图形学第一次作业

实验报告

1600012938 周尚彦

March 26, 2017

Contents

1 代码与运行环境

- 代码均在 windows10 64 位系统环境下使用 python3.5.3 编译运行。
- 调用第三方包 Pillow 中的 Image 读写图片，ImageDraw 对图片进行绘制。
- 第五章作业使用了 Pillow 包内置的画线算法

2 扫描线填充算法

2.1 算法流程

1. 读入填充区域的顶点
2. 使用 Bresenham 算法将各个顶点依次连接起来作为边界
3. 枚举每个像素，若当前像素在区域内部且未被涂色则将其设定为为种子（使用累计角度法判断一个点是否在区域内部）
4. 使用扫描线算法填充区域

2.2 反走样

将每个边界附近的像素点均分成 9 个小像素点，判断每个点是否在多边形内部，之后加权求和算出像素点应有的灰度值。

2.3 实验结果

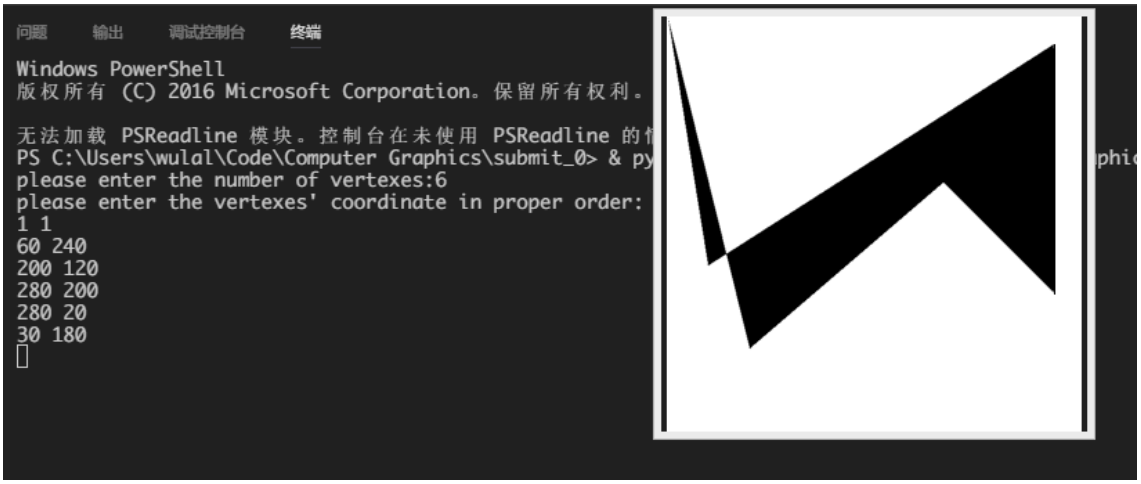

Figure 1: 缩略图效果

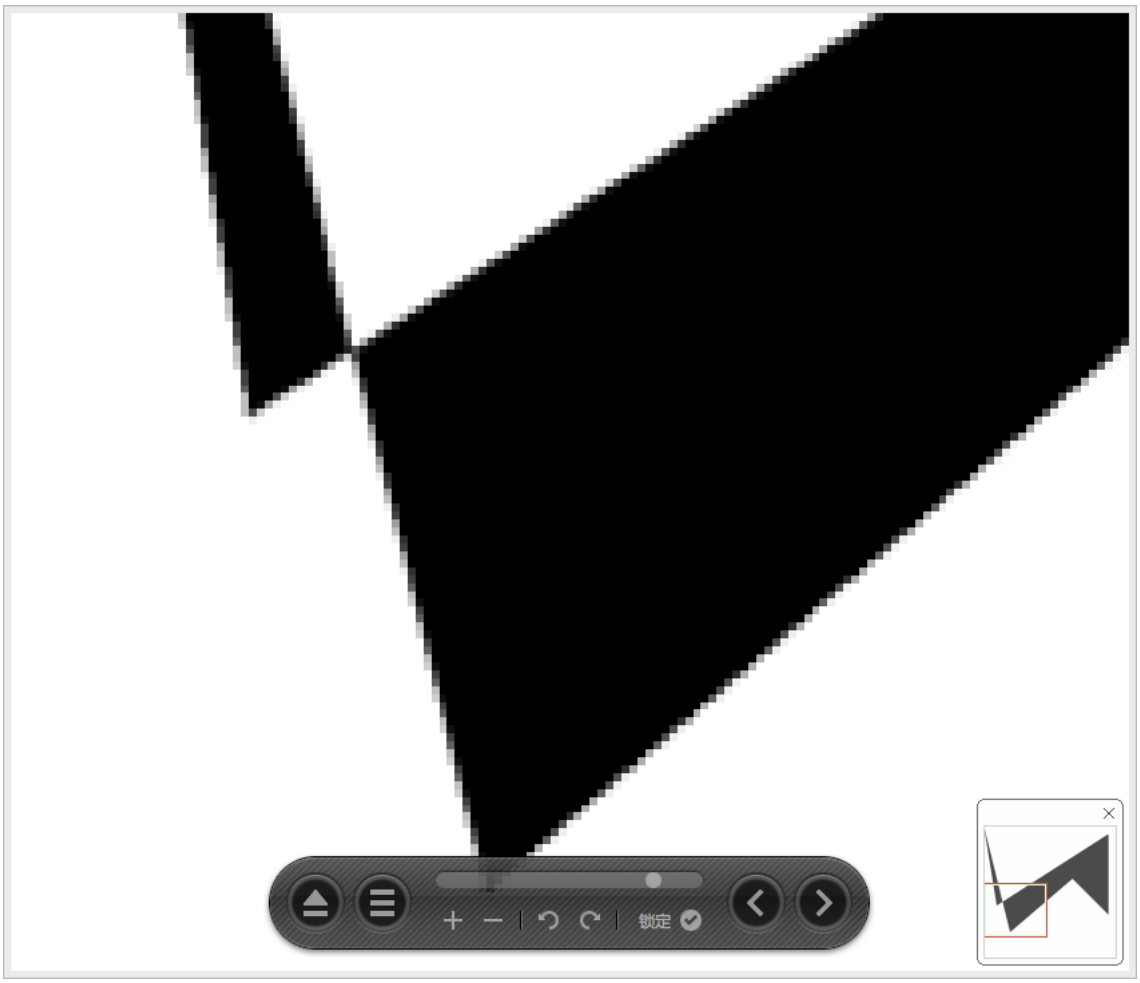

Figure 2: 放大细节

2.4 未解决问题

反走样时间花销很大，是否存在更好的方法?
老师课件中给的例子（字母 P）这样空心区域的边界如何表示?

3 直线反走样算法

3.1 算法流程

1. 读入线段的端点
2. 使用 Bresenham 算法画线，对直线附近的像素点使用加权区域采样的方法求出其灰度值
3. 将每个像素点分为 9 个小像素点并分配以不同的权值，求出每个小像素点到直线的距离，若距离小于 0.5 则认为其在直线内部，加权求和最后得到该像素点的灰度值

