SIMPLE ALGORITHM FOR ANTIALIASED SCAN CONVERSION OF STRAIGHT LINE SEGMENTS

Jacek Lebiedź
Technical University of Gdańsk
Faculty of Electronics, Telecommunications and Informatics
e-mail: jacekl@pg.gda.pl

ABSTRACT

The paper presents a simple algorithm for antialiased scan conversion of straight line segments. Its speed is estimated and its performance is demonstrated.

Keywords: antialiasing, discrete line segments, scan conversion, raster graphics

1. INTRODUCTION

Every line on a raster is a sequence of adjacent square (or rectangular) pixels. Therefore it can have jagged edges. This undesirable effect known as aliasing is the result of a binary approach to scan conversion in which each pixel is filled with one of two colors: color of either the line or a background.

2. ANTIALIASING

There are two basic approaches to reduce or eliminate aliasing. The first is by increasing resolution, but this is an expensive solution that only diminishes the problem of jaggies. The second is by painting pixels in various shades of intermediate color between the colors of the line and the background (e.g. gray for the black line and the white background) [Foley90].

There are a few techniques of setting pixel's intensity. The technique of setting intensity proportional to the amount of pixel's area covered by 1-pixel-thick line is called unweighted area sampling. The other technique, called weighted area sampling, allows us to assign different weights to different parts of the pixel. In weighted area sampling, a small area closer to the pixel center has greater influence on the intensity of the pixel than does one at a greater distance. This influence is determined by so-called weighting function [Foley90]. Particularly, the cone weighting function is often integrated into scan-conversion algorithms [Gupta81, Foley90].

3. SIMPLIFIED STRATEGY FOR STRAIGHT LINE SEGMENTS

Unweighted and weighted area samplings are computationally much more expensive than binary approach to scan-conversion. To determine the pixel's intensity, these techniques demand additional calculations using even fractional arithmetic. We can avoid these additional calculations only using a decision variable determining the choice of pixel (in incremental algorithm) directly to set pixel's intensity. In this case the pixel's intensity is proportional to the vertical or horizontal distance from the center of the pixel to the true line instead of to the amount of area covered.

Assume for simplification that we deal with a straight line segment which has an angle of slope $\varphi \in (0°,45°)$. At most two pixels with the same x coordinate will be intensified. Consider the line in fig. 1, where the previously selected pixels appear as a gray squares and the three pixels from which to choose two at the next step are shown as unfilled squares.

Assume that we have just selected the couple of pixels (x_i, y_i-1) and (x_i, y_i) and now must choose between the lower couple of pixels (x_i+1, y_i-1) and (x_i+1, y_i) or the upper couple of pixels (x_i+1, y_i) and (x_i+1, y_i+1). Let k_i be the vertical distance from the center of the pixel (x_i+1, y_i) to the intersection point of the true line with the grid line $x = x_i+1$. The sign of k_i determines the choice of couple of pixels. If

$k_i > 0$, then the lower couple is chosen ($x_{i+1}:=x_i+1$, $y_{i+1}:=y_i$, $k_{i+1}:=k_i-dy/dx$). If $k_i < 0$, then the upper couple is chosen ($x_{i+1}:=x_i+1$, $y_{i+1}:=y_i+1$, $k_{i+1}:=1+k_i-dy/dx$). The value $k_i = 0$ involves the choice of only one pixel (x_i+1, y_i). The intensity values for chosen pixels are proportional to $|k_i|$ for the pixel (x_i+1, $y_i \pm 1$) and to $1-|k_i|$ for the pixel (x_i+1, y_i).

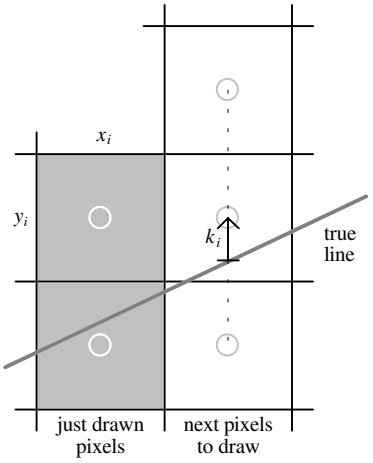

Fig. 1. Choice of the pixels to draw in the simple algorithm for antialiased scan conversion of straight line segments.

