PERGAMON

Computers & Graphics 25 (2001) 187–193

COMPUTERS & GRAPHICS

www.elsevier.com/locate/cag

Technical Section

A new antialiased line drawing algorithm☆

Bei-Chuan Chen[a], Yu-Tai Ching[b,*]

[a]Animeta Systems, Inc., 4F, No. 778, Sec. 4, Patch Road, Taipei, Taiwan

[b]Department of Computer and Information Science, National Chiao Tung University, 1001 Ta Hsueh Road, Hsinchu 30010, Taiwan, ROC

Abstract

Consider a line $f(x) = mx + b, 0 \leqslant m \leqslant 1$. Conventional line drawing algorithms sample $(x, f(x))$ on the line, where x must be an integer, and then map $(x, f(x))$ to the frame buffer according to the defined filter and $f(x)$. In this paper, we propose to simulate a sampled point $(x, f(x))$ by the four pixels around it where x and $f(x)$ are not necessary to be integers. Based on the proposed low-pass filtering, we show that the effect of sampling at infinite number of points along a line segment can be achieved since the closed form of the intensities assigned to pixels exists. Furthermore, we show the coherence properties that can reduce the cost for computing these intensities.

Keywords: Computer graphics; Line drawing algorithm; Antialiasing

1. Introduction

Line drawing is the most primitive operation in computer graphics. Currently, the most common display device is the raster display. Each pixel in the raster display has integer coordinates but can display gray scales. Drawing a line on the raster display simulates a continuous curve $y = f(x)$ by the set of points and the associated gray values.

Assume that a line segment has slope $m, 0 \leqslant m \leqslant 1$. A straightforward approach is to sample every point with integer x-coordinate on the line segment and to calculate the value $y = f(x)$ where it is not necessary for y to be an integer. An "all-or-nothing" approach is to simulate the locus of the function $y = f(x)$ by the set of pixels $\{(x, g(f(x)))\}$, where $g(f(x))$ is either $\lceil f(x) \rceil$ or $\lfloor f(x) \rfloor$ depending on which one is closer to $f(x)$. The DDA and the Bresenham's algorithms [1] are the implementation of this approach which intensifies one of the two points $(x, \lceil y \rceil)$ or $(x, \lfloor y \rfloor)$ with constant brightness. In this case, the curve shown in the raster display has "staircase effect". This annoying visual effect is known as aliasing.

An antialiasing technique involves low-pass filtering. A filter function is designed to assign proper intensities on the pixels close to the exact curve path to get a fuzzy edge. An example of this approach is of Wu [2]. Wu proposed an algorithm which simulates all the sampled points by a two-pixel wide band bounding the true curve $y = f(x)$. Wu's algorithm is similar to Bresenham's one which samples the integer points x and calculates $f(x)$. Both $(x, \lceil f(x) \rceil)$ and $(x, \lfloor f(x) \rfloor)$ receive intensities which are assigned inversely proportional to the distance between the pair of pixels to $(x, f(x))$ as

$$I_{(x, \lceil f(x) \rceil)} = I(f(x) - \lfloor f(x) \rfloor),$$
$$I_{(x, \lfloor f(x) \rfloor)} = I(\lceil f(x) \rceil - f(x)), \qquad (1)$$

where I Eq. (1) is the "intended intensity" for the sampled point $(x, f(x))$ and $I_{(x,y)}$ is the "received intensity" of pixel (x, y).

Many people have concentrated on designing filter functions [3–5]. Very often, the computational cost involved in producing a good filter is very high. For

☆This work was supported in part under contracts NSC-81-0408-E009-208 and NSC-81-0408-E009-534, National Science Council, Taiwan, Republic of China.

* Corresponding author. Fax: + 886-3-572-1490.

E-mail address: ytching@cis.nctu.edu.tw (Y.-T. Ching).

0097-8493/01/$ - see front matter
PII: S0097-8493(00)00123-0

example, Gupta, Sproull and Barkans used a Conical function and a Hamming function, respectively, and they precomputed a set of filter values which are stored in a look-up table.

Each of the above-mentioned methods samples integer values x then distributes intensities to the pixels neighbouring to $(x,f(x))$ according to their own filter functions.

