the curve closer to zero than the other pixel, is selected as the next pixel for drawing. For regions with slope other than gentle positive, similar technique can be applied using symmetry. In order to demonstrate the method it is applied on implicit curves of degree three (Folium of Descartes), four (Cassini Ovals) and on the curve $y - \sin(1/x) = 0$. The curves drawn are always one pixel wide even at the singular points. In case of some of the curves (e.g. Folium of Descartes, Cassini Ovals) integer arithmetic is sufficient but for some other curves (e.g. $y - sin(1/x) = 0$) floating point arithmetic is required.

The paper is organized as follows. The section 2 gives an overview of related works, section 3 introduces the *Method of Deviation*, section 4 gives the algorithm, section 5 demonstrates the method of deviation for two algebraic curves and a trigonometric curve and in section 6, we present a comparison of our procedure with a recent one and in section we draw the conclusion.

2. Related Works

A planar implicit curve is defined by the level set of a mapping f: R^2 to R. Basically there are two schemes for drawing implicit curves. One of these is the subdivision scheme and the other is the continuation scheme. The subdivision methods repeatedly divide a rectangular region into smaller regions and tests whether the curve intersects the regions until the region dimensions reduce to pixel dimensions. At each subdivision, those cells that do not intersect the curve are discarded. The fundamental problem of this scheme is of two folds. Firstly, the selection of the grid so that we do not lose small components of the curve and secondly, determination of whether a cell in the grid intersects the curve.

Continuation methods require one or more starting pixels on the curve and then draw the curve continuously. This approach has its root in the Bresenham`s algorithm for drawing lines and circles. The mid point method too employs continuation scheme. Usually the slope of the curve is used as the guiding factor for traversing the curve. But there also exists algorithm, iterative in nature that does not require slope computation. This paper introduces *Method of Deviation* which is a non-iterative continuation scheme that needs slope computation (it may be possible to avoid this computation using the finite difference approximation of partial derivative). A version of the method of deviation can be applied to lines and conic sections also and require the same operation counts and run time as compared to the mid point algorithm. Though there exists algorithms (e.g. [23]) that operate on any curve, but the disadvantage of these algorithms is that they cannot exploit special features (e.g. symmetry) present on the curve and so they require the pixels to be computed for the entire curve. The present procedure, as it is required to implement on specific curves, can exploit special features present on the curve. In case the curve consists of multiple components, then for each component it is necessary to find the starting point (seed pixel) after decomposing the curve into its components using cylindrical algebraic decomposition technique [19].

