计算机视觉第二次作业报告

16340242 吴宇祺

零、作业题目及操作环境

题目

1. 改写Canny边缘检测代码并将函数进行封装。封装要求：(1)所有的图像读写、数据处理只能用CImg 库(整个工程文件不允许使用Opencv 之类的第三方库)； (2)代码封装要求函数接口简洁清晰，可参考Code2 的方式封装。
2. 在原来的代码基础上, 增加一个函数：首先把相邻的边缘连成长的线条，并删除长度小于20 的Edge。
3. 对算法的若干组参数，对所有测试图像进行测试，并分析各参数对结果的影响。

实验环境

编译调试及测试均在windows7命令行使用minGW，编译命令如下。

```
$ g++ test.cpp canny_cimg.cpp -o main -O2 -lgdi32
```

一、代码改写及分析

改写

Code2的代码已经封装成类的形式，主要是将openCV使用的数据结构和库方法替换为CImg的，以及修改图像坐标系。

1. 类库替换：

   Mat矩阵替换为CImg<unsigned char>; openCV库中读取图像函数Mat imread(char*)

```
//openCV
img = imread(filename);     //读取图像
namedWindow("Original");    //显示图像
imshow("Original", img);
img.at<char>(i, j) = 0;     //访问像素值

//CImg
img.load_bmp(filename);     //读取图像
img.display("Original")     //显示图像
img.atXY(j, i)              //访问像素值
img.save(pathname)          //存储图像
```

2. 坐标系改写

   在CImg图像坐标系中，以右上角为原点(0, 0)，水平向右为x非负轴，垂直向下为y非负轴。类方法width()返回图像列数cols，height()返回图像行数rows，且访问像素时cols在前。源码实现的是从左到右从上到下的像素扫描，为了保持一致，遍历的循环结构如下：

```
for (int i = 0; i < img.heigh(); i++)
    for(int j = 0; j < img.width(); j++)
        img.atXY(j, i) = 0;
```

分析

canny类保存了如下几个成员变量

```
CImg<unsigned char> img;            //原始图像
CImg<unsigned char> grayscaled;     //灰度图像
CImg<unsigned char> gFiltered;      //高斯滤波结果图
CImg<unsigned char> sFiltered;      //梯度幅值图像
CImg<unsigned char> angles;         //梯度角度图
CImg<unsigned char> non;            //非极大值抑制结果图
CImg<unsigned char> thres;          //双阈值处理结果图
```

源码在canny构造函数中调用了所有类函数对输入图像进行了处理和结果展示保存。根据Canny边缘检测方法步骤为：

1. 高斯滤波器处理图像（高斯卷积）
2. 计算像素梯度强度和方向（sobel算子）
3. 非极大值抑制（比较当前点的梯度强度和沿正负梯度方向上的两个像素的梯度强度；若该点梯度强度最大，则保留之；）
4. 双阈值进行二值化处理
   - 遍历所有像素
     - 若该像素值大于高阈值，视为强边缘点并保留；
     - 若该像素值小于低阈值，视为非边缘点并删除；
     - 否则视为弱边缘点
       - 若该弱边缘点的八邻域有强边缘点，

或者八邻域内有弱边缘点（没有强边缘点）而24邻域内有强边缘点，

视为强边缘点并保留；

二、边缘连接和短边缘抑制

查阅资料得知用八邻域、24邻域判断弱边缘和强边缘已经进行了边缘连接，故低于20边缘抑制如下。

基于广度优先搜索的短边缘抑制算法

数据结构：

1. 二值图flag，标记某一像素是否已经被访问过，初始化为0，若已经访问置对应位置值为255.
2. vector<pair<int, int > > connected，记录从某一边缘点出发广搜得到的所有边缘点的集合。
3. 输入图shortEdgeSupped