In this paper, we propose to simulate a sampled point (x,y) by the four pixels around (x,y) where x and y are not necessarily to be integers. Based on this method, we show that sampling infinite number of points along a line is possible since the closed-form solutions for the received intensities of pixels can be derived.

In the next section, we shall first introduce the proposed low-pass filtering function and define the intended intensity. In Section 3, the closed form solutions for the received intensities of pixels are derived and Section 4 concludes.

2. Preliminary

In this section, we present the method to simulate a sampled point by the four pixels around the sampled point. We also define the intended intensity in this section.

A unit square, denoted $u_{(i,j)}$, in the raster display is a square with four pixels $(i,j),(i,j+1),(i+1,j)$ and $(i+1,j+1)$ as vertices. Consider a point $p=(x,y)$ in $u_{(\lfloor x\rfloor,\lfloor y\rfloor)}$ where x and y are not integers as shown in Fig. 1. We simulate the point p by the four pixels of $u_{(\lfloor x\rfloor,\lfloor y\rfloor)}$ as the following. Let I be the intended intensity of p and y_a, y_b, x_a and x_b be

$$\begin{aligned} y_a &= y-\lfloor y\rfloor,\\ y_b &= \lceil y\rceil - y,\\ x_a &= x-\lfloor x\rfloor,\\ x_b &= \lceil x\rceil - x. \end{aligned} \tag{2}$$

The received intensities of the four pixels are obtained by first distributing I inversely proportional to the distance in the vertical direction to t_1 and t_2 (Fig. 1), then distributing I_{t_1} and I_{t_2} inversely proportional to the distance in the horizontal direction to the four pixels. The derived received intensities of the four pixels are

$$I_{(\lfloor x\rfloor,\lfloor y\rfloor)} = I x_b y_b, \tag{3}$$

$$I_{(\lfloor x\rfloor,\lceil y\rceil)} = I x_b y_a, \tag{4}$$

$$I_{(\lceil x\rceil,\lceil y\rceil)} = I x_a y_a, \tag{5}$$

$$I_{(\lceil x\rceil,\lfloor y\rfloor)} = I x_a y_b, \tag{6}$$

The received intensity of a pixel is I times the area of the rectangle opposite to the pixel with respect to the sampled point p (Fig. 1).

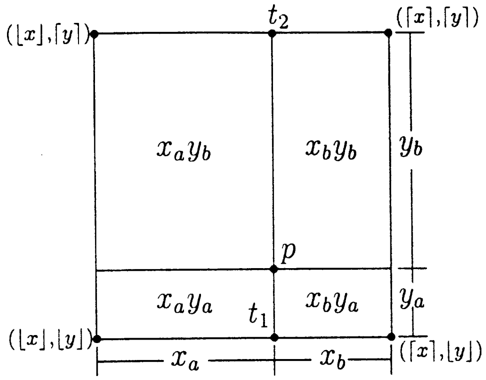

Fig. 1. A sampled point $p=(x,y)$ on a line segment in $u_{(\lfloor x\rfloor,\lfloor y\rfloor)}$. x and y are not integers.

The intended intensity I is designed to display line segments with different slopes at the same brightness level [6]. Let $l((x_0,y_0),(x_1,y_1))$ denote the line segment with two end points (x_0,y_0) and (x_1,y_1). Consider two line segments $S_1=l((0,0),(X,0))$ and $S_2=l((0,0),(X,X))$. Since the length in Euclidean distance of S_2 is equal to $\sqrt{2}$ times the length of S_1, the number of pixels used to simulate S_2 is $\sqrt{2}$ times the number of pixels used to simulate S_1. If we sample N points in the interval $[x_0,x_1]$ and each sampled point has the same intended intensity I, then the pixels simulating S_2 receive less intensity than the pixels simulating S_1 do. This problem can be fixed by giving different intended intensities to the sampled points on the lines with different slopes.