The earliest of the continuation methods is the Bresenham’s algorithm [3] that uses incremental technique similar to solving differential equations. Bresenham also designed algorithm for drawing circular arcs [4, 5] and showed in [4] that the lines and circles drawn by these algorithms are the best-fit curves. The error, that is, the vertical distance between the selected pixel and the analog line/circle is always less than or at the most equal to ½. Lennon's method [7] and Cohen's method [8] trace the implicit curves computing the tangent vectors at every step. van Aken [2] presented an incremental ellipse generator which is only used in plotting an ellipse. van Aken and Novak [3] presented the midpoint method for deriving curve drawing algorithms for generating implicit curves. The method may not be successful, however, in a region where two edges of a curve cross each other. Danielsson [9] presented a continuation method for curve generation using parametric equations. It not only deals with lines but also with other curves. In the same paper, Danielsson also discussed about curve generation using implicit equation. Pittway and Watkinson [10] addressed the problem of line drawing with gray scale that results in pleasing visual effect for display devices that permit multiple levels of intensities. Dan Field [11] presented two incremental linear interpolation algorithms and analyzed their speed and accuracy. The first of these algorithms is a simple digital differential algorithm employing fixed-point arithmetic and the second one is a new algorithm which makes use of integer arithmetic and is a generalization of the Bresenham's line drawing algorithm. The algorithm is accurate and faster than the fixed-point algorithm depending upon the underlying processor. Sproull [12] gives an elegant derivation of a line drawing algorithm as a series of program transformation from the original brute-force algorithm. Tran-Thong [13] proposed a symmetric algorithm for line drawing which is independent of the order of the end points. Macilroy [14] presented an algorithm for generation of circles. Kappel [15] reviewed a number of algorithms and the different methods used in its creation, and then presented an ellipse-plotting algorithm which he claimed incorporated the best features from each of the algorithms he reviewed. Algorithms designed specifically for the generation of hyperbolas and parabolas are harder to find. Surany [16] has addressed the generation of the hyperbola rather briefly. Metzger [17] has presented an algorithm for the generation of parabolas, but Jordan et al. [18] have called it inefficient. Arnon [19] used cylindrical algebraic decomposition (cad) algorithm for drawing algebraic curve with rational coefficients. The running time of the procedure is almost the same as the cad algorithm, which varies from seconds to hours depending upon the curve. Chandler [20] proposed a tracking algorithm for implicitly defined curve, which produces the next approximating pixel by looking for a sign difference in function evaluations at the midpoints between the eight nearest neighboring pixels. It is possible for the algorithm to track the implicit curves when there are multiple points. However, for curves even without multiple points, the procedure costs between two and eight evaluations of the curve function per moving step. Hobby [21] proposed general techniques for rasterization of implicit algebraic curves providing a general solution of under sampling problem. The method is efficient and is successfully applied on conic sections and higher order curves with emphasize on curve of degree three. Taubin [22] proposed algorithm for rasterizing algebraic curves using a recursive space subdivision scheme and an asymptotically correct test. The method is computationally efficient and renders a curve of constant width, even in the neighborhoods of singular points. Morgado and Gomes [23] proposed curvature-adaptive method for rendering implicit curves without using differentiation techniques to compute self-intersections and other singularities. It also uses a new numerical method for sampling curves point wise. The method involves iterations because angular false position method is used to find the next pixel. Zheng-sheng [24] presents a method for drawing a planar implicit curve tracking it in two domains of opposite sign according to two rules introduced in the paper. The method can draw implicit curves with large curvature at some points and can successfully handle curves with multiple points. The drawing procedure costs only one of two evaluations of the curve function per moving step. The method too involves iterations because successive binary subdivision is required to improve initial approximation obtained in the form of poly lines.

3. Method of Deviation

According to analytic geometry, the equation of a planar curve is the locus of a point that moves on a plane under a specific condition. If $f(x, y) = 0$ be the implicit equation of a planar curve (also called analog curve to distinguish it from digital one) then every point (x, y) that lies on the curve satisfies the equation and every other point that does not lie on the curve does not satisfy the equation. This statement is the basis of this paper. The slope of a curve at a point (x, y) is $m = dy/dx = -f_x/f_y$ and $-\infty < m < \infty$. This slope is called negative sharp if $-\infty < m \leq -1$, negative gentle if $-1 < m < 0$, positive gentle if $0 \leq m < 1$ and positive sharp if $1 \leq m < \infty$. The curve may be divided into regions based on the nature of the slope and the monotonic behavior of x and y (e.g. the slope is positive when both x and y increase or decrease). In order to draw the digital curve from the analog equation, the region of the curve where the slope is positive gentle, the x value should be incremented (decremented) at every step and decide, based on a decision criterion, whether to increase (decrease) the y value. In the region where the slope of the curve is negative gentle, as x is increased (decreased), it is necessary to decide whether to decrease (increase) the y value. The region of the curve where the slope is sharp, the role of x and y must be interchanged. Here the methodology is discussed assuming that the slope is gentle positive and y is increasing monotonically as x increases strictly monotonically. The similar methodology can be applied to other regions of the curve.

Since this technique selects pixels based on deviation of pixel location from the corresponding point on the analog curve hence the method is called *Method of Deviation*.

