14301027+EX2文档

1. 测试环境

Windows10（64bits），mingw

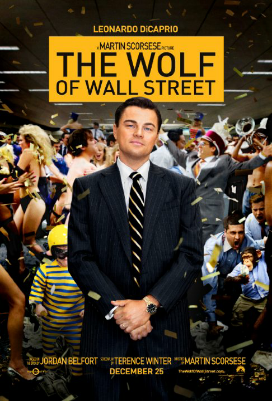
g++ -o main main.cpp can.cpp –O2 –lgdi32

编译直接输入make即可（一个bat文件）

运行为main

输入5个数据：i width radius low high，其中i为图像编号

1. 测试数据：四幅图像





1. 实验代码编写

由学号分配本次作业我做的是code0。相比另外两个code，code0是面向过程的代码，于是参照code1，我把canny这个算法封装成了一个类，同时图像的各种参数，中间结果和属性也在类里。

1. 测试结果与结果分析

Bigben：

阈值处理部分相对来说对结果影响不大，但是高斯核半径（sigma）对bigben有影响，因为这幅图纹路比较细，较大的高斯核容易丢失某些边缘：

|  |  |  |
| --- | --- | --- |
| Sigma | 1 | 2 |
|  | C:\Users\Sevy\AppData\Local\Microsoft\Windows\INetCache\Content.Word\16.1.png | C:\Users\Sevy\AppData\Local\Microsoft\Windows\INetCache\Content.Word\5.2.png |

可以看到sigma=1时左下角的阴影部分的不明显边缘被检测出来了，而sigma=2时这部分并没有被检测出来，而只检测出了大点的边缘。（但同时，sigma=2时钟楼整体的边缘没sigma=1时杂乱）

Lena：这幅图边缘有大有细，在阈值的选择上对结果影响较大，（sigma取2效果最好）

|  |  |  |
| --- | --- | --- |
| Low/high | 2/5 | 3/7 |
|  |  |  |

可以看出，低阈值比高阈值连续性强，但是比较杂乱，同理阈值调高之后，一些噪声点被当做weak point被排除，但代价是边缘连续性变差了。

Stpietro：

对这幅图，主要是高斯核半径对其边缘检测的结果有影响。因为城堡很多纹路，半径大小决定了细小纹路是否被检测出来。（同时半径过小又会导致边缘过于繁杂，此处2是最好的）

|  |  |  |
| --- | --- | --- |
| radius | 2 | 3 |
|  |  |  |

可以看出，半径为2时，稍微明显点的边缘都检测出来了，但是半径为3时，那些小纹路（窗口啊浮雕啊那种）都检测不出来了。（半径为1的图更为繁杂，此处不展示）

Twows：

同上，而且这幅图我觉得主要是要检测中间人物的边缘。

|  |  |  |
| --- | --- | --- |
| Radius | 2 | 3 |
|  |  |  |

可以看出，特别是中间人物左边的条纹衫那里，半径为3时那部分的条纹就被排除了，但同时因为下面那排英文字母比较密集，即纹路比较细，所以半径增大时也被排除了。

综上：参数width即高斯核宽度，代表要卷积的范围大小，一般不要太小，信息是足够的。

参数radius为高斯核半径，小的半径对细纹理检测较好，大半径主要用来检测明显的大边缘。参数low和high控制了边缘的杂乱性和连续性。