伪代码：

> 输入一个二值图像imgIn；
>
> 对于imgIn进行从左到右、从上到下的像素遍历，设当前位置(i, j)
>
> **BEGIN**
>
> **IF** imgIn(j, i)是未访问过的边缘点：
>
> **BEGIN**
>
> 将imgIn(j, i)加入connected；
>
> **WHILE** 现有连通区域内仍有待探索像素 ：（index不等于connected内元素数量）
>
> **BEGIN**
>
> index自增一；
>
> 取current = connected[index]作为当前探索位置
>
> **IF** current的八邻域位置(x, y)是未被访问过的边缘点
>
> 标记(x,y)已访问；
>
> 将位置(x,y)加入connected视为待探索点；
>
> **END**
>
> **IF** connected内像素数量大于等于20
>
> 保留该连通区域内所有点作为结果点；
>
> **END**
>
> **END**

代码：

```
//copy.cpp CImg<unsigned char> canny::shortEdgeSupp(CImg<unsigned char> imgin, int minLength)
//内层循环：广度优先
vector<pair<int, int> > connected;
```

```
connected.push_back(pair<int, int >(j, i));
int index = -1;

while(index != (connected.size() - 1)) {  //搜索完成条件:当前连通集中仍有未被访问的点
    index++;
    int currentX = connected[index].first, currentY = connected[index].second;//取当前点
    for (int x = currentX - 1; x <= currentX + 1; ++x) {          //搜索当前点的八邻域
        for (int y = currentY - 1; y <= currentY + 1; ++y) {
            if(x <= 0 || y <= 0 || x >= imgin.height()
               || y >= imgin.width() || flag.atXY(x, y) == 255)  //Out of bounds
              continue;
            flag.atXY(x, y) = 255;                                //标志已经访问过
            if (imgin.atXY(x, y) == (unsigned char)255)           //若八邻域中有边缘点
                connected.push_back(pair<int, int>(x, y));
        }
    }
}
//将连通集中点保留到shortEdgeSupped中
```

三、测试文档

- 测试环境 windows7 minGW
- 测试数据
  - 样例：lena.bmp
  - 参数：
    1. 高斯卷积中sigma（方差），缺省值2， 取1,2,3,4进行测试
    2. 低阈值，缺省值40，取20,30,40,50进行测试
    3. 高阈值，缺省值60，取50,60,70,80进行测试
    4. 是否抑制短边缘：取0或1
- 结果及分析
  1. 改变sigma得到图像： sigma越大噪声减少，强边缘（被误认为是噪声）减少。

Thresholded	**Thresholded**
sigma = 1	**sigma = 2**
Thresholded	Thresholded
sigma = 3	**sigma = 4**

  2. 改变低阈值

     可以观察到低阈值为30时人脸左边缘处边缘点明显增加，降低至20时噪声更为明显，主要表现在帽子、眼睛和下巴处。低阈值增大为60，又会丢失很多强边缘像素，如左边远景的柱子边缘、右边镜子边框的边缘都出现了缺失。这与已知结论吻合：对于弱边缘像素（在低阈值和高阈值之间），这些像素可以从真实边缘提取也可以是因噪声或颜色变化引起的；低阈值越高，保留的弱边缘像素就越少，图像信息和噪声也相应变化。

Thresholded	Thresholded
low threshold = 20	**low threshold = 30**
Thresholded	Thresholded
low threshold = 40	**low threshold = 50**

3. 改变高阈值

   高阈值越高，帽子上下边缘、右边柱子、帽穗的边缘点就越少，相应的镜子边框周围、人眼、帽子表面的噪声也变少了。对于强边缘像素（大于高阈值），这些像素视作从真实边缘提取；高阈值越高，保留的强边缘像素就越少（虽然弱边缘像素相应增加，但总体保留的像素少了），图像信息和噪声也相应变化。

Thresholded	Thresholded
high threshold = 50	**high threshold = 60**
Thresholded	Thresholded
high threshold = 70	**high threshold = 80**

4. 是否抑制短边缘

   长度小于20的边缘被删除。

Thresholded	Final
no	**yes**