3.1. Finding the Next Pixel

If (x_k, y_k) is the last pixel drawn for the curve $f(x, y) = 0$ and dx and dy are the step size in the x and y direction respectively, then the coordinates (x_{k+1}, y_{k+1}) of the next pixel is either $(x_k + dx, y_k)$ or $(x_k + dx, y_k + dy)$, depending upon which of these two points satisfies the equation of the curve $f(x, y) = 0$. In the digital domain it may not be possible for either of these two points to satisfy the equation. So it is fair to test which of these points is closer to the analog curve: $f(x, y) = 0$. To test this, substitute each of these points in turn, in the equation of the curve and take the absolute value of $d_1 = f(x_k + dx, y_k)$ and $d_2 = f(x_k + dx, y_k + dy)$. If absolute value of d_1 is greater than that of d_2 then the value of $f(x_k + dx, y_k + dy)$ is closer to zero than that of $f(x_k + dx, y_k)$ and so the pixel $(x_k + dx, y_k + dy)$ is closer to the curve than the pixel $(x_k + dx, y_k)$ and the next pixel to be drawn is $(x_k + dx, y_k + dy)$. On the other hand, if the absolute value of d_1 is not greater than that of d_2 then draw the pixel $(x_k + dx, y_k)$. So the decision criterion for selecting the next pixel is

if $|d_1| > |d_2|$ then change x to $x + dx$ and y to $y + dy$,

otherwise, change x to $x + dx$ and leave y unaltered.

It may be noted that the value of d_1 and d_2 is a measure of deviation of the pixel from the analog curve. Since it is required to compute the absolute value of d_1 and d_2 hence after computing d_1 (or d_2) if it turns out to be negative, its sign is changed, otherwise it is left as it is. This conversion from negative to positive requires if statement, one for each of d_1 and d_2. (In this case, for some of the curves double precision integer arithmetic is sufficient.) The if statement can be eliminated, if the test $|d_1| > |d_2|$ is replaced by the test $(d_1)^2 > (d_2)^2$ but the computation must be in double for all curves.

In order to start the algorithm for curve drawing, it is necessary to find a starting pixel known as seed pixel, and then

(i) divide the curve into regions of different types of slope, namely, positive gentle, negative gentle, negative sharp and positive sharp,

(ii) find out in which regions it is necessary to increase (decrease) x and increase (decrease) y.

(iii) process the curve in the same order in which different types of slopes occur along the curve.

To reduce the computation, one can exploit symmetry, if the curve is symmetrical. The algorithm for drawing curve is presented in section 4. A block diagram of the procedure is shown in figure 1.

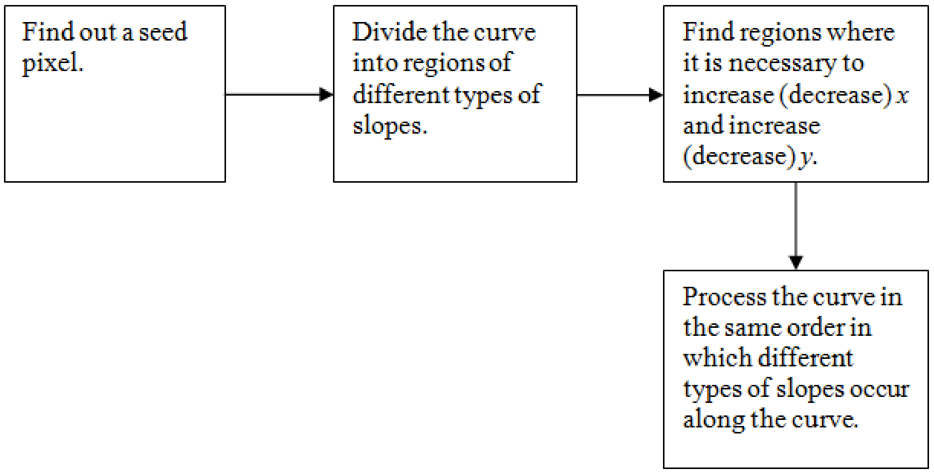
Find out a seed pixel.
Divide the curve into regions of different types of slopes.
Find regions where it is necessary to increase (decrease) x and increase (decrease) y.
Process the curve in the same order in which different types of slopes occur along the curve.

