**电工导实验报告7**

**一、实验目的**

1. **了解Canny边缘的检测**

**二、实验内容**

1. **学习Canny的算法原理**
2. **Opencv的实现**

**三、实验环境**

1. **Python 2.7 + opencv +numpy**

**四、实验步骤**

图象的边缘指图象局部区域亮度变化显著的部分，该区域的灰度剖面一般可以看作是一个阶跃，即灰度值在很小的区域内急剧的变化。

Canny算法的实现主要有以下几步：

1.灰度化处理

2.高斯滤波使图像更加清晰

3.灰一阶偏导的有限差分来计算梯度的幅值以及方向

4.对梯度幅值进行非最大值抑制

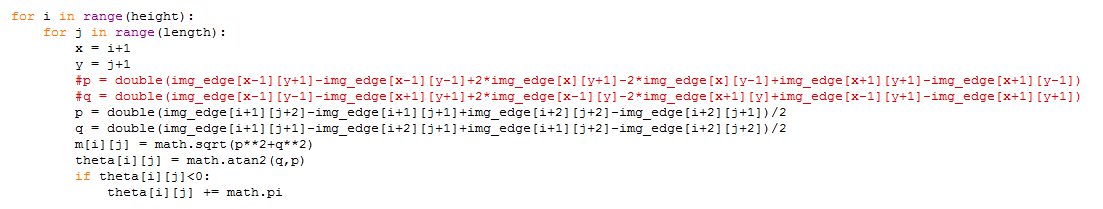
5.双阈值算法检测和连接边缘

**五、问题及其解决**

自己动手进行边缘的检测，并与opencv内置的canny进行对比。

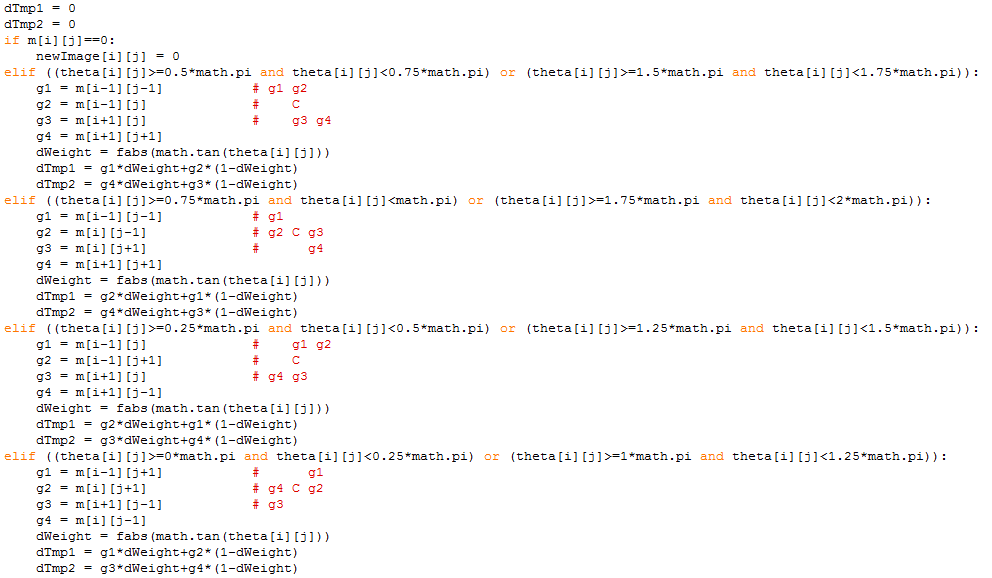
第一步的灰度化直接用灰度读入，第二步的高斯滤波用opencv的cv2.GaussianBlur()函数效率很高，比自己写的快三倍左右，而且不是重点，直接用了内置的。

