案例 5　Wu 反走样直线算法

知识要点

- Wu 反走样算法原理。
- 亮度级别的设置方法。
- 反走样直线类的定义与调用方法。
- 鼠标按键消息映射方法。

一、案例需求

1. 案例描述

在屏幕客户区内按下鼠标左键选择直线的起点，移动鼠标指针到直线终点，弹起鼠标左键绘制任意斜率的反走样直线段。

2. 功能说明

(1) 自定义屏幕二维坐标系，原点位于客户区中心，x 轴水平向右为正，y 轴垂直向上为正。

(2) 设计 CALine 反走样直线类，其成员变量为直线段的起点坐标 P_0 和终点坐标 P_1，成员函数为 MoveTo()和 LineTo()函数。

(3) CALine 类的 LineTo()函数使用中点 Bresenham 算法绘制任意斜率 k 的反走样直线段，包括 $k=\pm\infty$、$k>1$、$0\leqslant k\leqslant 1$、$-1\leqslant k<0$ 和 $k<-1$ 这 5 种情况。

(4) 设置屏幕背景色为白色，反走样直线用灰度表示。

3. 案例效果图

任意斜率的反走样直线段绘制效果如图 5-1 所示。案例 2 绘制的走样直线段与本案例绘制的反走样直线段对比效果如图 5-2 所示。

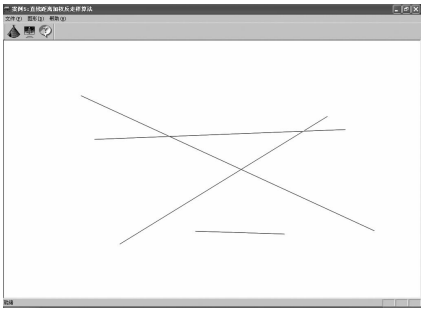

图 5-1　鼠标绘制反走样直线段

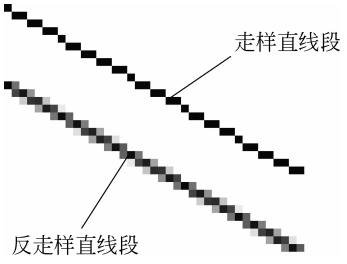

图 5-2　走样直线段与反走样直线段放大对比图

二、案例分析

MFC 提供的 MoveTo()函数只能绘制走样直线段，不能绘制反走样直线段。Wu 算法

是用两个像素来共同表示理想直线上的一个点，依据两个像素与理想直线的距离而调节其亮度，使所绘制的直线达到视觉上消除锯齿的效果。

三、算法设计

(1) 使用鼠标选择直线的 $p_0(x_0,y_0)$、终点坐标 $p_1(x_1,y_1)$。要求起点的 x 坐标小于等于终点的 x 坐标。

(2) 定义直线当前点坐标(x,y)，定义直线下方像素点和直线的距离为 e，定义直线的斜率为 k，定义直线下方的像素点颜色为 c_0，直线上方的像素点颜色为 c_1。

(3) $x=x_0$，$y=y_0$，$e=k$；$k=(y_1-y_0)/(x_1-x_0)$。

(4) 设置像素点(x,y)的亮度为 $c_0=$CRGB(e,e,e)，像素点$(x,y+1)$的亮度为 $c_1=$CRGB$(1.0-e,1.0-e,1.0-e)$。

(5) 计算 $e=e+k$，判断 e 是否大于 1。若 $e>1$，则(x,y)更新为$(x+1,y+1)$，$e=e-1$；否则(x,y)更新为$(x+1,y)$。

(6) 如果 x 小于 x_1，重复步骤(4)和(5)，否则结束。

四、案例设计

1. 设置反走样直线的亮度值

任意斜率 k 包括 $k=\pm\infty$、$k>1$、$0\leqslant k\leqslant 1$、$-1\leqslant k<0$ 和 $k<-1$ 这 5 种情况。对于垂线、水平线和 45°线不必进行反走样，即当 $k=\pm\infty$时直接绘制单像素直线段不进行反走样处理，当 $k=1$ 或 $k=0$ 时可以由 $0\leqslant k\leqslant 1$ 统一处理。反走样直线的亮度值见表 5-1。

表 5-1　任意斜率反走样直线段的相邻像素的亮度值设置

斜率	主位移方向	相邻像素	亮度值
$k>1$	y 方向	x	$c_0=$RGB$(e\times255,e\times255,e\times255)$
		$x+1$	$c_1=$RGB$((1-e)\times255,(1-e)\times255,(1-e)\times255)$
$0\leqslant k\leqslant 1$	x 方向	y	$c_0=$RGB$(e\times255,\ e\times255,\ e\times255)$
		$y+1$	$c_1=$RGB$((1-e)\times255,(1-e)\times255,(1-e)\times255)$
$-1\leqslant k<0$	x 方向	y	$c_0=$RGB$(e\times255,\ e\times255,\ e\times255)$
		$y-1$	$c_1=$RGB$((1-e)\times255,(1-e)\times255,(1-e)\times255)$
$k<-1$	y 方向	x	$c_0=$RGB$(e\times255,\ e\times255,\ e\times255)$
		$x+1$	$c_1=$RGB$((1-e)\times255,(1-e)\times255,(1-e)\times255)$