Figure 1. Block Diagram Describing the Curve Drawing Procedure

3.2. Removing the Absolute Value Computation

Further algebraic simplification (shown in the next couple of paragraphs) on the test $(d_1)^2 > (d_2)^2$ can eliminate evaluation of the second power of d_1 and d_2 and in this case computation in double is not mandatory for all curves.

The test $(d_1)^2 > (d_2)^2$ is algebraically equivalent to $(d_1 + d_2)\ (d_1 - d_2) > 0$. Writing $g_k = f(x_k, y_k)$,

$$d_1 = g_{k+1} = g_k + \varphi(x_k,\ y_k) \text{ and } d_2 = g_{k+1} = g_k + \psi(x_k,\ y_k),$$

the sum and the difference of d_1 and d_2 are

$$p_k = d_1 + d_2 = 2g_k + \varphi(x_k,\ y_k) + \psi(x_k,\ y_k) \qquad (1)$$

$$q_k = d_1 - d_2 = \varphi(x_k,\ y_k) - \psi(x_k,\ y_k). \qquad (2)$$

The sign of p_k is the same as that of q_k and the sign of q_k may be different in the regions of the curve with different types of slope. The sign of q_k can be determined either trivially or empirically, using the test for positive (or negative) definiteness (or semi-definiteness) of polynomial expression.

Assuming that the slope is positive gentle, the incremental formulae for p_k and the code segment to determine the $(k+1)^{st}$ pixel are derived in the following way.

$$p_{k+1} \quad = 2g_{k+1} + \varphi(x_{k+1},\ y_{k+1}) + \psi(x_{k+1},\ y_{k+1})$$

$$= 2g_k + 2\varphi(x_k,\ y_k) + \varphi(x_{k+1},\ y_{k+1}) + \psi(x_{k+1},\ y_{k+1}),$$
$$\text{if } x_{k+1} = x_k + dx \text{ and } y_{k+1} = y_k \qquad (3)$$

and

$$p_{k+1} = 2g_{k+1} + \varphi(x_{k+1} + dx,\ y_{k+1}) + \psi(x_{k+1} + dx,\ y_{k+1} + dy)$$
$$= 2g_k + 2\psi(x_k,\ y_k) + \varphi(x_{k+1},\ y_{k+1}) + \psi(x_{k+1},\ y_{k+1}),$$
$$\text{if } x_{k+1} = x_k + dx \text{ and } y_{k+1} = y_k + dy. \qquad (4)$$

From (1) and (3) it follows that

$$p_{k+1} = p_k + \varphi(x_k,\ y_k) - \psi(x_k,\ y_k) + \varphi(x_{k+1},\ y_{k+1}) + \psi(x_{k+1},\ y_{k+1}),$$
$$\text{if } x_{k+1} = x_k + dx \text{ and } y_{k+1} = y_k$$

i.e. $p_{k+1} = p_k + \eta(x_k + dx, y_k)$.

From (1) and (4)

$$p_{k+1} = p_k + \psi(x_k,\ y_k) - \varphi(x_k,\ y_k) + \varphi(x_{k+1},\ y_{k+1}) + \psi(x_{k+1},\ y_{k+1}),$$
$$\text{if } x_{k+1} = x_k + dx \text{ and } y_{k+1} = y_k + dy$$

i.e. $p_{k+1} = p_k + \zeta(x_k + dx, y_k + dy)$.

When the initial point is one of the known points on the analog curve, $g_k = 0$ at the starting point and the initial value of $p_k = \varphi(x_k,\ y_k) + \psi(x_k,\ y_k)$.