Let the line segment $\overline{p_1,p_2}=l((x_1,y_1),(x_2,y_2))$. We define the intended intensity, I_0 of a unit square to be

$$I_0 = \frac{|\overline{p_1,p_2}|_2}{|\overline{p_1,p_2}|_\infty}. \tag{7}$$

In Eq. (7), $|\overline{p_1,p_2}|_2$ is the length of $\overline{p_1,p_2}$ in L_2 metric (the Euclidean distance) and $|\overline{p_1,p_2}|_\infty$ is the length of $\overline{p_1,p_2}$ in L_∞ metric. $|\overline{p_1,p_2}|_\infty$ is $\max(|x_2-x_1|,|y_2-y_1|)$ which is the number of points sampled in any of the previous scan conversion line drawing algorithms.

Suppose we are drawing a line $\overline{p_1,p_2}$ with slope $0\leqslant m\leqslant 1$ by using the proposed filtering function. We can sample n points within a unit square, each point Δx distance apart along x-direction. The intended intensity for a sampled point is then I_0/n which should be distributed to the four pixels around it by Eqs. (3)–(6). Since a pixel receives contribution from many sampled points, the received intensity of a pixel is obtained by summing the contributions from all the sampled points. We shall show that it is possible to achieve the effect of the sample at infinite number of points since there are closed forms for the received intensity.

3. Closed forms

In this section, we derive the closed forms of the received intensities of pixels. For ease of presentation, we consider only the case of drawing line segments having slopes $m, 0 \leqslant m \leqslant 1$ (because the other cases are symmetric). We also assume that the line segment is specified by two end points with integer coordinates. Thus, we can assume, without loss of generality, that the line segment is $l((0,0),(x,y)), x > 0, y \geqslant 0$. The slope of $l((0,0),(x,y))$ is then $m = y/x$. Since $0 \leqslant m \leqslant 1, x \geqslant y$.

Let $\{(i,f(i)) | i = 0,\ldots,x\}$ be the set of point on $l((0,0),(x,y))$ with integer x-coordinates. Each line segment $l((i,f(i)),(i+1,f(i+1)))$ is an *element*, denoted s_i. Consider an element s_i. If $\lfloor f(i+1) \rfloor = \lfloor f(i) \rfloor$ or $\lceil f(i+1) \rceil = \lceil f(i) \rceil$ then s_i is totally in the unit square $u_{(i,\lfloor f(i) \rfloor)}$. We call this Case 1. Case 2 occurs when $\lfloor f(i+1) \rfloor = \lfloor f(i) \rfloor + 1$. In this case, s_i passes through two unit squares, namely $u_{(i,\lfloor f(i) \rfloor)}$ and $u_{(i,\lfloor \lceil f(i) \rceil)}$. Since $x \geqslant y, f(i+1) - f(i) \leqslant 1$, these must be the only two cases. The closed forms of the received intensities of pixels for these two cases are discussed in the following.

3.1. Case 1

For a case 1 element s_i in the unit square $u_{(i,\lfloor f(i) \rfloor)}$ as shown in Fig. 2, there are four vertices which receive intensities from the element $s_i = l((i,f(i)),(i+1,f(i+1)))$. Let $h = f(i) - \lfloor f(i) \rfloor$.

Theorem 1. *If we sample an infinite number of points along s_i, the received intensities of the four vertices of $u_{(i,\lfloor f(i) \rfloor)}$ are*

$$I_{(i,\lfloor f(i) \rfloor)} = I_0\left(\frac{1}{2} - \frac{m}{6} - \frac{h}{2}\right),$$

$$I_{(i,\lceil f(i) \rceil)} = I_0\left(\frac{m}{6} + \frac{h}{2}\right),$$

$$I_{(i+1,\lceil f(i) \rceil)} = I_0\left(\frac{m}{3} + \frac{h}{2}\right),$$

$$I_{(i+1,\lfloor f(i) \rfloor)} = I_0\left(\frac{1}{2} - \frac{m}{3} - \frac{h}{2}\right).$$

