A Simple Anti-Aliasing Method
For Straight Line Drawing Based On DSP Platform

Dusheng Wang
Institute of Electromechanical & Control Engineering
Dalian Nationalities University
Dalian Development Zone, Liao He West Road 18
Dalian City, Liaoning Province, P.R.China, 116600
dusheng3791@hotmail.com

Hock Lye Toh, Xiang Chen, Fan Yang

Centre for Signal Processing
Nanyang Technological University
Nanyang Drive 637722, Singapore
{ ehltoh, exchen, efyang}@ntu.edu.sg

ABSTRACT

A simple and practical anti-aliasing method for a color straight line drawing is presented in this paper. The method has been applied in a DSP-based display system to remove the undesired jaggies occurred in the line drawing. The experimental results show that this method can produce a good visual effect on the low resolution display screen.

Keywords: anti-aliasing, straight line drawing, DSP-based system, raster graphics

1. INTRODUCTION

When we try to draw a line on a low resolution screen , we are certainly to see the ugly jagged "stair-steps", this is because the screen does not have a enough resolution to display a line smoothly. Anti-aliasing is a way to use color information to make up for a lack of screen resolution. Among the existed methods, Wu's algorithm[Wu91] is mostly used and efficient. When the background is black, the anti-aliasing effect with Wu's method is rather good in visual aspect. But when the background is not black, the visual effect will become worse. This paper presents a modified anti-aliasing algorithm based on Bresenham's algorithm. It deployed the Wu's idea that draws the two pixels bracketing the line along the minor axis at each point. But each pixel's drawing intensity is assigned the weighting combination of drawing color and background color at that point. Section 2 will describe our proposed method in details. Section 3 and 4 are experimental result and conclusion.

2. PROPOSED METHOD

Our proposed method for anti-aliasing is based on Bresenham's line drawing algorithm. When an x-majored line is plotted using it, all pixels that are overlapped by the line area are displayed with an intensity proportional to the area of overlap Instead of plotting the line with a single pixel at each x position. There are various ways to calculate the overlapping area[Gupta81, Foley90]. In our approach, a simplified technique is adopted. For a x-majored line, on each column, the line overlaps at most two pixels. The ratio of these two overlapped areas is proportional to the ratio of (1-D_1) and (1-D_2), as depicted in Fig. 1.

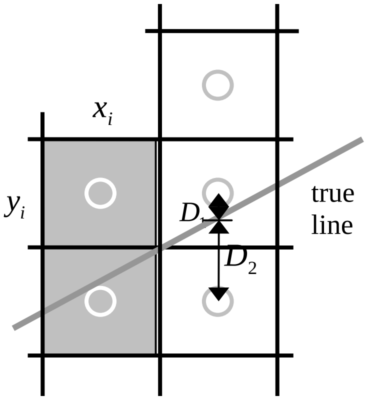


Fig. 1 Vertical distance of measurement for a x-majored line

To satisfy the requirement of change in background color, we choose the weighting combination between drawing color and background color, that is, pixel ($x_i + 1, y_i$) will be assigned an intensity of being equal to the sum of (1-D_1) times intensity of drawing color and D_1 times that of background color; while

Posters proceedings ISBN 80-86943-04-6
WSCG'2006, January 30-February 3, 2006
Plzen, Czech Republic.

$(x_i + 1, y_i - 1)$ will be assigned an intensity of being equal to the sum of (1-D_2) times intensity of drawing color and D_2 times that of background color. For y-majored line, this intensity division will be applied to two neighboring pixels bracketing it in a row. For the RGB color line, the following equations express clearly the weighting intensity operation on it:

$$\begin{aligned} R &= (1 - D_1) * R_d + D_1 * R_b \\ G &= (1 - D_1) * G_d + D_1 * G_b \qquad \mathit{for}\ (x_i + 1, y_i) \\ B &= (1 - D_1) * B_d + D_1 * B_b \end{aligned}$$

$$\begin{aligned} R &= (1 - D_2) * R_d + D_2 * R_b \\ G &= (1 - D_2) * G_d + D_2 * G_b \qquad \mathit{for}\ (x_i + 1, y_i - 1) \\ B &= (1 - D_2) * B_d + D_2 * B_b \end{aligned}$$

where R, G, B is the resulting red, green, blue component of the drawing point, R_d, G_d, B_d is the red, green, blue component of the drawing color, while R_b, G_b, B_b is the red, green, blue component of the background, and also it exists $D_1 + D_2 = 1$ in the distance measurement.

3. EXPERIMENTAL RESULTS

Based on DSP(SHARC ADSP21060 chip) platform, we developed a Video Graphics System with the support of dedicated OpenGL[Neider93] driver. The most common 2-Dimension related basic OpenGL functions had been implemented on this platform. Drawing line function was finished based on Bresenham's algorithm, and our proposed anti-aliasing method was added to remove the jagged "stair-step" in the straight line drawn. In our raster display screen with 640×480 resolution, some graphic objects would be required to be displayed on the video picture or different intensity level of background. When we draw a line on the white background and use Wu's anti-aliasing method, the anti-aliasing effect looks like as Fig.2 shown, Fig.2 is depicted as follows: a line with the length of 60 pixels was rotating around the end point at the center of screen at the interval of 10 degree every frame. We got 18-19 frames' data from the display buffer in the real-time running simulation platform and plotted it with MATLAB.

As the background's white color is not used to do color mixing, the line looks like a dirty "stair-step" edge. Fig.3 depicts the effect with our proposed anti-aliasing method for a straight RGB color line. The data collection and plotting procedures are the same as that in Fig.2. Fig. 3(a) shows different dimensions lines without anti-aliasing. Fig. 3(b) demonstrates different dimensions lines with anti-aliasing.

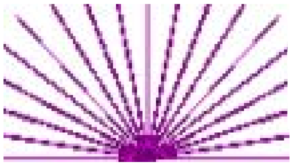
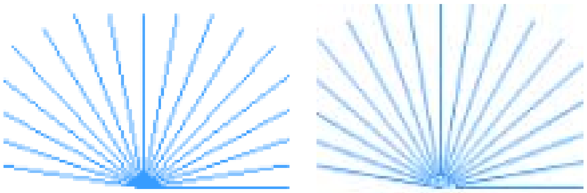


Fig. 2　The example using Wu's anti-aliasing method in white background



(a) The figure without anti-aliasing (b)The figure with anti-aliasing

Fig. 3　The single-pixel-line with different dimensions drawn by Bresenham's algorithm

4. CONCLUSION

This paper presents a simple and practical anti-aliasing method for a color straight line drawing. From the experimental result we can draw the conclusion that our anti-aliasing method is more adaptable for anti-aliasing RGB color line than Wu's method and produces a good visual effect on the low resolution screen. But it is a time-consuming function too. It takes about 60 cycles on DSP platform to write an anti-aliased pixel's color information into display buffer while writing a pixel color information without anti-aliasing is only needed to take one cycle. Therefore, implementing anti-aliasing function had better been finished by hardware module if necessary.

5. REFERENCES

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

[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.

[Neider93] J. Neider, T. Davis, and M. Woo: *OpenGL Programming Guide.* Addison-Wesley, Menlo Park, 1993.

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