基于点到直线距离的直线扫描转换算法

陈幼明
(西北师范大学 数学与信息科学学院,甘肃 兰州 730070)

摘 要:直线扫描转换算法是计算机图形学和计算机辅助设计等领域最基本、最重要的算法之一,直线反走样算法也是光栅化图形算法中的重要内容。文中提出了一种基于点到直线距离的直线扫描转换算法,给出了算法的推导过程及代码表示,并介绍了算法在直线反走样中的具体应用。该算法基于增量技术,采用点到直线的距离作为判别式,在扫描转换过程中,可方便地根据点到直线的距离,采用加权区域采样的方法进行直线的反走样,提高了反走样的效率;具有只使用整型变量、不涉及乘除运算的特点,适合硬件实现。

关键词:直线扫描转换;算法;反走样
中图分类号:TP391 **文献标识码**:A **文章编号**:1673-629X(2012)03-0051-04

A Line Scan Conversion Algorithm Based on Distance of Point to Line

CHEN You-ming
(College of Mathematics and Information Science, Northwest Normal University, Lanzhou 730070, China)

Abstract: Line generating algorithm is one of the most important and basic algorithms in computer graphics and computer aided design. Anti-aliasing algorithm is very important content in graphics algorithm. It presents a line scan conversion algorithm based on distance of point to line and gives the algorithm derivation process and code. It introduces specific application of algorithm in line anti-aliasing. This algorithm is a method which uses increment technology. It regards distance of point to line as discriminant, it draws anti-aliasing line by method of weighted region sampling, which is convenient to draw anti-aliasing line according to the distance of point to line on the scan conversion process. It can improve efficiency of anti-aliasing. The algorithm possesses special features which use integer variable, not concerned with multiplication and division. It is suit for hardware implementation.

Key words: line scan conversion; algorithm; anti-aliasing

0 引 言

直线段是最基本的图形,一个复杂的图形,往往由成千上万的直线段组成,即便是圆弧、抛物线等曲线也可由直线段逼近表示;对图形的诸如图形裁剪、图形变换等操作,也都是以直线段的生成为基础的,因此,直线扫描转换算法是计算机图形学和计算机辅助设计等领域最基本、最重要的算法之一。为了减少或消除显示直线时出现的失真,获得更好的视觉效果,需要对输出的直线进行反走样处理。

文中首先对直线扫描转换算法与直线反走样的相关研究进行了介绍,提出了一种基于点到直线距离的直线扫描转换算法,给出了算法的推导过程及其代码表示,最后介绍了该算法在直线反走样中的具体应用。

收稿日期:2011-08-09;**修回日期**:2011-11-11
基金项目:甘肃省自然科学基金项目(0803RJZA109)
作者简介:陈幼明(1964-),男,陕西城固人,副教授,研究方向为图形图像处理。

1 直线扫描转换算法与直线反走样

1.1 直线扫描转换算法

经典的直线扫描转换算法有DDA算法和Bresenham算法[1]等。最著名的是Bresenham算法,一直被认为是一种简单、高效的直线生成算法。这些算法大多是根据当前已知像素的位置,经过简单地计算来推出下一个像素的位置,从而得到线段上的每一点,并将其显示输出。随着计算机应用技术的发展,对图形绘制的效率提出了越来越高的要求,寻求更有效的直线扫描转换算法日益迫切,因此,产生了各种改进算法或加速算法,这些算法从不同角度进行了改进,多为一次能生成多个像素点的多步直线扫描转换算法或并行算法[2~7],进一步提升了直线扫描转换的效率。

1.2 直线反走样

在光栅显示系统上显示图形时,直线段或图形边

界大多会呈现锯齿状,这种用离散的量(像素)表示连续的量(图形)而引起的失真,叫做走样。显示图形时,需要消除或减小走样的程度,即需要进行反走样。

对于直线的反走样常采用提高分辨率、简单区域采样和加权区域采样等方法[8],以及后续的大量研究成果所提出的反走样算法[9~13]等,以提高反走样效率和效果。其中,加权区域采样方法对简单区域采样方法进行了改进,使得相交区域对像素亮度的贡献依赖于该区域与像素中心的距离,对于相同面积的相交区域,当它距离像素中心近时,它对像素亮度的贡献大,当它远离像素中心时,对像素亮度的贡献小。这种处理方法更符合人的视觉系统对图像信息的处理方式,反走样的效果更好一些。