Fig. 2 presents the simple algorithm for antialiased scan conversion of straight line segments. To eliminate the fraction in k_i, the decision variable is redefined by multiplying it by the horizontal distance Δx between the segment's ends. The pixel's intensity is set by call to procedure shade(x, y, c, c_{max}: integer), where x, y are pixel's coordinates and c is pixel's intensity measured on a scale of 0 to c_{max}. This procedure may use a predefined table of intensity to convert the pixel's intensity to the scale of a device.

```
var x, y, xe, ye, dx, dy, kdx: integer;
dx := xe-x;  dy := ye-y; {dx ≥ dy ≥ 0 for φ∈(0°,45°)}
kdx := 0;
while x ≠ xe do
    begin
        shade(x, y, dx-kdx, dx);
        if kdx ≠ 0 then  shade(x, y-1, kdx, dx);
        x := x+1; kdx := kdx-dy;
        if kdx < 0 then
            begin y := y+1; kdx := kdx+dx end
    end;
shade(xe, ye, 1, 1).
```

Fig. 2. Simple algorithm for antialiased scan conversion of straight line segment connecting pixels (x, y) (xe, ye) for an angle of segment slope $\varphi \in (0°,45°)$.

4. SPEED COMPARISON

Table 1 shows speed comparison of the simple antialiased segment algorithm (fig. 2) with the bilevel conditional Bresenham's segment algorithm (fig. 4 in Appendix A). The latter algorithm is twice as fast as the former, but the former draws almost twice as many pixels as the latter. The time needed to draw one pixel is near the same for both of the algorithms. Additionally this table contains execution time of the Wu's antialising algorithm [Wu91] without bisection (fig. 5 in Appendix B). Bisection is abandoned because it can be applied to the simple antialiasing algorithm too. Presented results are based on the assumption that the execution times of basic operations are defined by table 2.

algorithm	execution time $t_{algorithm}$
simple antialiasing algorithm	$12 + 19\cdot\Delta x + \Delta y - 5\cdot\gcd(\Delta x, \Delta y)$
bilevel Bresenham's algorithm	$15 + 9\cdot\Delta x + \Delta y$
Wu's antialiasing algorithm	$27 + 21\cdot\Delta x - 6\cdot\gcd(\Delta x, \Delta y)$

Tab. 1. Calculated execution times of considered algorithms (gcd - greatest common divisor).

Operation	execution time $t_{operation}$
:=	1
+, –	1
$\cdot\, 2^{\pm n}$ (shl, shr)	1
in-, de-crementation	1
=, ≠, >, ≥, <, ≤	1
overflow	1
plot	3
shade	5
div	12
if *cond* then *op*	$t_{cond} + cond\ ?\ t_{op} : 1$
if *cond* then *op1* else *op2*	$t_{cond} + 1 + cond\ ?\ t_{op1} : t_{op2}$
while *cond* do *op*	$(t_{cond} + 1)\cdot(\#+1) + \#\cdot t_{op}$
for *var*:=1 to # do *op*	$(\#+2) + \#\cdot t_{op}$

Tab. 2. Assumed execution times of basic operations (# - the number of loop runs, *cond* ? *u* : *v* ≡ if *cond* then *u* else *v*).

We omit the speed comparison with other antialiased segment algorithms because of their rather great complexity (floating-point operations) [Foley90, Gupta81, Thomp90].