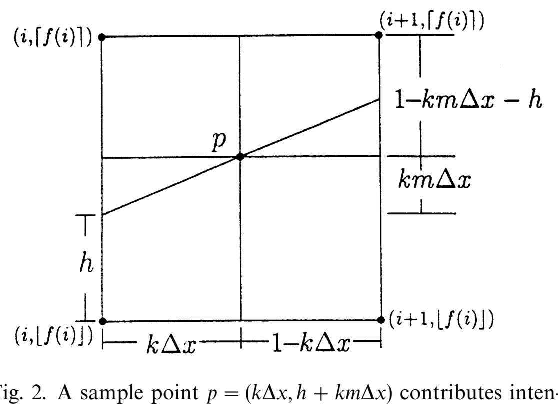

Fig. 2. A sample point $p = (k\Delta x, h + km\Delta x)$ contributes intensities to the four pixels on $u_{(i,\lfloor f(i) \rfloor)}$.

Proof. Assume that we sample n points along s_i. The intended intensity of a sampled point is I_0/n. The received intensity for each pixel of $u_{(i,\lfloor f(i) \rfloor)}$ is obtained by accumulating the products of I_0/n and the area of the rectangles opposite to the sampled points for all the n sampled points (Fig. 2). The intensity contributed from s_i to the pixel $(i,\lfloor f(i) \rfloor)$ is

$$\begin{aligned} I_{(i,\lfloor f(i) \rfloor)} &= \sum_{k=0}^{n-1} \frac{I_0}{n}(1 - k\Delta x)(1 - km\Delta x - h) \\ &= \frac{I_0}{n}\sum_{k=0}^{n-1}(1 - k\Delta x - km\Delta x + k^2(\Delta x)^2 m - h + hk\Delta x) \\ &= \frac{I_0}{n}\left(n - \frac{n(n-1)}{2}\Delta x - \frac{n(n-1)}{2}m\Delta x \right. \\ &\quad \left. + \frac{n(n-1)(2n-1)}{6}(\Delta x)^2 m - nh + \frac{n(n-1)}{2}h\Delta x\right). \end{aligned}$$

If we sample an infinite number of points, then we have

$$\begin{aligned} I_{(i,\lfloor f(i) \rfloor)} &= \lim_{n\to\infty} \frac{I_0}{n}\left(n - \frac{n(n-1)}{2}\Delta x - \frac{n(n-1)}{2}m\Delta x \right. \\ &\quad \left. + \frac{n(n-1)(2n-1)}{6}(\Delta x)^2 m - nh + \frac{n(n-1)}{2}h\Delta x\right) \\ &= I_0\left(1 - \frac{1}{2} - \frac{m}{2} + \frac{m}{3} - h + \frac{h}{2}\right) \quad \text{since } \Delta x = \frac{1}{n} \\ &= I_0\left(\frac{1}{2} - \frac{m}{6} - \frac{h}{2}\right) \end{aligned}$$

$I_{(i,\lceil f(i) \rceil)}$, $I_{(i+1,\lceil f(i) \rceil)}$, and $I_{(i+1,\lfloor f(i) \rfloor)}$ can be obtained in a similar manner.

$$\begin{aligned} I_{(i,\lceil f(i) \rceil)} &= \lim_{n\to\infty} \sum_{k=0}^{n-1} \frac{I_0}{n}(1 - k\Delta x)(h + km\Delta x) \\ &= \lim_{n\to\infty} \frac{I_0}{n}\sum_{k=0}^{n-1}(h - hk\Delta x + km\Delta x - k^2(\Delta x)^2 m) \\ &= \lim_{n\to\infty} \frac{I_0}{n}\left(nh - \frac{n(n-1)}{2}h\Delta x + \frac{n(n-1)}{2}m\Delta x \right. \\ &\quad \left. - \frac{n(n-1)(2n-1)}{6}(\Delta x)^2 m\right) \\ &= I_0\left(h - \frac{h}{2} + \frac{m}{2} - \frac{m}{3}\right) \\ &= I_0\left(\frac{m}{6} + \frac{h}{2}\right), \end{aligned}$$

$$I_{(i+1,\lceil f(i)\rceil)} = \lim_{n\to\infty} \frac{I_0}{n}\sum_{k=0}^{n-1}((k\Delta x)(km\Delta x + h))$$

$$= \lim_{n\to\infty} \frac{I_0}{n}\sum_{k=0}^{n-1}(k^2(\Delta x)^2 m + k\Delta x h)$$