采用加权区域采样的直线反走样处理过程为:以斜率在0、1之间的直线为例,在扫描转换算法中,利用判别式的符号确定下一点,进一步确定与之上下相邻的两个点,计算直线到这三个点的距离,通过对圆锥形滤波器进行积分或者查表获得每个需处理点的加权区域采样的近似值,用相应的灰度值对其进行设置[8]。

若采用Bresenham算法等扫描转换算法绘制直线段,并进行反走样,需要额外计算每个点到直线的距离。下面提出一种基于点到直线距离的直线扫描转换算法,采用点到直线的距离作为判别式,在扫描转换的过程中,直接得到相应的距离,进行反走样处理。

2 基于点到直线距离的直线扫描转换算法

2.1 基本原理

与Bresenham算法一样,为讨论方便,假定直线斜率 $k \in [0,1]$。当 $P(x_p,y_p)$ 为直线上已确定的像素点,那么下一个与直线最近的像素只能是右方的 $P_1(x_p+1,y_p)$ 或右上方的 $P_2(x_p+1,y_p+1)$,如图1所示。

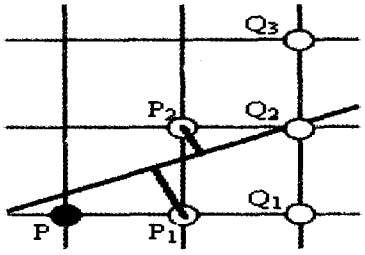

图1 已确定点与可能点示意图

本算法的基本原理是:根据 P_1 到直线的距离 d_1 和 P_2 到直线的距离 d_2 来确定下一像素点是 P_1 还是 P_2。若 $d_1>d_2$,则取点 P_2;若 $d_1<d_2$,则取点 P_1;若 $d_1=d_2$,则点 P_1 与点 P_2 均可,这里约定取点 P_1。

2.2 算法设计

设直线的起点和终点分别为 (x_0,y_0) 和 (x_1,y_1),则直线方程为 $F(x,y)=kx-y+b=0$

其中,$k=\frac{\Delta y}{\Delta x}$,$b=y_0-kx_0$,$\Delta x=x_1-x_0$,$\Delta y=y_1-y_0$。