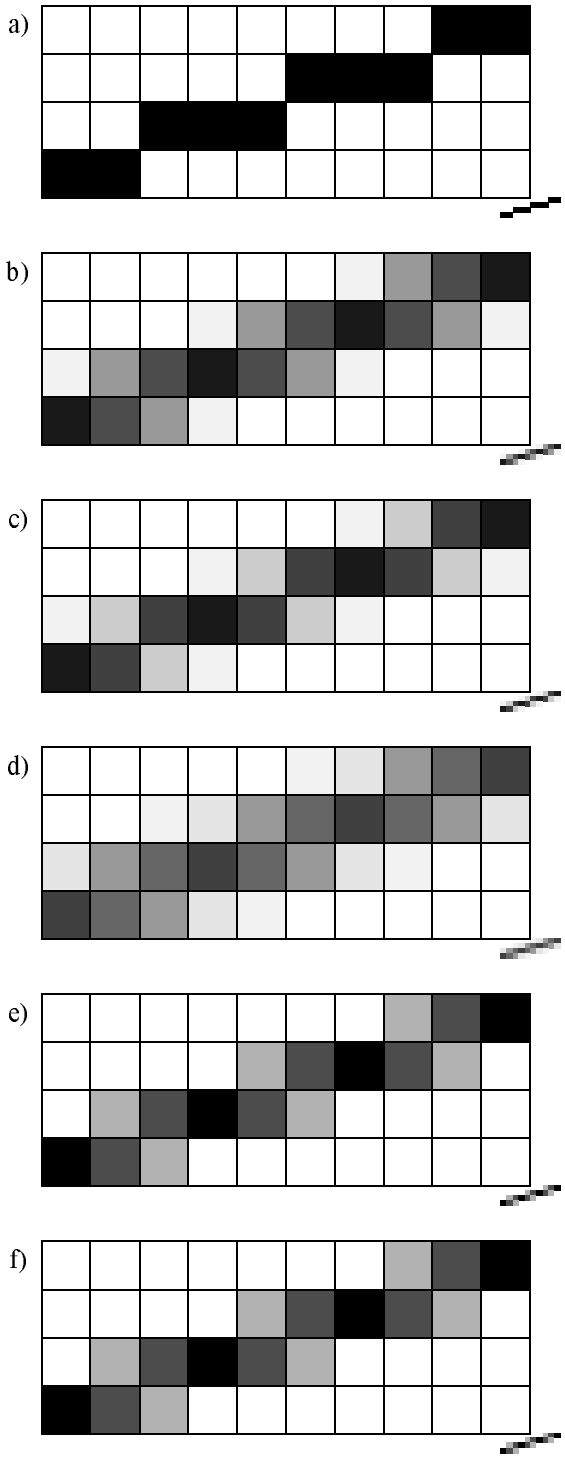

Fig. 3. Straight line segment drawn
by different scan conversion algorithms:
a) bilevel conditional Bresenham's algorithm
(fig. 4 in Appendix A),
b) unweighted area sampling,
c) weighted area sampling
with the pyramid weighting function,
d) weighted area sampling
with the cone weighting function
for a base radius equal to a grid unit,
e) simple antialiasing algorithm (fig. 2),
f) Wu's algorithm (fig. 5 in Appendix B).

5. VISUAL RESULTS

The performance of the simple antialiased segment algorithm is demonstrated by fig. 3. It shows that the line generated by the simple algorithm is not inferior in quality to those produced by the other antialiasing techniques.

6. CONCLUSION

The paper presents a simple algorithm for antialiased scan conversion of straight line segments. This algorithm offers smooth lines with high processing speed and low implementation and computational costs. The quality of drawn lines is comparable with the quality given by other antialiasing algorithms, but the speed of drawing is comparable with the speed reached by the traditional bilevel Bresenham's algorithm.

7. REFERENCES

[Barro79] J. Barros, H. Fuchs: Generating Smooth 2-D Monocolor Line Drawings on Video Displays. *Computer Graphics* 13 (2) (*Proceedings of SIGGRAPH'79*), August 1979, ACM SIGGRAPH New York, pp. 260÷269.

[Brese65] J. E. Bresenham: Algorithm for Computer Control of a Digital Plotter. *IBM Systems J.*, January 1965, pp. 25÷30.

[Brese87] J. E. Bresenham: Ambiguities in Incremental Line Rastering. *IEEE Computer Graphics and Applications* 7 (5), May 1987, pp. 31÷43.

[Brese96] J. E. Bresenham: Pixel-Processing Fundamentals. *IEEE Computer Graphics and Applications* 16 (1), January 1996, pp. 74÷82.

[Crow78] F. Crow: The Use of Grayscale for Improved Raster Display of Vectors and Characters. *Computer Graphics* 12 (3) (*Proceedings of SIGGRAPH'78*), August 1978, ACM SIGGRAPH New York, pp. 1÷5.

[Field86] D. Fields: Algorithms for Drawing Anti-Aliased Circles and Ellipses. *Computer Vision, Graphics and Image Processing* 33 (1), January 1986, pp. 1÷15.