$$= \lim_{n\to\infty} \frac{I_0}{n}\left(\frac{n(n-1)(2n-1)}{6}(\Delta x)^2 m + \frac{n(n-1)}{2}\Delta x h\right)$$

$$= I_0\left(\frac{m}{3}+\frac{h}{2}\right),$$

$$I_{(i+1,\lfloor f(i)\rfloor)} = \lim_{n\to\infty} \sum_{k=0}^{n-1}\frac{I_0}{n}((k\Delta x)(1 - km\Delta x - h))$$

$$= \lim_{n\to\infty} \sum_{k=0}^{n-1}\frac{I_0}{n}(k\Delta x - k^2(\Delta x)^2 m - hk\Delta x)$$

$$= \lim_{n\to\infty} \frac{I_0}{n}\left(\frac{n(n-1)}{2}\Delta x - \frac{n(n-1)(2n-1)}{6}m(\Delta x)^2 - \frac{n(n-1)}{2}h\Delta x\right)$$

$$= I_0\left(\frac{1}{2}-\frac{m}{3}-\frac{h}{2}\right). \qquad \square$$

3.2. Case 2

Let s_j be an element of case 2. s_j passes through two unit squares, $u_{(j,\lfloor f(j)\rfloor)}$ and $u_{(j,\lceil f(j)\rceil)}$. There are about six pixels which receive intensities from s_j. We now show that the closed forms for the received intensities of the six pixels can still be derived.

Theorem 2. *Let $s_j = \overline{q_1, q_3}$ be a case 2 element passing through two unit squares $u_{(j,\lfloor f(j)\rfloor)}$ and $u_{(j,\lceil f(j)\rceil)}$ as shown in* Fig. 3. *q_2 is the intersection between $\overline{q_1, q_3}$ and $y = \lceil f(j)\rceil$. Let $L = X(q_2) - j$ and $c = f(j) - \lfloor f(j)\rfloor$ where $X(q_i)$ denotes the x-coordinate of q_i. The received intensities of six pixels are, respectively, the following:*

$$I_{(j,\lfloor f(j)\rfloor)} = I_0\left(\frac{m}{2}L^2 - \frac{m}{6}L^3\right),$$

$$I_{(j,\lceil f(j)\rceil)} = I_0\left(L - \frac{m}{2}L^2 - \frac{L^2}{2} + \frac{m}{6}L^3\right) + I_0\left(\frac{(1-L)^2}{2} - \frac{m}{6}(1-L)^3\right),$$

$$I_{(j,\lceil f(j)\rceil+1)} = I_0\left(\frac{m}{6}(1-L)^3\right),$$

$$I_{(j+1,\lceil f(j)\rceil+1)} = I_0\left(\frac{m}{6}L^3\right),$$

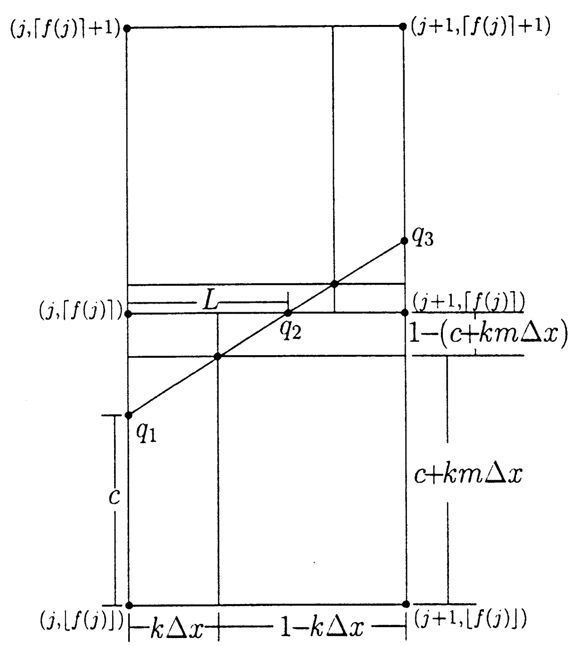

Fig. 3. $\overline{q_1, q_3}$ a case 2 element, passes through two unit squares $u_{(j,\lfloor f(j)\rfloor)}$ and $u_{(j,\lceil f(j)\rceil)}$. $c = f(j) - \lfloor f(i)\rfloor$ and $L = x(q_2) - j$.