则 $d_1=\frac{|k(x_p+1)-y_p+b|}{\sqrt{k^2+1}}$,

$d_2=\frac{|k(x_p+1)-(y_p+1)+b|}{\sqrt{k^2+1}}$

当 $d_1-d_2\leqslant 0$ 时,取点 P_1,否则,取点 P_2。

考虑到只需判断 d_1-d_2 的正负,可以将 d_1 和 d_2 简化为下列形式

$d_1=|k(x_p+1)-y_p+b|$,$d_2=|k(x_p+1)-(y_p+1)+b|$

仍然是:当 $d_1-d_2\leqslant 0$ 时,取点 P_1,否则,取点 P_2。

直线与两个可能点的位置关系有三种,如图2所示,下面分别进行讨论。

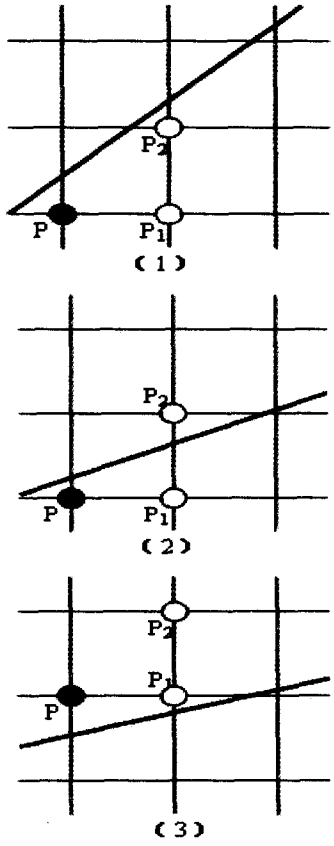

图2 直线与可能点三种位置关系

位置(1):直线在两个可能点的上方。这时有,

$d_1-d_2=|k(x_p+1)-y_p+b|-|k(x_p+1)-(y_p+1)+b|$

$=[k(x_p+1)-y_p+b]-[k(x_p+1)-(y_p+1)+b]>0$

位置(2):直线在两个可能点的中间。这时有,

$d_1-d_2=|k(x_p+1)-y_p+b|-|k(x_p+1)-(y_p+1)+b|$

$=[k(x_p+1)-y_p+b]+[k(x_p+1)-(y_p+1)$

$+b]$

位置(3):直线在两个可能点的上方。这时有,

$d_1-d_2=|k(x_P+1)-y_p+b|-|k(x_P+1)-(y_p+1)+b|$

$=-[k(x_P+1)-y_p+b]+[k(x_P+1)-(y_p+1)+b]<0$

记 $e_1=k(x_p+1)-y_p+b$, $e_2=k(x_P+1)-(y_p+1)+b$

这样, d_1-d_2 就可以用 e_1 和 e_2 表示出来。

位置(1)时, $e_1>0$, $e_2>0$, $e_1>e_2$, $d_1-d_2=e_1-e_2>0$ (这时 $e_1+e_2>0$);

位置(2)时, $e_1>0$, $e_2<0$, $d_1-d_2=e_1+e_2$;

位置(3)时, $e_1<0$, $e_2<0$, $|e_1|<|e_2|$, $d_1-d_2=e_2-e_1<0$(这时 $e_1+e_2<0$)。

可以得出下面的结论: e_1+e_2 与 d_1-d_2 具有相同的符号。通过判断 e_1+e_2 的正负也可确定下一像素,即: $e_1+e_2\leqslant 0$ 时,取点 P_1,否则,取点 P_2。

由于 $e_1=k(x_p+1)-y_p+b=\frac{1}{\Delta x}(\Delta y(x_p+1)-y_p\Delta x+b\Delta x)$,

$e_2=k(x_P+1)-(y_p+1)+b=\frac{1}{\Delta x}(\Delta y(x_p+1)-(y_p+1)\Delta x+b\Delta x)$;

可以进一步将 e_1 和 e_2 简化为:

$e_1=\Delta y(x_p+1)-y_p\Delta x+b\Delta x$

$e_2=\Delta y(x_p+1)-(y_p+1)\Delta x+b\Delta x$

下面讨论 e_1 和 e_2 的计算。

若 $e_1+e_2\leqslant 0$,取点 P_1,下一组可能点为 $Q_1(x_p+2,y_p)$ 和 $Q_2(x_p+2,y_p+1)$,则

$e_1'=\Delta y(x_p+2)-y_p\Delta x+b\Delta x=e_1+\Delta y$,

$e_2'=\Delta y(x_p+2)-(y_p+1)\Delta x+b\Delta x=e_2+\Delta y$;

若 $e_1+e_2>0$,取点 P_2,下一组可能点为 $Q_2(x_p+2,y_p+1)$ 和 $Q_3(x_p+2,y_p+2)$,则

$e_1'=\Delta y(x_p+2)-(y_p+1)\Delta x+b\Delta x=e_2+\Delta y$,

$e_2'=\Delta y(x_p+2)-(y_p+2)\Delta x+b\Delta x=e_2+\Delta y-\Delta x$;

最后,确定 e_1 和 e_2 的初始值:

$e_{1初}=\Delta y(x_0+1)-y_0\Delta x+b\Delta x=\Delta y$

$e_{2初}=\Delta y(x_0+1)-(y_0+1)\Delta x+b\Delta x=\Delta y-\Delta x$

2.3 算法代码

```
//x0<x1,0<=k<=1
LineOnDisdanceOfPoint2Line (x0, y0, x1,y1,color)
int x0, y0, x1, y1, color;
{  int d1,d2,x,y,dx,dy;
   dx=x1-x0;dy=y1-y0;
   d1=dy;d2=dy-dx;
x=x0;y=y0;
   while(x<=x1)
   { DrawPixel(x,y,color);
if( d1+d2<=0)
{ d1=d1+dy;d2=d2+dy; }
else
{d1=d2+dy;d2=d2+dy-dx;
y++;}
  x++;
   }
}
```

