山东大学 泰山学堂 学院

数字图像处理 课程实验报告

学号：201500150146	姓名：晁大任	班级： 泰山学堂 2015 级计算机
实验题目：OpenCV 配置及图像基本操作		
实验学时：5	实验日期：2017/10/17	

实验内容：

实验 4.1：高斯滤波

实现图像的高斯滤波：

- 通过调整高斯函数的标准差(sigma)来控制平滑程度；
- 滤波窗口大小取为[6*sigma-1]/2*2+1，[.]表示取整；
- 利用二维高斯函数的行列可分离性进行加速；先对每行进行一维高斯滤波，再对结果的每列进行同样的一维高斯滤波；

一开始计算了一维高斯模板，用一个一维向量表示，每一个值的计算方法：

```
t1 = (pow(x - xcore, 2) + pow(y - ycore, 2)) / 2.0 / sigma / sigma;
temp = base*exp(-t1);
```

根据行列可分离性原理，先对行进行高斯滤波，然后对列进行高斯滤波，对每一个像素点的滤波值计算为：

```
for (int i = -start + n; i <= start + n; i++)
    sum += input.data[m*step + i] * gaussTem[i - n + start];
output.data[m*step + n] = uchar(sum);
```

结果如附件 2 所示，sigma 选择为 0.5

实验 4.2 快速均值滤波

实现图像的均值滤波

- 滤波窗口大小通过参数来指定：

  void MeanFilter(const MyImage &input, MyImage &output, int window_size);

- 采用积分图进行加速，实现与滤波窗口大小无关的效率；

先计算积分图，即进行一次循环扫描所有的像素点，将每一个点的积分值放在一个二维数组中，对每一个值的计算为：

```
sum += input.at<uchar>(x, y);
int temp = sum + res[x-1][y];
v.push_back(temp);
```

最后计算每一个像素点的新值，公式为：

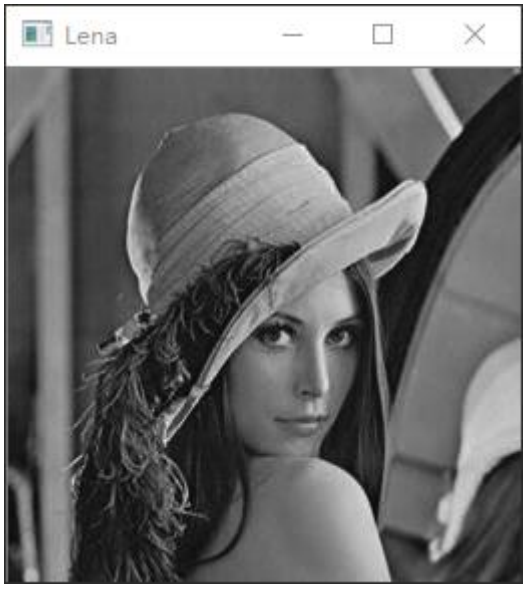
```
int sum = integralGraph[m+start][n+start] - integralGraph[m + start][n - start] -integralGraph[m - start][n + start] + integralGraph[m - start][n - start];
output.at<uchar>(m, n) = uchar(sum / window_size / window_size);
```

由此可以得到均值滤波图像，如附件 3 所示，windows_size 选择为 5

硬件环境：

Intel(R) Core(TM) i5-5200U 8GB 64 位操作系统
软件环境： Windows10 C++语言环境 VS2015 Project 类型：WIN32 控制台应用程序
实验过程中遇到和解决的问题： 1 由于数组比较难进行管理，于是使用了 vector 类代替数组 2 在边界处理时分别考虑了原值拷贝和镜像化处理，原值拷贝会使得边界无法自然过渡，而镜像化处理需要考虑多种情况。
结论分析与体会： 通过对高斯滤波和均值滤波算法的实现，让我对相关知识有了初步的了解，使我对图像处理的算法充满了好奇与热情，同时锻炼了实践能力。 我一开始实现了普通的高斯滤波和普通均值滤波，然后在此基础上实现了应用行列可分离性原理的高斯滤波以及采用积分图进行加速的均值滤波。 我相信，在老师的辛勤教导以及我的努力学习下，我一定可以出色的完成这门课并掌握好课程教授内容。

附件 1：原图



附件 2：高斯滤波 sigma=0.5

附件 3：均值滤波 windows_size=5