

一、测试环境

Ubuntu 15.10

在 Ubuntu 15.10 下，Cimg 读取 jpeg 图片的依赖包没有预安装，需要：

```
sudo apt-get install imagemagick libmagickcore-dev
```

二、实验代码编写

我做的是 code0，所以这个实验报告的数据都来自于 code0 跑出来的数据。我也改了 code1 的代码，但由于跑数据再把图片粘到报告上太麻烦了所以就没有对 code1 写实验报告。

由于 code0 源代码是用 C 写的，为了程序的可读性所以改写成了 C++，把所有属性和接口放在 canny 类中，各个接口定义在.h 文件中。

三、测试数据

总共有五个参数：

1. Low threshold
Double threshold 中的低的 threshold 和高的 threshold
2. Gaussian kernel width
高斯核的宽度，单位为像素
3. Gaussian kernel radius
高斯核的半径
4. Contrast normalized
是否对数据进行 normalised 处理

测试方法为：

编译命令：g++ -o canny main.cpp canny.cpp -O2 -lm -lpthread -L/usr/X11R6/lib -lm -lpthread -lX11 -std=c++11

运行命令：./canny filepath _1, _2, _3, _4, _5

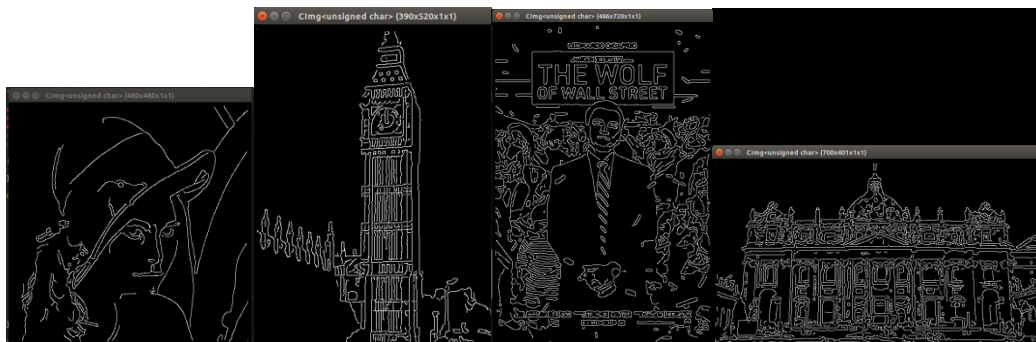
其中，_n 代表以上的五个参数，前四个 float 类型，第五个输入为 0 的话为 false，其余情况为 true。Filepath 代表想要进行处理的文件的路径。

四、测试结果

该算法的默认参数为：

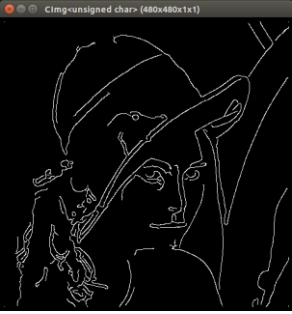
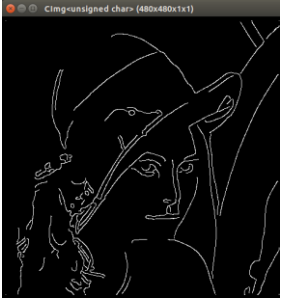
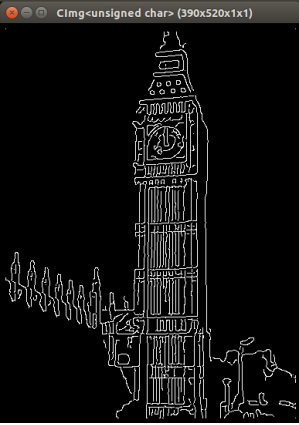
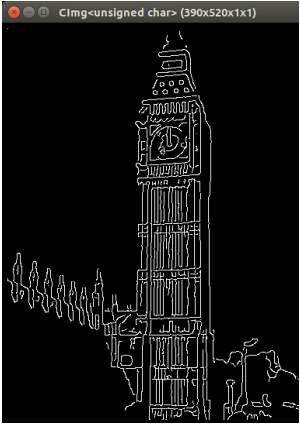
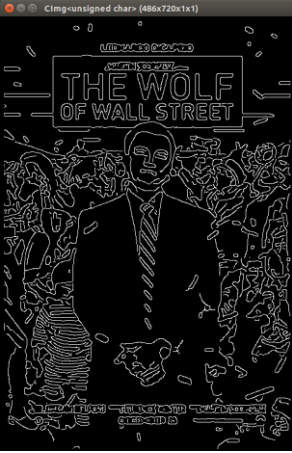
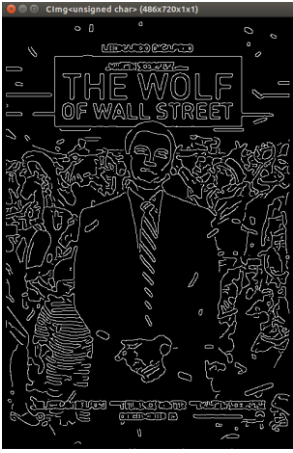
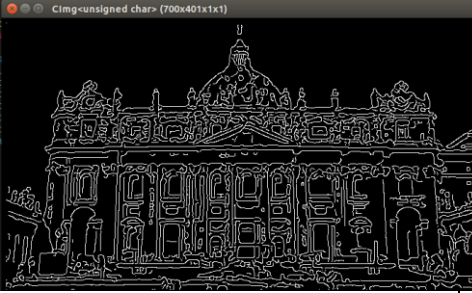
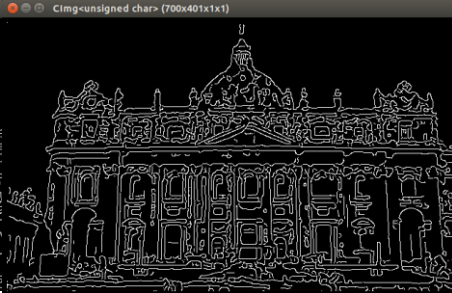
Low threshold = 2.5 High threshold = 7.5
Gaussian kernel width = 16 Gaussian Kernel Radius = 2.0
Contrast normalized = false