[Foley90] J. D. Foley, A. van Dam, S. K. Feiner, J. F. Hughes: *Computer Graphics: Principles and Practice, Second Edition*. Addison-Wesley, Reading 1990.

[Fujim83] A. Fujimoto, K. Iwata: Jag-Free Images on Raster Displays. *IEEE Computer Graphics and Applications* 3 (9), December 1983, pp. 26÷34.

[Gupta81] S. Gupta, R. E. Sproull: Filtering Edges for Gray-Scale Displays. *Computer Graphics* 15 (3) (*Proceedings of SIGGRAPH'81*), August 1981, ACM SIGGRAPH New York, pp. 1÷5.

[Pitte80] M. L. V. Pitteway, D. J. Watkinson: Bresenham's Algorithm with Grey-Scale. *Communications of the ACM* 23 (11), November 1980, pp. 625÷626.

[Pitte87] M. L. V. Pitteway: Soft Edging Fonts. Computer Graphics Technology and Systems. *Proceedings of the Conference Held at Computer Graphics '87* (*London, October 1987), Advanced computing series* 9, Online Publications, London 1987.

[Schil91] A. Schilling: A New Simple and Efficient Antialiasing with Subpixel Masks. *Computer Graphics*, 25 (4) (*Proceedings of SIGGRAPH'91*), July 1991, ACM SIGGRAPH New York, pp. 133÷141.

[Thomp90] K. Thompson: Rendering Anti-Aliased Lines. *Graphics Gems (edited by A. S. Glassner)*, Academic Press Professional, Cambridge 1990, pp. 105÷106 (and C implementation, pp. 690÷693).

[Turko82] K. Turkowski: Anti-Aliasing Through the Use of Coordinate Transformations. *ACM Transactions on Graphics* 1 (3), July 1982, pp. 215÷234.

[Whitt83] T. Whitted: Anti-Aliased Line Drawing Using Brush Extrusion. *Computer Graphics*, 17 (3) (*Proceedings of SIGGRAPH'83*), July 1983, ACM SIGGRAPH New York, pp. 151÷156.

[Wu90] X. Wu: Anti-Aliased Circle Generation. *Graphics Gems II (edited by J. Arvo)*, Academic Press, San Diego 1990, pp. 446÷450.

[Wu91] X. Wu: An Efficient Antialiasing Technique. *Computer Graphics*, 25 (4) (*Proceedings of SIGGRAPH'91*), July 1991, ACM SIGGRAPH New York, pp. 143÷152.

APPENDIX A: BILEVEL CONDITIONAL BRESENHAM'S ALGORITHM

```
var x, y, xe, ye, dx, dy, p, pinc, pdec: integer;
dx := xe-x;  dy := ye-y; {dx ≥ dy ≥ 0 for φ∈(0°,45°)}
pinc := dy+dy;   p := pinc-dx;   pdec := p-dx;
while x ≠ xe do
    begin
        plot(x,y);
        x := x+1;
        if  p < 0 then  p := p+pinc
        else       begin y := y+1; p := p+pdec end
    end;
plot(xe,ye).
```

Fig. 4. Bilevel conditional Bresenham's algorithm for scan conversion of straight line segment connecting pixels (*x*, *y*) (*xe*, *ye*) for an angle of segment slope φ∈(0°,45°) [Brese65, Brese87].

APPENDIX B: WU'S ANTIALIASING ALGORITHM

```
const  n {bits of integer},
       nm {n-m>0,
             where 2^m is the number of gray levels},
       em {2^m-1};
var x, y, xe, ye, d, dinc: integer;
d := 0;
dinc := (((ye-y) shl n)+((xe-x) shr 1)) div (xe-x);
                    { 0 ≤ dinc ≤ 1  for  φ∈(0°,45°) }
while x ≠ xe do
    begin
        shade(x, y, em - (d shr nm) , em);
        if d ≠ 0 then  shade(x, y+1, d shr nm, em);
        x := x+1; d := d+dinc;
        if overflow(d) then  y := y+1
    end;
shade(xe, ye, em, em).
```

Fig. 5. Wu's algorithm for antialiased scan conversion of straight line segment connecting pixels (*x*, *y*) (*xe*, *ye*) for an angle of segment slope φ∈(0°,45°) [Wu91] without bisection.