$$I_{(j+1,\lceil f(j)\rceil)} = I_0\left(\frac{L^2}{2} - \frac{m}{6}L^3\right) + I_0\left(L(1-L) + \frac{(1-L)^2}{2} - \frac{m}{2}L(1-L)^2 - \frac{m}{3}(1-L)^3\right),$$

$$I_{(j+1,\lfloor f(j)\rfloor)} = I_0\left(\frac{m}{2}L(1-L)^2 + \frac{m}{3}(1-L)^3\right),$$

Proof. The complete proof is a lengthy and tedious work. We only derive the received intensities for $(j, \lfloor f(j)\rfloor)$ and $(j, \lceil f(j)\rceil + 1)$ in the proof.

The line segment $\overline{q_1, q_2}$ contributes intensities to the four pixels of $u_{(j,\lfloor f(j)\rfloor)}$ and $\overline{q_2, q_3}$ contributes intensities to the four pixels of $u_{(j,\lceil f(j)\rceil)}$. Since, we have sampled n points from q_1 to q_3, we assume that there are r and s points sampled from q_1 to q_2 and q_2 to q_3, respectively. Since $L = X(q_2) - X(q_1)(1 - L = X(q_3) - X(q_2))$, the intended intensity of $\overline{q_1, q_2}(\overline{q_2, q_3})$ is $I_0' = LI_0(I_0'' = (1-L)I_0)$. Note also that we sample $r(s)$ points along $\overline{q_1, q_2}$ $(\overline{q_2, q_3})$, each pair of consecutive sampled point is Δx apart, thus $L = r\Delta x(1 - L = s\Delta x)$.

The pixel $(j, \lfloor f(j) \rfloor)$ receives intensity contributed from $\overline{q_1, q_2}$. We have

$$I_{(j, \lfloor f(j) \rfloor)} = \lim_{r \to \infty} \frac{I'_0}{r} \left(\sum_{k=0}^{r-1} (1 - (c + km\Delta x))(1 - k\Delta x) \right)$$

$$= \lim_{r \to \infty} \frac{I'_0}{r} \left(\sum_{k=0}^{r-1} (1 - k\Delta x - c + ck\Delta x - k\Delta x m + k^2 (\Delta x)^2 m) \right)$$

$$= \lim_{r \to \infty} \frac{I'_0}{r} \left(r - \Delta x \frac{r(r-1)}{2} - rc + c\Delta x \frac{r(r-1)}{2} - \Delta x m \frac{r(r-1)}{2} + (\Delta x)^2 m \frac{r(r-1)(2r-1)}{6} \right)$$

$$= I'_0 \left(1 - \frac{L}{2} - c + \frac{cL}{2} - \frac{mL}{2} + \frac{mL^2}{3} \right)$$

$$= I_0 \left(L - \frac{L^2}{2} - cL + \frac{cL^2}{2} - \frac{mL^2}{2} + \frac{mL^3}{3} \right)$$

$$= I_0 \left(L(1-c) - (1-c)\frac{L^2}{2} - \frac{mL^2}{2} + \frac{mL^3}{3} \right)$$

$$= I_0 \left(LLm - Lm\frac{L^2}{2} - \frac{mL^2}{2} + \frac{mL^3}{3} \right)$$

since $1 - c = Lm$

$$= I_0 \left(\frac{mL^2}{2} - \frac{mL^3}{6} \right).$$

The pixel $(j, \lceil f(j) \rceil + 1)$ receives contribution from $\overline{q_2, q_3}$ and the received intensity can be derived in a similar way.