Assuming that q_k is positive definite,

```
write $x_{k+1} = x_k + dx$
If $p_k > 0$ then
        write $y_{k+1} = y_k + dy$
        $p_{k+1} = p_k + \zeta(x_k + dx, y_k + dy)$
else
        $p_{k+1} = p_k + \eta(x_k + dx, y_k)$
```

Similar code segment may be written for other regions of a curve. It may be noted that whenever the nature of slope of a curve changes, the decision parameter p is to be reinitialized.

4. Algorithm

Algorithm: Method of Deviation

The input is the starting pixel, the expression $f(x, y)$ of the implicit curve, the step size dx and dy. The output is the digital curve.

Step1. Divide the curve into regions of different types of slope, namely, positive gentle, negative gentle, negative sharp and positive sharp, find out in which regions it is necessary to increase (decrease) x and increase (decrease) y, and process the curve in the same order in which different types of slopes occur along the curve. To reduce the computational load, one can exploit symmetry, in case the curve is symmetrical.

Step 2. The region where the curve has gentle positive slope and both x and y increases.

Step 2.1 Change x to $x + dx$, compute $d_1 = f(x, y)$ and $d_2 = f(x,\ y + dy)$

Step 2.2 If d_1 is negative then assign $-d_1$ to d_1 -- It is possible to avoid this if statement -- and replace it by an assignment of $d_1 * d_1$ to d_1 but higher precision is required.

Step 2.3 If d_2 is negative then assign $-d_2$ to d_2

Step 2.4 If $d_1 > d_2$ then change y to $y + dy$.

Step 2.5 Repeat steps 2.1 to 2.4 as long as the slope is positive gentle.

Step 3. The region where the curve has gentle positive slope but both x and y decreases do steps 2.1 to 2.5 replacing dx by $-dx$ and dy by $-dy$

Step 4. The region where the curve has sharp positive slope and both x and y increases perform steps 2.1 to 2.5 interchange the role of x and y.

Step 5. The region where the curve has sharp positive slope and both x and y decreases perform step 4 replacing dx by $-dx$ and dy by $-dy$

Step 6. The region where the curve has sharp negative slope and x decreases as y increases.

Step 6.1 Change y to $y + dy$, compute $d_1 = f(x, y)$ and $d_2 = f(x_k - dx,\ y)$

Step 6.2 If d_1 is negative then assign $-d_1$ to d_1

Step 6.3 If d_2 is negative then assign $-d_2$ to d_2

Step 6.4 If $d_1 > d_2$ then change x to $x + dx$.

Step 6.5 Repeat steps 6.1 to 6.4 as long as the slope is sharp negative.

Step 7. The region where the curve has sharp negative slope but x decreases as y increases do steps 6.1 to 6.5 replacing dx by $-dx$ and dy by $-dy$

Step 8. The region where the curve has gentle negative slope and y decreases as x increases.

Step 8.1 Change y to $y - dy$, compute $d_1 = f(x, y)$ and $d_2 = f(x + dx,\ y)$

Step 8.2 If d_1 is negative then assign $-d_1$ to d_1

Step 8.3 If d_2 is negative then assign $-d_2$ to d_2

Step 8.4 If $d_1 > d_2$ then change x to $x + dx$.

Step 8.5 Repeat step 8.1 to 8.4 as long as the slope is gentle negative.

Step 9. The region where the curve has gentle negative slope but x increases as y decreases do steps 8.1 to 8.5 after replacing dx by $-dx$ and dy by $-dy$

5. Examples

This section shows application of the Method of Deviation on three different curves namely, Folium of Descartes, Cassini ovals and $y - \sin(1/x) = 0$.