3.2 实验结果

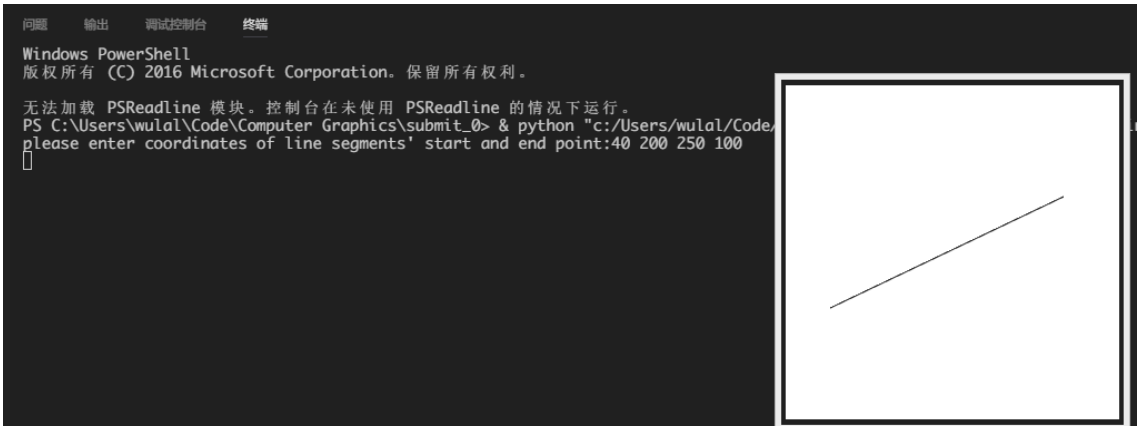

Figure 3: 缩略图效果

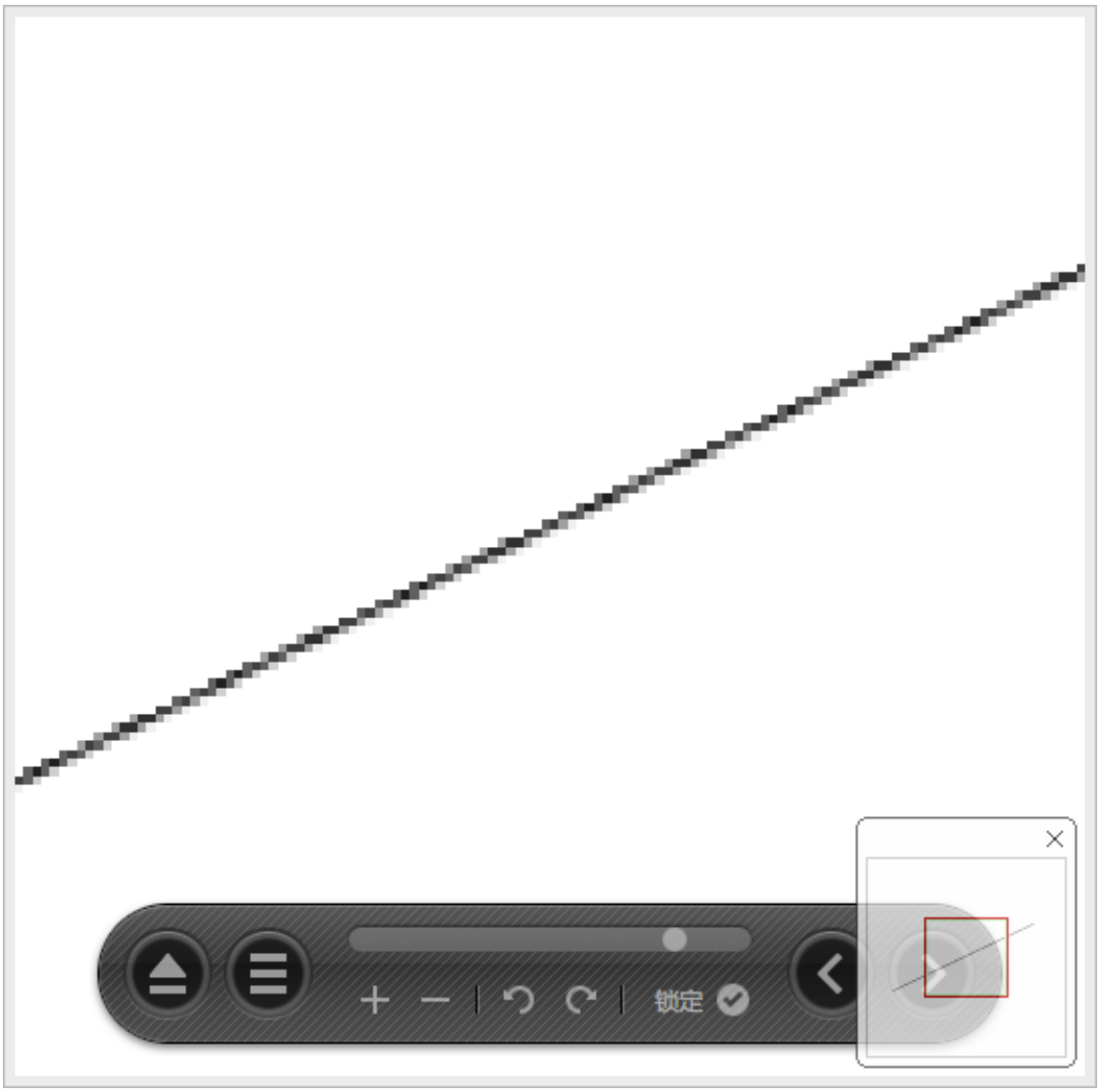

Figure 4: 放大细节

4 Cohen-SutherLand 算法

4.1 算法流程

1. 读入矩形窗口的边界和线段的端点
2. 对线段的端点进行编码，利用编码判断线段的可见性
3. 利用编码求出第一个端点与边界所在直线的交点
4. 修改第一个端点为此交点并重新编码，再次利用编码判断线段的可见性
5. 利用编码求出第二个端点与边界所在直线的交点
6. 使用红色线段标示不可见区域，绿色线段标识可见区域