$$I_{(j, \lceil f(j) \rceil + 1)} = \lim_{s \to \infty} \frac{I''_0}{s} \left(\sum_{k=0}^{s-1} (1 - (L + k\Delta x)) km\Delta x \right)$$

$$= \lim_{s \to \infty} \frac{I''_0}{s} \left(\sum_{k=0}^{s-1} (km\Delta x - Lkm\Delta x - k^2 m (\Delta x)^2) \right)$$

$$= \lim_{s \to \infty} \frac{I''_0}{s} \left(m\Delta x \frac{s(s-1)}{2} - m\Delta x L \frac{s(s-1)}{2} - m(\Delta x)^2 \frac{s(s-1)(2s-1)}{6} \right)$$

$$= I''_0 \left(\frac{m(1-L)}{2} - \frac{mL(1-L)}{2} - \frac{m(1-L)^2}{3} \right)$$

$$= I_0 \left(\frac{m(1-L)^2}{2} - \frac{mL(1-L)^2}{2} - \frac{m(1-L)^3}{3} \right)$$

$$= I_0 \left(\frac{m}{6}(1-L)^3 \right).$$

The pixels $(j+1, \lceil f(j) \rceil + 1)$ and $(j+1, \lfloor f(j) \rfloor)$ receive intensities from $\overline{q_2, q_3}$ and $\overline{q_1, q_2}$, respectively. The received intensities of these two pixels can be derived in a way similar to that stated above. $(j, \lceil f(j) \rceil)$ and $(j+1, \lceil f(j) \rceil)$ receive contributions from both $\overline{q_1, q_2}$, and $\overline{q_2, q_3}$. Therefore, the received intensities of these two pixels are the sum of the contributions from the two. □

As shown in Theorems 1 and 2, the received intensities of pixels depend on slope but have nothing to do with the number of points sampled. We can achieve the effect of sampling at infinite number of points by simple arithmetic operations.

The above theorems give a simple line drawing algorithm. For each element from 0 to $x-1$, we first determine to which case the element belongs. We then compute the contributions from the element to the vertices by the closed forms given above. The efficiency of this implementation does not depend on the number of points sampled.

In what follows, we show that the efficiency of the algorithm can be further improved. We show that we can calculate the contributions of case 1 elements with less cost if the contributions of the previous case 1 elements are available. For case 2 elements, we show that the received intensities of 2 of the 6 vertices can be obtained by simple arithmetic.

Let s_i and s_j be a pair of consecutive case 1 elements. If the contributions of s_i to the four vertices are known, the contributions of s_j to the four vertices can be obtained by the following corollary.

Corollary 3. *Given two parallel line segments* $\overline{p_1, p_2}$ *and* $\overline{q_1, q_2}$, *are totally in a unit square* u_{v_0}. *Let* h *be the distance between these parallel line segments in the y-direction* (Fig. 4). *The differences in the received intensities of* v_i

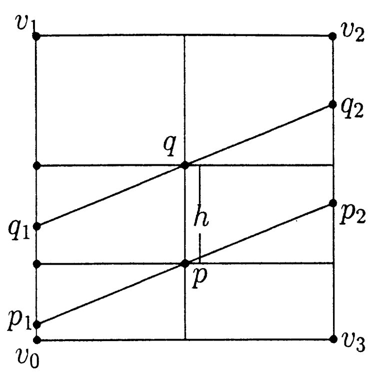

Fig. 4. If we place two case 1 elements in the same unit square. Let $p = (x, y)$ and $q = (x, y + h)$. The differences of the contributions between these elements are the sum of the differences of the contributions between p and q.

contributed from $\overline{p_1, p_2}$ *and* $\overline{q_1, q_2}$, *denoted* $\Delta_i(h), i = 0, \ldots, 3,$ *are*

$$\Delta_0(h) = -\frac{h}{2},$$

$$\Delta_1(h) = \frac{h}{2},$$

$$\Delta_2(h) = \frac{h}{2},$$

$$\Delta_3(h) = -\frac{h}{2},$$

Proof. Immediate from Theorem 1. □

Corollary 3 shows that the differences of the contributions between two case 1 elements are a function of the distance h. Since the first element s_0 is a case 1 element. We can always compute the contributions of case 1 element by subtraction from the contributions of s_0.