默认参数下的处理结果：

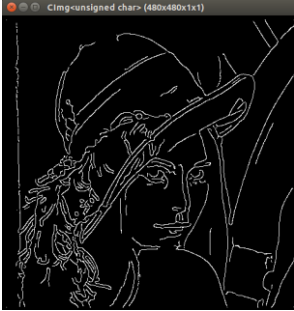


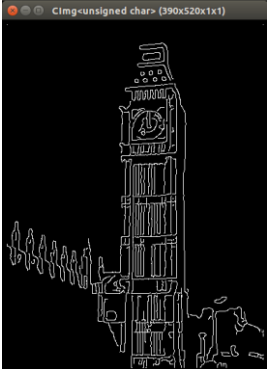
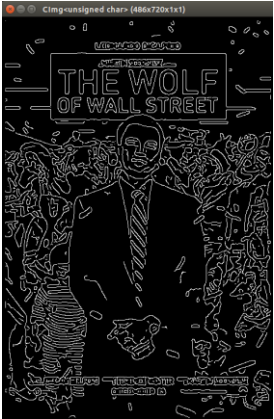
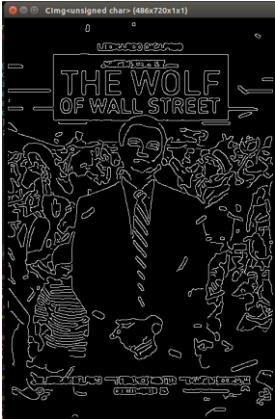
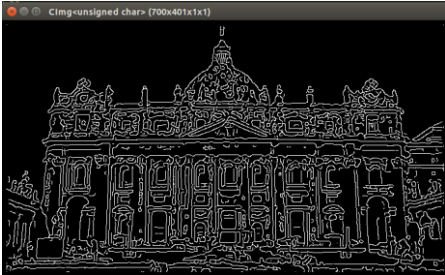
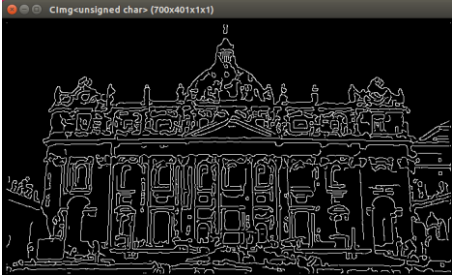


下列测试结果是在只在默认参数情况下改变一个参数得到的结果:

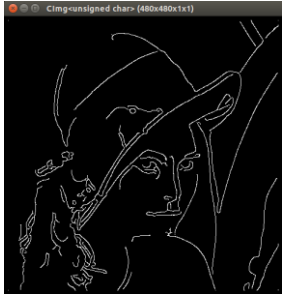
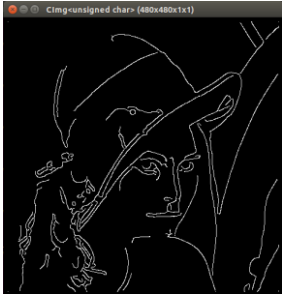
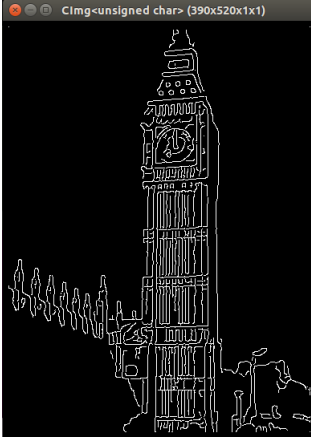
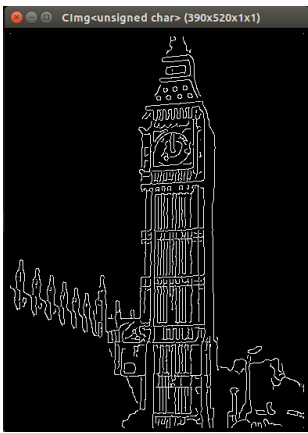
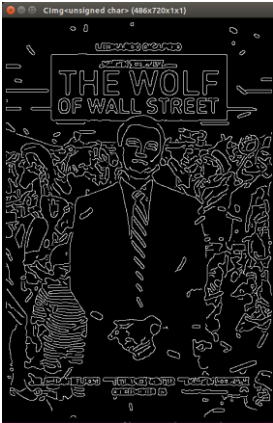
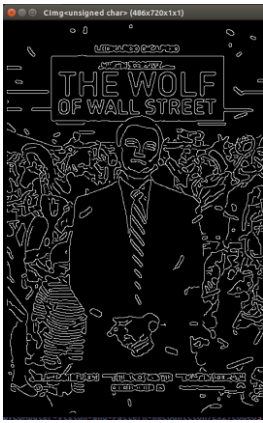
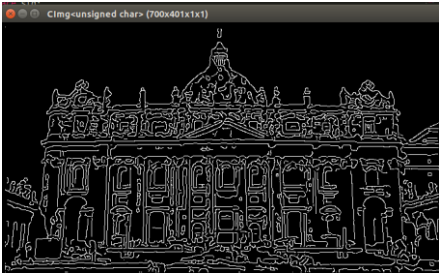
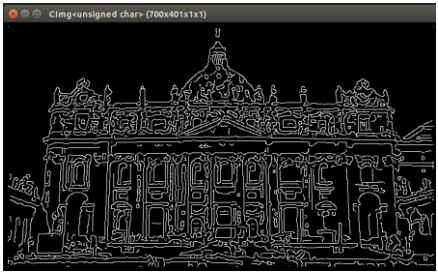
Low threshold:

Image/ Value	0.5	4.5
Lena		
Big ben		
Twows		
Stpietro		

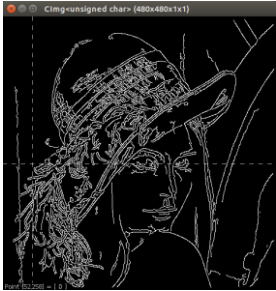

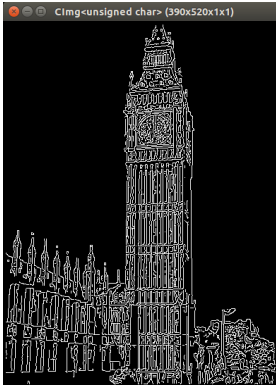
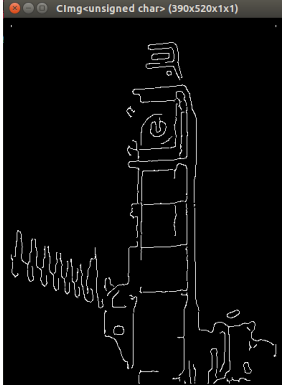
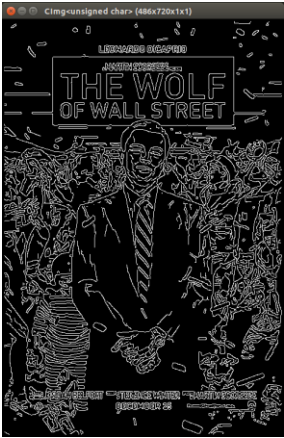
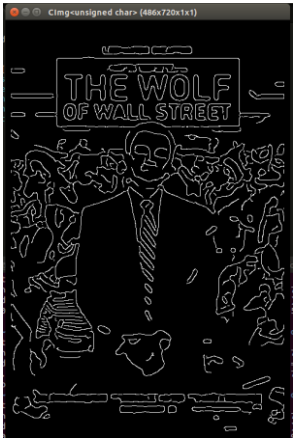
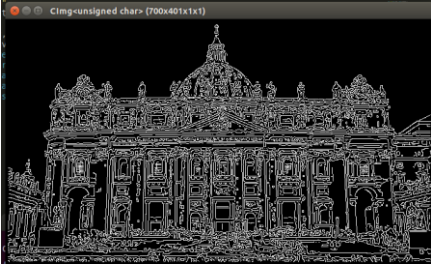