第三步比较关键，求梯度和方向。这里可以使用不同的算子来得到不同的结果。我采用了canny的算子，之后又采用了sobel的算子，对最终结果会产生不同的影响。



灰一阶偏导的有限差分来计算梯度的幅值以及方向部分代码

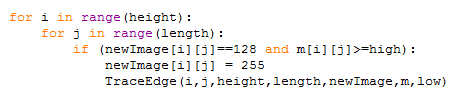
第四步是最重要的一步，对梯度幅值进行非最大值的抑制。如果中间的点梯度小于梯度方向上与周围一圈的交点的幅值，那么这个点一定不是边缘上的点，这样就可以判断是否是边缘上的点。但是由于梯度方向与周围的交点很可能不是周围的九个点，有可能在中间，所以要进行插值计算，计算权重加权得出。由于有四种可能性，所以这里分类讨论。

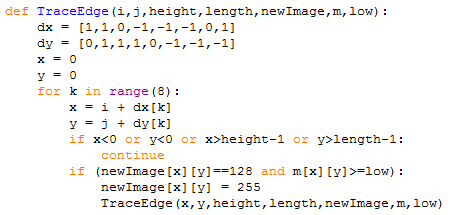


分类讨论的四种情况

紧接着把不可能的点灰度调制0，而如果这个点的幅值大于dTmp1和dTmp2的幅值，那么这个点有可能是边缘点，把它标记一下。

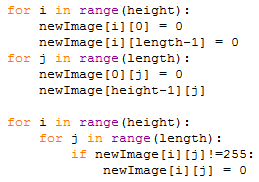
之后是双阈值算法检测和连接边缘，以此来将离散的点练成线。这里有两个阈值，大的那个用来将那些幅值较小的点排除掉，小的阈值用来寻找周围能不能连上别的点。



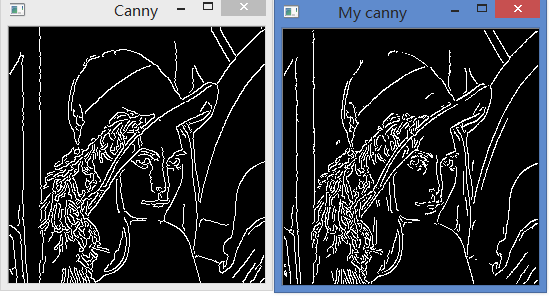


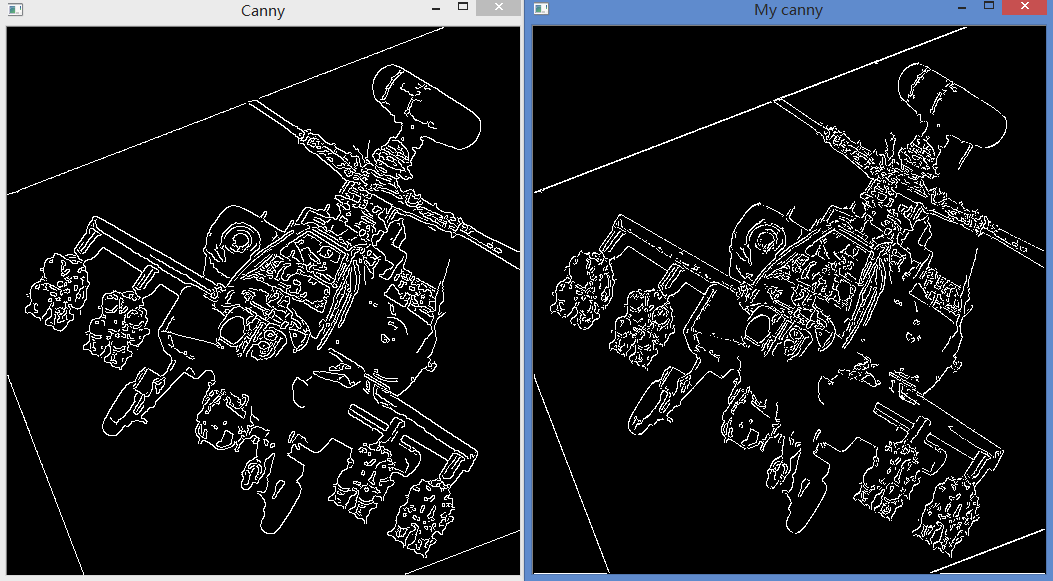
这里通过低的阈值来连接别的点的时候采用递归的算法，递归到周围没有高于低阈值的被标记的点为止。节省了很多代码，逻辑也很清晰。