2. 设计 CALine 反走样直线类

定义反走样直线类绘制任意斜率的反走样直线，其成员函数为 MoveTo()和 LineTo()。

```
class CALine
{
public:
    CALine();
    virtual ~ CALine();
```

```
    void MoveTo(CDC *,CP2);                              //移动到指定位置
    void MoveTo(CDC *,double,double);
    void MoveTo(CDC *,double,double,CRGB);
    void LineTo(CDC *,CP2);                              //绘制直线,不含终点
    void LineTo(CDC *,double,double);
    void LineTo(CDC *,double,double,CRGB);
public:
    CP2 P0;                                              //起点
    CP2 P1;                                              //终点
}
;CALine::CALine()
{
}
CALine::~ CALine()
{
}
void CALine::MoveTo(CDC *pDC,CP2 p0)
{
    P0=p0;
}
void CALine::MoveTo(CDC *pDC,double x,double y)          //重载函数
{
    MoveTo(pDC,CP2(x,y,CRGB(0.0,0.0,0.0)));
}
void CALine::MoveTo(CDC *pDC,double x,double y,CRGB c)
{
    MoveTo(pDC,CP2(x,y,c));
}
void CALine::LineTo(CDC *pDC,CP2 p1)
{
    P1=p1;
    CP2 p,t;
    CRGB c0,c1;
    if(fabs(P0.x-P1.x)==0)                               //绘制垂线
    {
        if(P0.y>P1.y)                                    //交换顶点,使得起始点低于终点顶点
        {
            t=P0;P0=P1;P1=t;
        }
        for(p=P0;p.y<P1.y;p.y++)
        {
            pDC->SetPixelV(Round(p.x),Round(p.y),
                RGB(p.c.red*255,p.c.green*255,p.c.blue*255));
        }
    }
```

```
    else
    {
        double k,e;
        k=(P1.y-P0.y)/(P1.x-P0.x);
        if(k>1.0)                              //绘制 k>1
        {
            if(P0.y>P1.y)
            {
                t=P0;P0=P1;P1=t;
            }
            for(p=P0,e=1/k;p.y<P1.y;p.y++)
            {
                c0=CRGB(e,e,e);
                c1=CRGB(1.0-e,1.0-e,1.0-e);
                pDC->SetPixelV (Round(p.x),Round(p.y),
                                RGB(c0.red*255,c0.green*255,c0.blue*255));
                pDC->SetPixelV (Round(p.x+1),Round(p.y),
                                RGB(c1.red*255,c1.green*255,c1.blue*255));
                e=e+1/k;
                if(e>=1.0)
                {
                    p.x++;
                    e--;
                }
            }
        }
        if(0.0<=k && k<=1.0)                   //绘制 0≤k≤1
        {
            if(P0.x>P1.x)
            {
                t=P0;P0=P1;P1=t;
            }
            for(p=P0,e=k;p.x<P1.x;p.x++)
            {
                c0=CRGB(e,e,e);
                c1=CRGB(1.0-e,1.0-e,1.0-e);
                pDC->SetPixelV (Round(p.x),Round(p.y),
                                RGB(c0.red*255,c0.green*255,c0.blue*255));
                pDC->SetPixelV (Round(p.x),Round(p.y+1),
                                RGB(c1.red*255,c1.green*255,c1.blue*255));
                e=e+k;
                if(e>=1.0)
                {
                    p.y++;
```

```
                e--;
            }
        }
    }
    if(k>=-1.0 && k<0.0)                    //绘制-1≤k＜0
    {
        if(P0.x>P1.x)
        {
            t=P0;P0=P1;P1=t;
        }
        for(p=P0,e=-k;p.x<P1.x;p.x++)
        {
            c0=CRGB(e,e,e);
            c1=CRGB(1.0-e,1.0-e,1.0-e);
            pDC->SetPixelV (Round(p.x),Round(p.y),
                            RGB(c0.red*255,c0.green*255,c0.blue*255));
            pDC->SetPixelV (Round(p.x),Round(p.y-1),
                            RGB(c1.red*255,c1.green*255,c1.blue*255));
            e=e-k;
            if(e>=1.0)
            {
                p.y--;
                e--;
            }
        }
    }
    if(k<-1.0)                              //绘制 k＜-1
    {
        if(P0.y<P1.y)
        {
            t=P0;P0=P1;P1=t;
        }
        for(p=P0,e=-1/k;p.y>P1.y;p.y--)
        {
            c0=CRGB(e,e,e);
            c1=CRGB(1.0-e,1.0-e,1.0-e);
            pDC->SetPixelV (Round(p.x),Round(p.y),
                            RGB(c0.red*255,c0.green*255,c0.blue*255));
            pDC->SetPixelV (Round(p.x+1),
                            Round(p.y),RGB(c1.red*255,c1.green*255,c1.blue
                            *255));
            e=e-1/k;
            if(e>=1.0)
            {
```

```
                    p.x++;
                    e--;
                }
            }
        }
    }
    P0=p1;
}
void CALine::LineTo(CDC * pDC,double x,double y)            //重载函数
{
    LineTo(pDC,CP2(x,y,CRGB(0.0,0.0,0.0)));
}
void CALine::LineTo(CDC * pDC,double x,double y,CRGB c)
{
    LineTo(pDC,CP2(x,y,c));
}
```