High threshold

Image/ Value	5	10
Lena	 The image shows the edge detection of the Lena test image using a high threshold of 5. The background is black, and the edges are represented by white lines. The features are somewhat sparse and thin.	 The image shows the edge detection of the Lena test image using a high threshold of 10. The edges are more prominent and thicker than in the threshold 5 image.
Big ben	 The image shows the edge detection of the Big Ben test image using a high threshold of 5. The background is black, and the edges are represented by white lines. The structure of the clock tower is clearly visible.	 The image shows the edge detection of the Big Ben test image using a high threshold of 10. The edges are more prominent and thicker than in the threshold 5 image.
Twows	 The image shows the edge detection of the Twows test image using a high threshold of 5. The background is black, and the edges are represented by white lines. The text 'THE WOLF OF WALL STREET' is visible at the top.	 The image shows the edge detection of the Twows test image using a high threshold of 10. The edges are more prominent and thicker than in the threshold 5 image.
Stpietro	 The image shows the edge detection of the Stpietro test image using a high threshold of 5. The background is black, and the edges are represented by white lines. The architectural details of the building are visible.	 The image shows the edge detection of the Stpietro test image using a high threshold of 10. The edges are more prominent and thicker than in the threshold 5 image.

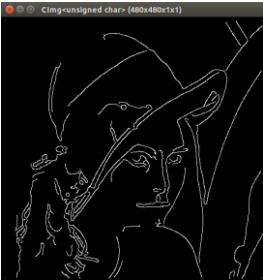
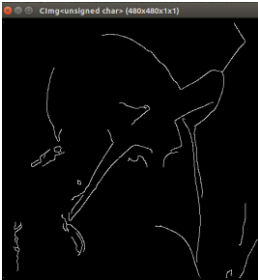

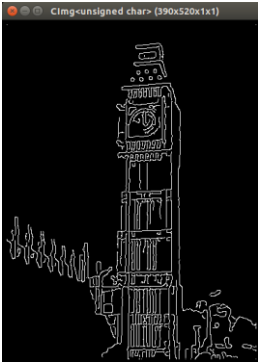

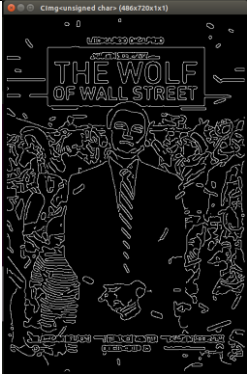
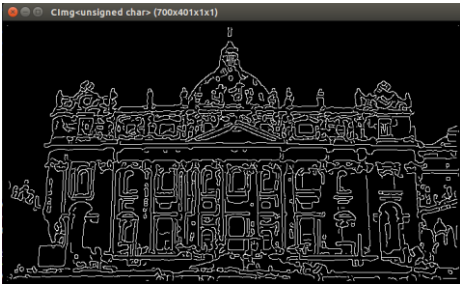
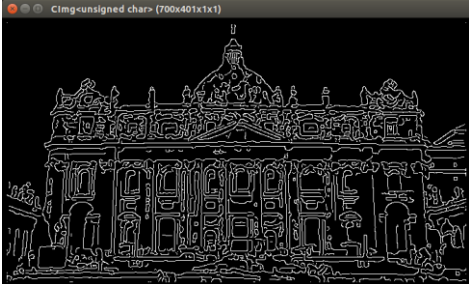
Gaussian kernel width:

Image/ Value	8	24
Lena		
Big ben		
Twos		
Stpietro		

Gaussian kernel radius:

Image/ Value	1	3
Lena		
Big ben		
Twows		
Stpietro		



Contrast normalized:

Image/ Value	False	True
Lena		
Big ben		
Twows		
		

五、结果分析



由于图片大小的原因，所以只拿 lena 的图片进行说明。

Low threshold:

Image/ Value	0.5	4.5
Lena		

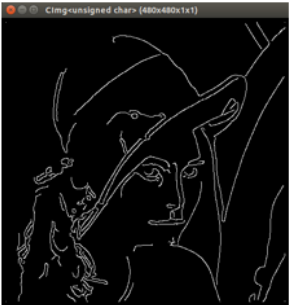

可以看出，当提高了 low threshold 之后，边缘变得残缺了一些。说明提高 low threshold 把一些 weak pixel 给过滤掉了。而 low threshold = 0.5 的时候，边缘有点杂乱，说明很多噪声都没有被筛掉。

High threshold:

Image/ Value	5	10
Lena		

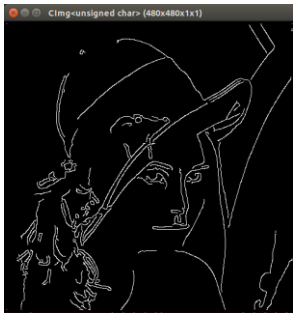
提高 high threshold 之后，edge 残缺了很多。提高 threshold 影响图片中被标为 strong pixel 的数目，因为一个 weak pixel 附近没有 strong 的话会被筛，从而造成在 perform hysteresis 中有更多的 weak pixel 被筛，所以图形会显得残缺。

Gaussian kernel width

Image/ Value	8	24
Lena		

两张图片看上去没有什么区别，这可能是因为 width 在 8 以上对于很多图像都已经够用了，也可能是因为默认的 gaussian kernel radius 为 2，根据 3 sigma 原理，该高斯核卷积在中心点 6 个像素之内收集了 97% 的数据所以窗口为 8 或 24 没什么区别。从这里我们也看到默认参数的 radius 为 2，width 为 16 是比较合理的。其实我认为默认参数的 width 可以更小一点，因为 width 为 6 的时候就能收集到 97% 的数据，width 大了 3 sigma 这么多会让卷积的速度变慢。

我们看 kernel width 更低的时候会发生什么，下图是其他条件不变，gaussian kernel width = 4 的时候的图像

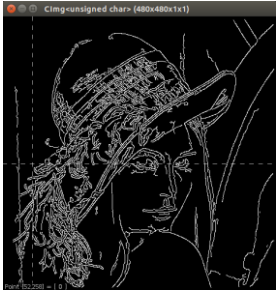
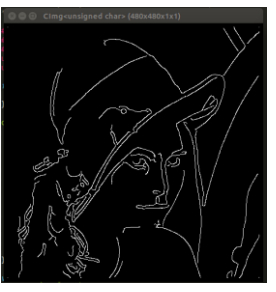



其实大致上也没什么变化，说明这张图片的噪声较小，大部分边缘的强度都是挺高的。但是降为 3 的时候变化就比较明显了：



这是因为卷积的时候收集周围的信息不够多，导致像素的强度偏低，所以很多边缘都被筛掉了。

Gaussian kernel radius

Image/ Value	1	2	3
Lena			

可以看出，该参数对边缘检测的效果影响比较大。**Radius** 设的小一点，细节表现得比较多（线比较多）。设的大，细节就比较少。**Radius** 实际上指的是卷积的时候用的高斯核的标准差，标准差越大，中心点周围区域的权值越大，卷积的时候带来的模糊效应就会越明显，在之后算梯度的时候会让梯度比较小，所以该值越大，处理后的边缘越少。同时可以看到，默认参数 2 的效果确实很好。

Contrast normalized:

Image/ Value	False	True
Lena		

这个参数的带来的变化有点出乎我的意料，在做了 **normalization** 之后很多边缘都没有显示出来，尤其是脸部的细节。所以我就把 **lena.jpg** 经过直方图均衡的图片处理了出来如下：

	处理前	处理后
Lena		

可以看到，处理后比处理前要暗。这是因为处理前比较白，灰度集中在靠近 255 一侧，所以在直方图均衡之后很多原本靠近 255 的映射到了低一点的数值。而效果变差可能是因为脸部边缘的像素在直方图均衡的过程中灰度差值减小从而造成梯度减小，所以没有检测出来。