Example 1. Folium of Descartes

The implicit equation of Folium of Descartes is $f(x, y) = x^3 + y^3 - 3axy = 0$. The algorithm starts at the point (0, 0) and draws the leaf and then draws the tail of the curve. The slope of the curve at any point (x, y) is $m = (x^2 + ay)/(y^2 + ax)$. The slope is sharp and positive ($x^2 + ay > y^2 + ax$) along a part of the upward arc (Figure 2) then the slope becomes gentle positive (x^2

$+ ay \geq 0$), then it transforms into gentle negative ($x^2 + ay < -y^2 - ax$), and ultimately equals exactly to -1. After the leaf is drawn the coordinates are reset to $x = 0$ and $y = 0$ to draw the tails. Along the upper tail of the leaf, the slope is gentle negative ($x^2 + ay < -y^2 - ax$) and as x decreases, y increases. The curve is symmetrical about the line $y = x$ and so the algorithm generates the coordinates (x, y) of the pixels for only one half of the curve and the pixel coordinates for the other half are (y, x).
The Figure 2 shows this curve drawn with the method of deviation for $a = 50$. The arithmetic is performed in long integers.

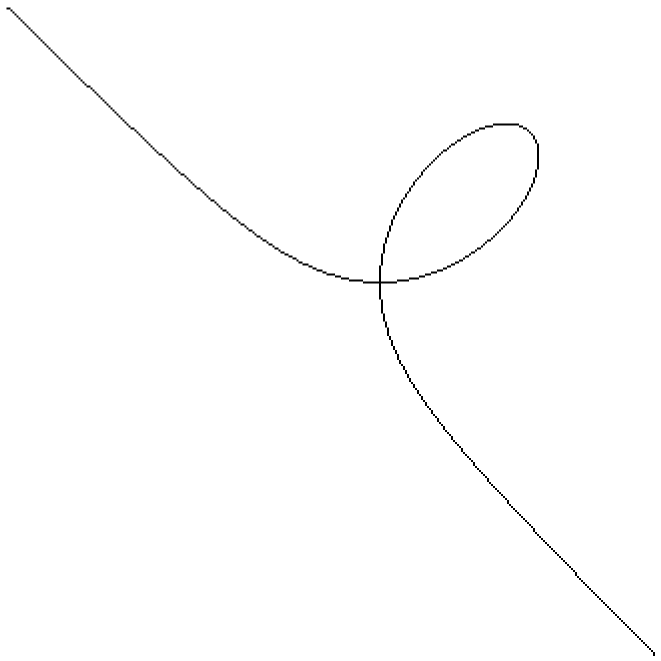

Figure 2. Folium of Descartes Drawn Using the Proposed Method

Example 2. Cassini Ovals

The Cassini ovals are a family of quartic curves, also called Cassini ellipses, described by a point such that the product of its distances from two fixed points a distance $2a$ apart is a constant (b^2). The shape of the curve depends on b/a. If $a < b$, the curve has a single loop with an oval or dog bone shape. The case $a = b$ produces a lemniscates of Bernoulli. If $a > b$, then the curve consists of two loops. The implicit equation of Cassini ovals may be written as

$$f(x, y) = x^4 + y^4 + 2x^2y^2 - 2a^2x^2 + 2a^2y^2 + a^4 - b^4 = 0.$$

The curve is symmetrical about the x axis as well as the y axis. So it is sufficient to generate pixels in the first quadrant of the coordinate system and use reflection about the x-axis and the y-axis to generate the other three pixels. Starting from the point ($\sqrt{(a^2 + b^2)}$, 0), the slope is initially sharp negative as the curve is traversed upward and as y increases, x decreases and transforms into negative gentle with as y increases, x decreases and then it transform into positive gentle, as x decreases, y also decreases. Finally, the slope becomes sharp positive and as x decreases, y also decreases. The algorithm should be stopped before x and y become negative.

The Figure 3 shows this curve drawn with the method of deviation with a = 50, b/a = 0.1 to 1.5 with a step size of 0.1 of b/a. The arithmetic is performed in long integers.