对于斜率 k 不在[0,1]的情况,可利用直线的对称性,进行相应的处理即可。另外,本算法中,只使用整型变量、不涉及乘除运算,在整个运算中只有整数的加减操作,从而效率高,速度快,适合硬件实现。

3 在直线反走样中的应用

3.1 直线到需处理点距离的确定

以 1.2 中的采用加权区域采样的直线反走样为例。根据 2.2 中 e_1 和 e_2 值的含义及其演变过程,算法代码中的 d_1 和 d_2 的绝对值 $|d_1|$ 和 $|d_2|$ 分别为直线到 P_1 和 P_2 的距离的 $\sqrt{d_x^2+d_y^2}$ (记为 α) 倍,也就是,直线到 P_1 的距离 $=|d_1|/\alpha$,直线到 P_2 的距离 $=|d_2|/\alpha$。 d_1+d_2 的符号决定了反走样需处理的三个点,如图 3 所示,当 $d_1+d_2\leqslant 0$ 时,下一点为 $P_1(x+1,y)$,需处理的三个点为 $P_0(x+1,y-1)$、$P_1(x+1,y)$、$P_2(x+1,y+1)$,直线到三个需处理点的距离分别为 $(2d_x-|d_2|)/\alpha$, $|d_1|/\alpha$, $|d_2|/\alpha$;当 $d_1+d_2>0$ 时,下一点为 $P_2(x+1,y+1)$,需处理的三个点为 $P_1(x+1,y)$、$P_2(x+1,y+1)$、$P_3(x+1,y+2)$,直线到三个需处理点的距离分别为 $|d_1|/\alpha$, $|d_2|/\alpha$, $(2d_x-|d_1|)/\alpha$。

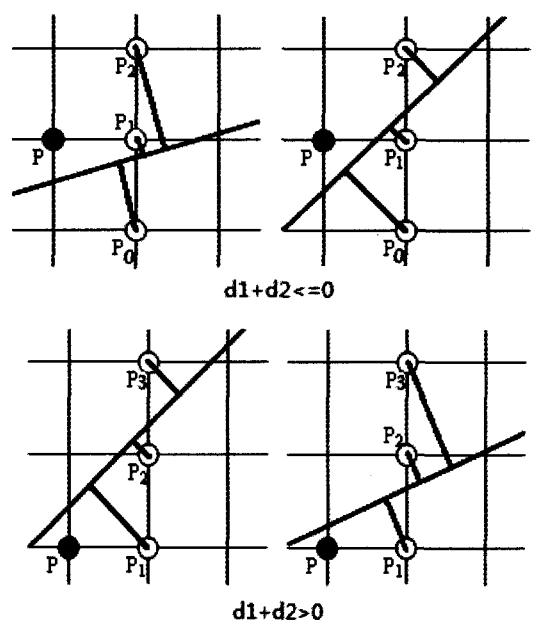

图 3　直线到三个需处理点的距离示意

根据距离，对圆锥滤波器进行积分或者查表获得每个待处理点的灰度值，并对其进行设置。

由于 $|d_1|$、$|d_2|$、$(2d_x-|d_1|)$、$(2d_x-|d_2|)$ 与对应的距离只相差一个因子 $(1/\alpha)$，$|d_1|$、$|d_2|$、$(2d_x-|d_1|)$、$(2d_x-|d_2|)$ 均为整数，因此，可直接根据 $|d_1|$、$|d_2|$、$(2d_x-|d_1|)$、$(2d_x-|d_2|)$ 的值从 0 到 $3d_x/2$ 把圆锥滤波器子体的体积计算出来，放在整型数组 Filter(distance) 中，查表时传递的参数与查表后返回的灰度值都均为整数。表 1 中列出了采用本算法进行扫描转换的过程中，每一步所确定的需处理点及其到直线的距离。