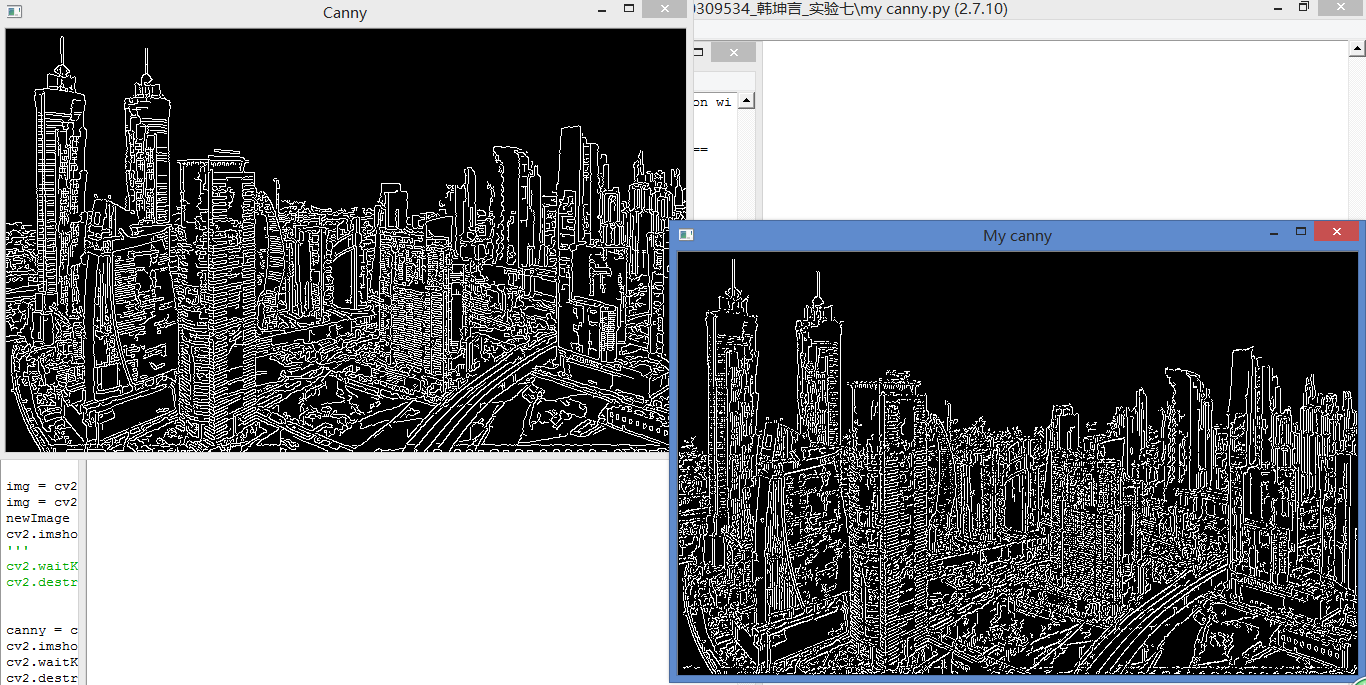
最后还有一些收尾的工作，比如将剩下的点的灰度都调成1，已经处理周围的边框，应为通过算子计算的时候最外面的一圈是无法计算的。代码如下：



与canny的对比如下：







总体而言效果还是不错的。经过尝试，不同的阈值下会得到比较理想的边缘，同样的，改变算子也能一定程度上改善边缘的质量，但是尝试和调节是比较化肥时间的。而且有的时候一幅图的一部分效果好了，但是另一部分的效果就差了，有的时候也很矛盾。

**六、实验总结**

这一次的实验内容非常有趣，边缘检测虽然对我们来说看起来很容易，但对机器来说要识别还是很困难的。通过梯度这一间接的方法真的非常巧妙，数学的知识原来也能在计算机等领域有着这么广泛的应用。这次实验的几步环环相扣，而且让我对图像处理有了更深的了解和掌握。虽然我的结果和canny还是有一些差距，但是看到自己有模有样的成品的时候，心里还是很自豪的。我很期待接下来的课程。

F1403023 5140309534 韩坤言