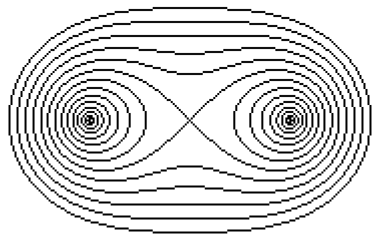

Figure 3. Cassini Ovals Drawn Using the Proposed Method

Example 3. $y - \sin(1/x) = 0$

This is a non-algebraic implicit curve. This curve oscillates rapidly in the neighborhood of x = 0 and its slope rapidly oscillates between sharp positive and sharp negative and ultimately as x increases, the slope also becomes gentle positive and gentle negative apart from being sharp positive and sharp negative. The derivative of this curve is $-\cos(1/x)/x^2$. The loop starts with an initial value of x and ends with a final value of x. In this paper, the implementation of the algorithm is done in the range of x = 0.1 to 2.7 with a step size of $dx = dy$ = (2.7 – 0.1)/13000. The test on the slope is performed inside if statements which in turn are embedded inside a loop on x. The curve drawn is shown in Figure 4.

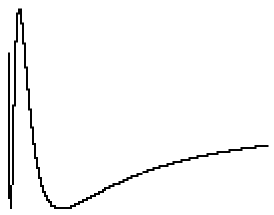

Figure 4. The Curve $y - \sin(1/x) = 0$ Drawn Using the Proposed Method.

6. Comparisons

The algorithm Method of Deviation presented in section 4 is compared with the procedure described in [24]. The later is a two stage process. At the first stage, a ladder polyline is created as a first approximation to the curve and at the second and final stage recursive binary subdivision is performed. For finding the ladder polyline, generation of the next point requires one function evaluation and four if statements in the non-monotonic segments of the curve and in the monotonic segments it requires one function evaluation and two if statements. The second stage is iterative in nature because it is recursive. If the curve is traced with adaptive step size then additional if statement is required. On the contrary, the method of deviation does not require recursive subdivision. One scan of the analog curve can draw the digital curve. For finding the next value of (x, y), the method requires two function evaluations and three if statements irrespective of whether the segment being traced is monotonic or not.

7. Conclusion

An algorithm for drawing implicit curves is presented. The core algorithm may directly be applied to a curve without any simplification. The simplification is required so as to avoid use of absolute operation on deviations. The arithmetic required depends upon the curve to be drawn. In case of some of the curves integer arithmetic is sufficient but for some other curves floating point arithmetic is required. The method is compared with that described in [24] and is found to require less number of if statements for monotonic segments of a curve irrespective of whether the step size is adaptive or constant. Since the method is based on computation of the deviation of pixel location from the corresponding point on the analog curve hence the method may be called *Method of Deviation.*

References

[1] M. Pitteway, “Algorithm for drawing ellipses or hyperbolae with a digital plotter”, Comput. J. vol. 10, no. 3, 1967, pp. 282-289,

[2] J. Van Aken, “An efficient ellipse-drawing algorithm”, IEEE Computer Graphics & Applications, vol. 4, no. 9, 1984, pp. 24-35.

[3] J. Van Aken, and M. Novak, “Curve-drawing algorithms for raster displays”, ACM Transactions on Graphics, vol. 4, no. 2, 1985, pp. 47-169..

[4] J.E. Bresenham, “Algorithm for computer control of a digital plotter”, IBM System Journal. vol. 4, no. 1, 1965, pp. 25-30.

[5] J. E. Bresenham, “Algorithms for circular arc generation”, Fundamental Algorithms for Computer Graphics, R.A. Earnshaw, ed., NATO ASI Series, F17, Springer Verlag, Berlin, 1985, pp. 197-218.

[6] J. E. Bresenham, “A linear algorithm for incremental digital display of circular arcs”, Communications of the ACM, vol. 20, no. 2, 1977, pp.100-106.

[7] W.J. Lennon, B.W. Jordan, and B.C. Holm, “An improved algorithm for the generation of nonparametric curves”, IEEE Transactions on Computers, **C-22**, 1973, pp. 1052-1060.

[8] E. Cohen, “A method for plotting curves defined by implicit equation”, Computer Graphics (SIGGRAPH’76), 1976.

[9] P. E. Danielsson, “Incremental Curve Generation”, IEEE Transactions on Computers, vol. C-19, 1970, pp.763 – 793.