4.2 实验结果

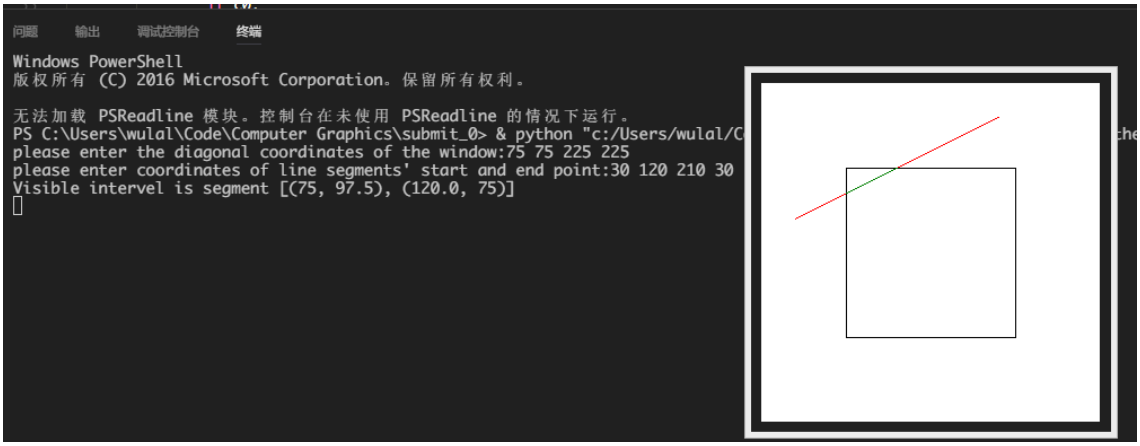

Figure 5: Demo0

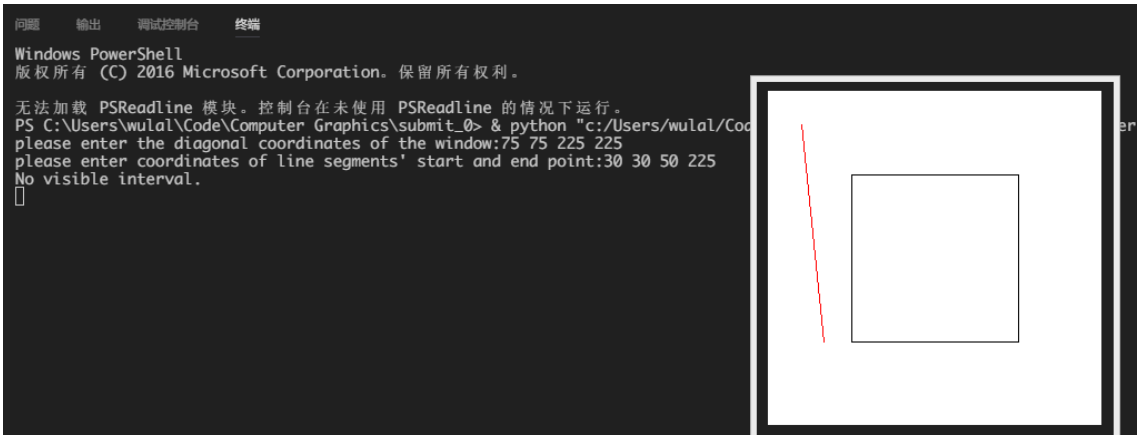

Figure 6: Demo1

5 Cyrus-Beck 算法

5.1 算法流程

1. 读入多边形窗口的顶点和线段的端点
2. 求出每条边的法向量和中点
3. 利用 $n_i \bullet (p(t) - a_i)$ 列出不等式组并求解出 t 的最小最大值 l, r
4. 若 $r \leq l$ 则线段不可见，否则可见区域的交点为 $p(l), p(r)$
5. 使用红色线段标示不可见区域，绿色线段标识可见区域

5.2 实验结果

Figure 7: Demo