本案例分别为 MoveTo()函数和 LineTo()函数定义了重载函数,可以处理 double 类型参数和 CP2 类型参数。Wu 反走样算法是用两个像素来共同表示理想直线上的一个点,依据两个像素与理想直线的距离而调节其亮度,使所绘制的直线达到视觉上消除锯齿的效果。从绘制效果可以看出,两个像素宽度的直线反走样的效果较好,视觉效果上直线的宽度会有所减小,看起来好像是一个像素宽度的直线。本案例没有考虑背景色对反走样效果的影响,默认背景色为白色,反走样直线段颜色为黑色。

3. 设计鼠标消息映射

本案例要求在屏幕客户区按下鼠标左键后,拖动鼠标到另一位置,释放鼠标左键绘制直线段,所以需要映射 WM_LBUTTONDOWN 消息和 WM_LBUTTONUP 消息。当鼠标左键按下时,设置鼠标光标位置点为直线段的起点坐标,鼠标左键弹起时,设置鼠标光标位置点为直线段的终点坐标。在 WM_LBUTTONUP 消息映射函数 OnLButtonUp()中绘制了反走样直线段。

(1) 添加 CTestView 类的数据成员

向 CTestView 类添加两个 CP2 类型的数据成员 p0 和 p1 分别代表直线段的起点坐标和终点坐标。

(2) 添加 WM_LBUTTONDOWN 消息

```
void CTestView::OnLButtonDown(UINT nFlags, CPoint point)
{
    //TODO: Add your message handler code here and/or call default
    p0.x=point.x;
    p0.y=point.y;
    p0.x=p0.x-rect.Width()/2;                    //设备坐标系向自定义坐标系转换
    p0.y=rect.Height()/2-p0.y;
    CView::OnLButtonDown(nFlags, point);
```

```
}
```

(3) 添加 WM_LBUTTONUP 消息

```
void CTestView::OnLButtonUp(UINT nFlags, CPoint point)
{
    //TODO: Add your message handler code here and/or call default
    p1.x=point.x;
    p1.y=point.y;
    CALine * line=new CALine;
    CDC * pDC=GetDC();                                          //定义设备上下文指针
    pDC->SetMapMode(MM_ANISOTROPIC);                            //自定义坐标系
    pDC->SetWindowExt(rect.Width(),rect.Height());              //设置窗口比例
    pDC->SetViewportExt(rect.Width(),-rect.Height());
                                        //设置视区比例,且 x 轴水平向右,y 轴垂直向上
    pDC->SetViewportOrg(rect.Width()/2,rect.Height()/2);
                                        //设置客户区中心为坐标系原点
    rect.OffsetRect(-rect.Width()/2,-rect.Height()/2);          //矩形与客户区重合
    p1.x=p1.x-rect.Width()/2;
    p1.y=rect.Height()/2-p1.y;
    line->MoveTo(pDC,p0);
    line->LineTo(pDC,p1);
    delete line;
    ReleaseDC(pDC);
    CView::OnLButtonUp(nFlags, point);
}
```

本案例使用 CALine 类对象 line 调用 MoveTo()和 LineTo()成员函数绘制反走样直线段。

五、案例总结

本案例自定义 CALine 类来绘制反走样直线段,提供了 MoveTo()成员函数和 LineTo()成员函数,颜色处理使用了案例 2 提供的 CRGB 类。本案例映射了 WM_LBUTTONDOWN 消息来确定直线段的起点坐标,映射 WM_LBUTTONUP 消息来确定直线段的终点坐标并绘制反走样直线段。本案例绘制的反走样直线段与 Word 中使用绘图工具绘制的直线段效果类似。图 5-3 是使用 Word 的绘图工具绘制的直线段。

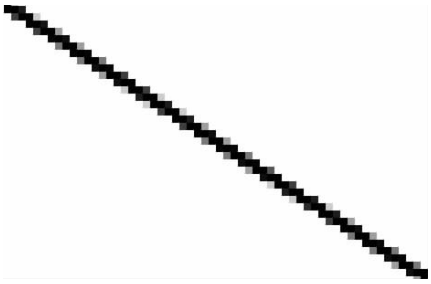

图 5-3　文字处理软件 Word 绘制的反走样直线段