**EX2 实验报告**

**15331159 李沁航**

实验环境：Ubuntu 16.04 LTS 虚拟机

可执行文件：canny.out

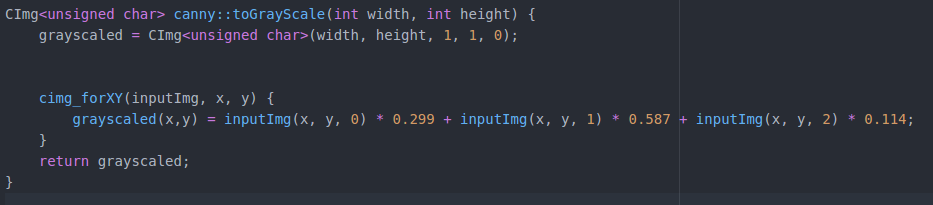
代码文件: canny.h canny.cpp main.cpp

1. 实验部分

学号尾数为9，余数为0，改写的是code0

1. 将图像转换为灰度图

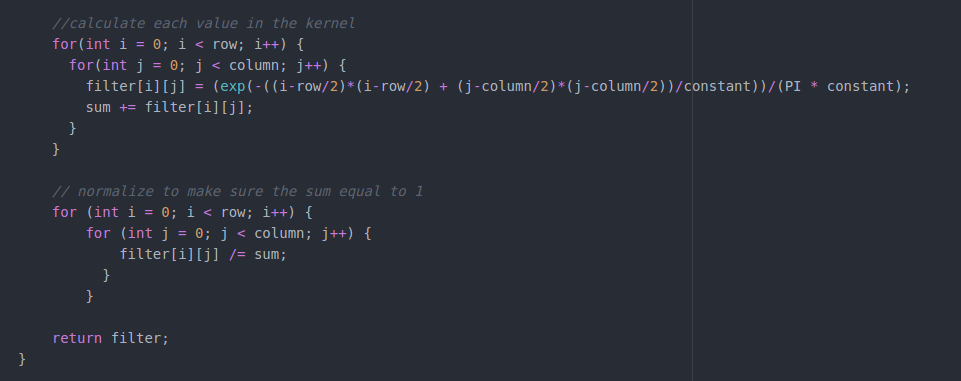
根据公式，转化的公式为R\*0.1226+G\*0.7152+B\*0.0722

转化之后得到的灰度图如下图：

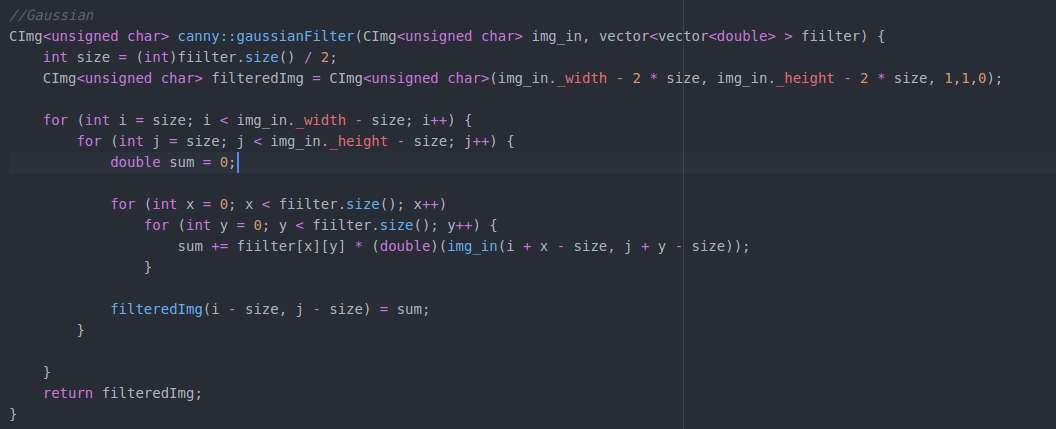


2. 生成Gaussian Kernel，并用其对图片进行滤波处理。

首先根据核的大小计算核中每一个数值的大小，并且使得核中的数值的核为1.0。



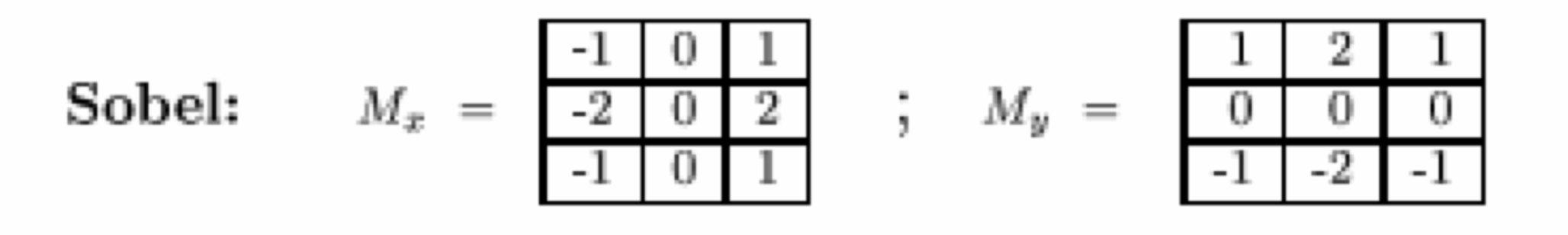
然后使用生产的核对图像进行滤波处理。

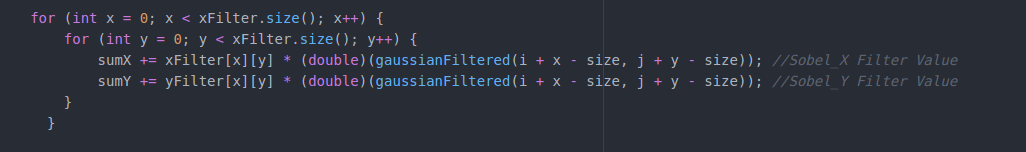
 通过使用刚刚得到的核，计算被一个像素点的滤波之后的数值。

在使用size=5，sigma=5的情况下，滤波之后得到的图像为:



3. 利用sobel算子计算出图像在x，y方向的梯度大小，然后通过x，y两个方向求出的数值，计算得到在该点的方向以及量级。

算子如下：



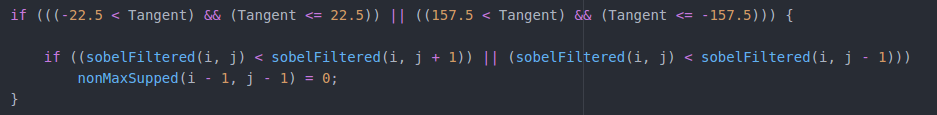
完成之后的结果如图所示：

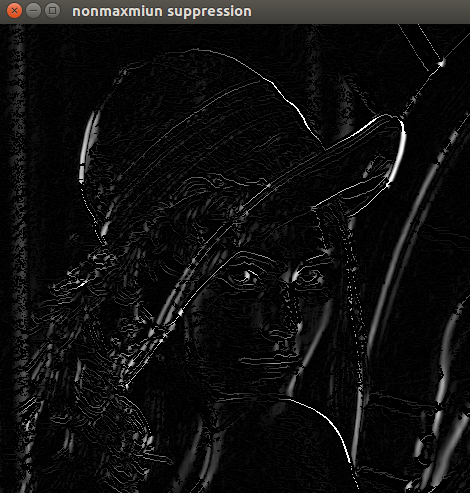




4. 进行NoneMaximum Suppression，减小edge的宽度。

在各个像素点的，沿着其梯度方向，比较前后两个点的大小，若该点的值大于前后两个点的值，则保留，否则将其置0.其中分别为上下、左右、左上右下、右上左下几个方向。

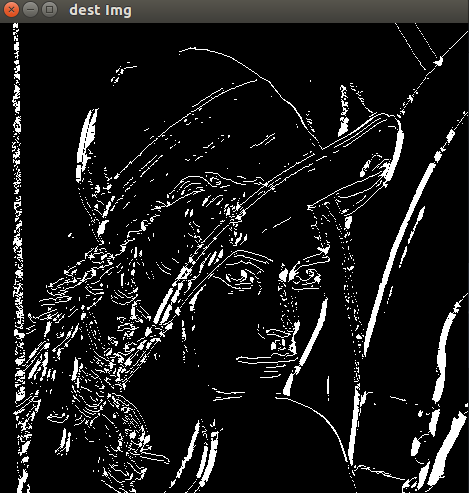
对每一个方向进行类似的比较，完成none Maximum Suppression。



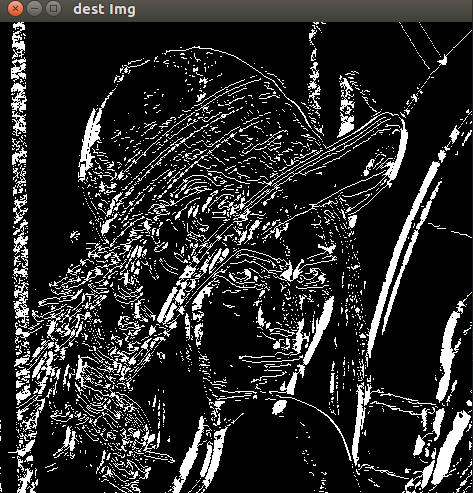
结果如图，可以发现边界确实变细了。

5. 使用阀值进行处理。

使用上下阀值分别为160， 40得到的结果如下图：



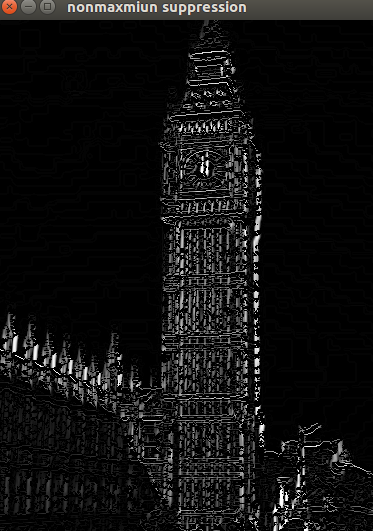
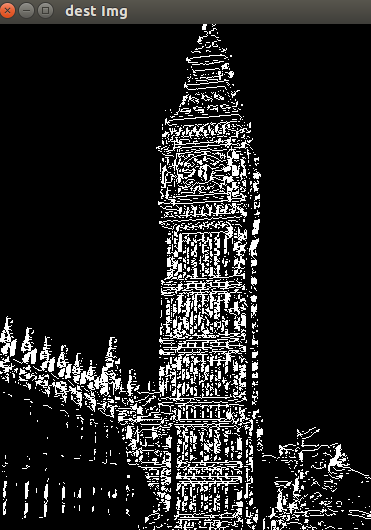
使用上下阀值分别为50， 20得到的结果如下图。



实验结果：







实验之后可以发现，设置的阀值过高会导致遗漏部分的信息，阀值过低可能会导致检测出虚假的边缘。初次之外，在滤波的时候，较大的滤波器会导致更模糊的现象，有利于尖刺较大较平滑的边缘，而较小的滤波器产生的模糊效果比较小，就有利于检测出较小，变化较明显的的边缘。