表 1 直线到需处理点距离

当前点	d_1+d_2	下一点	需处理点	距离 $*\alpha$
(x,y)	$\leqslant 0$	$(x+1,y)$	$(x+1,y-1)$	$2d_x-\|d_2\|$
			$(x+1,y)$	$\|d_1\|$
			$(x+1,y+1)$	$\|d_2\|$
	>0	$(x+1,y+1)$	$(x+1,y)$	$\|d_1\|$
			$(x+1,y+1)$	$\|d_2\|$
			$(x+1,y+2)$	$2d_x-\|d_1\|$

3.2 直线反走样算法代码

```
//x0<x1,0<=k<=1
AntiAliasingLine (x0, y0, x1,y1)
int x0, y0, x1, y1;
{  int d1,d2,x,y,dx,dy;
int c1,c2,c3 ;
//Filter 数组定义及赋值略
   dx=x1-x0;dy=y1-y0;
   d1=dy;d2=dy-dx;
x=x0;y=y0;
c1=Filter(abs(dx));
c2=Filter(0);c3=c1;
   while(x<=x1)
       { DrawPixel(x,y-1,c1);
DrawPixel(x,y,c2);
DrawPixel(x,y+1,c3);
      if( d1+d2<=0)
{ c1=Filter(2*dx-abs(d2));
c2=Filter(abs(d1));
c3=Filter(abs(d2));
d1=d1+dy;d2=d2+dy; }
else
{ c1=Filter(abs(d1));
c2=Filter(abs(d2));
c3=Filter(2*dx-abs(d1));
d1=d2+dy;d2=d2+dy-dx;
y++; }
x++;
   }
}
```

在上述直线反走样处理中，除了一次性给数组 Filter 的赋值外，其余处理均使用整型数据，而且不需要专门计算直线到相关点的距离，显示出基于点到直线距离的直线扫描转换算法的优越性。

4 结束语

文中提出了一种新的直线扫描转换算法，该算法基于点到直线的距离，以点到直线的距离作为判别式，采用增量算法，只使用整型变量、不涉及乘除运算，在整个处理中只有整数的加减操作，从而效率高，速度快，适合硬件实现。

该算法应用在采用加权区域采样的方法进行直线的反走样处理中，显示出独特的优越性。在扫描转换的每一步，不需要专门计算直线到需处理点的距离，便可直接查表确定需处理点的灰度值，整个处理过程同样可以做到只涉及整型数据及其加减运算，明显提高了反走样处理的效率。

参考文献：

[1] 孙正兴，周 良，郑宏源. 计算机图形学基础教程[M]. 北京：清华大学出版社，2004：57-61.

[2] 贾银亮，张焕春，经亚枝，等. Bresenham 直线生成算法的改进[J]. 中国图象图形学报，2008，13(1)：158-161.

[3] 郑宏珍. 改进的 Bresenham 直线生成算法[J]. 中国图象图形学报，1999，4(7)：606-608.

[4] 林 笠. 基于 Bresenham 算法的四步画直线算法[J]. 暨南大学学报(自然科学版)，2003，24(5)：19-22.

[5] Wu Xiaolin，Rokne J G. Double-step incremental generation of lines and circles[J]. Computer Vision，Graphics and Image Processing，1987，37(3)：331-344.

[6] 王晓云，王 青. 直线扫描转换中的最小生成机制[J]. 计算机辅助设计与图形学学报，2006，18(3)：433-437.

[7] 孙 岩，唐 棣. 并行的 Bresenham 直线生成算法[J]. 计算机工程与应用，2001，37(21)：136-137.

[8] 孙家广. 计算机图形学[M]. 北京：清华大学出版社，1998：210-215.

[9] Wu Xiaolin. An efficient antialiasing technique[J]. ACM SIGGRAPH Computer Graphics，1991，25(4)：143-152.

[10] 楼剑涛，王秀和. 基于对称的反走样直线生成算法[J]. 计算机工程与应用，2011，47(1)：173-175.

[11] 杭后俊，付 勇. 一种基于加权区域采样的直线反走样生成算法[J]. 计算机技术与发展，2009，19(6)：138-141.

[12] 刘建国. Bresenham 画线反走样算法[J]. 计算机与现代化，2005(2)：10-13.

[13] 贾银亮，张焕春，经亚枝. 基于 FPGA 的直线反走样算法研究[J]. 计算机技术与发展，2011，21(2)：26-29.