A more efficient method to calculate the contributions of a case 2 element is obtained by simple arithmetic as shown in the following corollary:

Corollary 4. *For a case* 2 *element passing through two unit squares* $u_{(j, \lfloor f(j) \rfloor)}$ *and* $u_{(j, \lceil f(j) \rceil)}$,

$$\frac{I_0}{2} = I_{(j, \lfloor f(j) \rfloor)} + I_{(j, \lceil f(j) \rceil)} + I_{(j, \lceil f(j) \rceil + 1)}$$

$$= I_{((j+1, \lfloor f(j) \rfloor)} + I_{(j+1, \lceil f(j) \rceil)} + I_{(j+1, \lceil f(j) \rceil + 1)}.$$

Proof. Immediate from Theorem 2. □

Since $I_{(j, \lceil f(j) \rceil)}$ and $I_{(j+1, \lceil f(j) \rceil)}$ are the two most expensive terms to evaluate, we calculate these two terms by applying Corollary 4.

4. Conclusion

In this paper, we presented a method for antialiased line drawing. The method is based on a proposed low-pass filtering function. Under the proposed low-pass filtering function, we can achieve the effect of sampling at infinite number of points along a line segment since the closed forms of the received intensities of pixels exist. We show that the cost for computing the received intensities of pixels can be reduced by applying simple rules. We also propose a way to define the intended intensity which can ensure that line segments with different slopes are displayed with the same brightness.

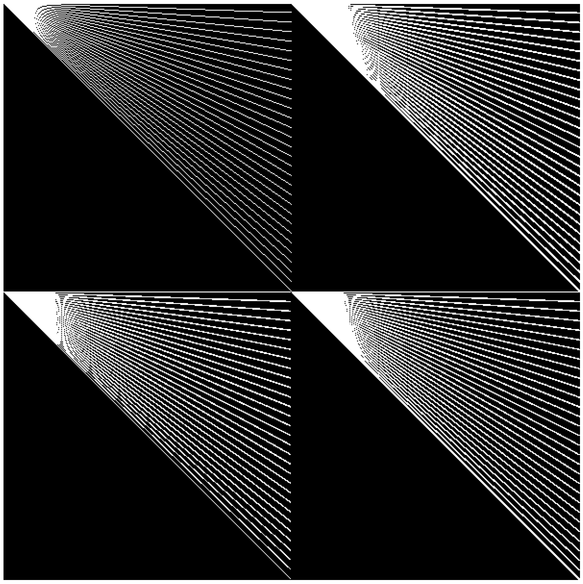

Fig. 5. Lines produced using different algorithms, 1. upper left: no antialiasing applied, 2. upper right: Gupta and Sproull approach, 3. lower left: Wu's algorithm, 4. lower right: the proposed algorithm.

It is difficult to give a mathematical judgment to verify the quality of antialiasing effects in an image. Furthermore, there are many factors, for example the gamma correction, that could affect the antialiasing effect. The antialiasing effect is generally judged by human perception. Fig. 5 shows the lines obtained using different algorithms. The upper left image shows that the line segments were drawn without any antialiasing technique applied. The upper right image contains the line segments obtained using Sproull and Gupta approach. The lower left image shows line segments produced using Wu's algorithm. The line segments obtained using the proposed algorithm are shown in the lower right image. Gamma correction was applied using the equation

$$I' = I^{(1/\gamma)}$$

given in [7] where $\gamma = 2.3$. The proposed method can achieve very good antialiasing effect.

References

[1] Bresenham JE. Algorithm for computer control of digital plotter. IBM Systems Journal 1965;4(1):25–30.

[2] Wu X. An efficient antialiasing technique. Computer Graphics 1991;25(4):143–52.

[3] Barkans AC. High speed high quality antialiased vector generation. Computer Graphics 1990;24(4):319–26.
[4] Crow F. The aliasing problem in computer-generated shaded images. Communications of the ACM 1977;20:11.
[5] Gupta S, Sproull R. Filtering edges for gray-scale displays. Computer Graphics 1981;15(3):1–5.
[6] Hearn D, Baker MP. Computer graphics. Englewood Cliffs, NJ: Prentice-Hall, 1986.
[7] Watt A. Fundamentals of three-dimensional computer graphics. Reading, MA: Addision-Wesley, 1991. p. 350.