[10] M. L. V. Pittway, and D.J.Watkinson, “Bresenham’s algorithm with gray scale”, Communications of the ACM, vol.23, 1980, pp. 625 – 626.

[11] D. Field, “Incremental Linear Interpolation”, ACM Transactions on Graphics, vol. 4, 1985, pp. 1– 11.

[12] R.F. Sproull, “Using program transformations to derive line-drawing algorithms”, ACM Transactions on Graphics, vol.1, 1982, pp. 259 –273.

[13] Tran-Thong, “A symmetric linear algorithm for line segment generation”, Computers and Graphics, vol. 6, 1982, pp. 15 – 17.

[14] M. D. Mcilroy, “Best approximate circles on integer grids”, ACM Transaction on Graphics, vol. 2, no. 4, 1983, pp. 237 – 263.

[15] M. R. Kappel, ”An ellipse-drawing algorithm for raster displays”, Fundamental Algorithms for Computer Graphics, R.A. Earnshaw, ed., NATO ASI Series, vol. F17, Springer Verlag, Berlin, 1985, pp. 257 – 280.

[16] A. P. Surany, ”An ellipse-drawing algorithm for raster displays”, Fundamental Algorithms for Computer Graphics, R.A. Earnshaw, ed., NATO ASI Series, vol. F17, Springer Verlag, Berlin, 1985, pp. 281– 285.

[17] R. A. Metzger, “Computer generated graphics segments in a raster display”, Spring 1969 Joint Computer Journal Conference, AFIPS Conference Proceeding, pp. 161 – 172.

[18] B. W. Jordan, W. J. Lennon, and B. C. Holm, “An improved algorithm for the generation of nonparametric curves”, IEEE Transaction on Computers C-22, no. 12, 1973, pp.1052-1060.

[19] D. S. Arnon., "Topcdogically reliable display of algebraic curves", Computer Graphics, vol. 17, no. 3, 1983, pp. 219-227.

[20] R.E. Chandler, "A tracking algorithm for implicitly defined curves", IEEE Computer Graphics and Applications, vol. 8, no. 2, 1988, pp. 83-89.

[21] J.D. Hobby, "Rasterization of nonparametric curves", ACM Transactions on Graphics, 9, no. 3, July 1990, pp. 162-277.

[22] G. Taubin, "Rasterizing implicit curves by space subdivision", IBM Research Division Technical Report RC17913, April 1992.

[23] F.M. Morgado José, and J.P Gomes Abel., "A Derivative-Free Tracking Algorithm for Implicit Curves with Singularities", Lecture Notes in Computer Science, vol. 3039, 2004,. pp. 221 – 228.

[24] YU Zheng-sheng, CAI Yao-zhi, OH Min-jae, KIM Tae-wan, and PENG Qun-sheng, "An efficient method for tracing planar implicit curves", Journal of Zhejiang University Science A, vol. 7, no. 7, 2006, pp. 1115-1123.

Authors

Kumar Sankar Ray received his BE in Mechanical Engineering in 1977 from Calcutta University, Kolkata, India, MSc in Control Engineering in 1980 from the University of Bradford, Bradford, UK, and PhD in Computer Science in 1987 from Calcutta University. He is currently a Professor of Electronics and Communication Sciences Unit, Indian Statistical Institute, Calcutta. He was a Visiting Faculty Member of the University of Texas at Austin under a United Nations Development Programme (UNDP) fellowship in 1990. His fields of interests are control theory, computer vision, AI, fuzzy reasoning, neural networks, genetic algorithms, qualitative physics and DNA computing.

Bimal Kumar Ray received his Ph.D. degree in Computer Science from Indian Statistical Institute, Kolkata, India. He received his Master degree in Applied Mathematics from Calcutta University and his Bachelor degree in Mathematics from St. Xavier's college, Kolkata. His research interests are in computer graphics, computer vision and image processing. He has published a number of research papers in peer reviewed journals.

View publication stats