

# 浙江大学实验报告

专业：计算机科学与技术

姓名：张三

学号：1111111

日期：2021/10/20

课程名称：\_\_\_\_图像信息处理\_\_\_\_ 指导老师：\_\_\_\_宋明黎\_\_\_\_ 成绩：\_\_\_\_

实验名称：\_\_\_\_均值滤波以及拉普拉斯图像增强\_\_\_\_

## 一、实验目的和要求

实验目的：了解图像滤波的意义并熟悉一些常见的滤波操作；

了解拉普拉斯算子的原理和计算方法，掌握图像锐化的步骤

实验要求：使用 C 语言对 bmp 图像实现均值滤波和拉普拉斯算子增强。

## 二、实验内容和原理

### 1.均值滤波

使用一定大小掩膜（矩阵）对图像像素进行覆盖，用所有像素点的（加权）平均作为中心像素点的新像素值，如此即使有噪声，如果噪声较小，可以减少噪声影响；最简单的情况即对掩膜内像素值求平均，也可以根据图像噪声的特点对各处加权，使滤波效果更好。

### 2.拉普拉斯算子

拉普拉斯算子是一种各向同性算子，二阶微分算子，在只关心边缘的位置而不考虑其周围的像素灰度差值时比较合适。拉普拉斯存在两种掩膜：

0	-1	0
-1	4	-1
0	-1	0

-1	-1	-1
-1	8	-1
-1	-1	-1

容易看出，原图与拉普拉斯掩膜作乘法时是会出现负数的，在拉普拉斯结果和原图融合保持锐化时要考虑符号的不同来判断使用加或减。总体原则是灰度值增大，即拉普拉斯结果为父则减，反之则加。

### 三、实验步骤与分析

#### 1.均值滤波

均值滤波操作如其名，就是将像素点周围的像素值取平均作为新像素值，对于图像边缘的像素，由于影响不大，在此处省略不进行操作。可通过  $i,j$  对原图进行遍历，另设参数  $m,n$  对周围像素进行遍历求和，注意和的类型应当是整型，这样才有足够的位宽存储像素和。代码如下：

```
1. for(i=1;i<height-1;i++){
2.     for(j=1;j<width-1;j++){
3.         int sum=0; //像素值计算的和;
4.         for(m=i-1;m<=i+1;m++){ //计算九宫格的像素和
5.             for(n=j-1;n<=j+1;n++){
6.                 sum+=bmpImg->imageData[m*width_+n];
7.             }
8.         }
9.         bmp_m->imageData[i*width_+j]=sum/9; //求均值
10.    }
11. }
```

#### 2.拉普拉斯图像增强

拉普拉斯算子在数学上和均值滤波操作相同，原图像逐一与一个矩阵相乘便可得到新图像的像素值，只是拉普拉斯算子的掩膜有固定大小与权重，在结果出来后，真正的锐化还需要再与原图结合才可以显示出来，注意同号相加，异号相减。代码如下：

```
1. for(i=1;i<height-1;i++){
2.     for(j=1;j<width-1;j++){
3.         if(M==16){ //扩展掩膜
4.             int sum=0;
5.             for(m=i-1;m<=i+1;m++){
6.                 for(n=j-1;n<=j+1;n++){
7.                     sum+=bmpImg->imageData[m*width_+n]*1;
8.                 }
9.             }
10.            sum-=bmpImg->imageData[i*width_+j]*8;
11.            if(sum<0)sum=0;
12.            else if(sum>255)sum=255;
13.            bmp_l->imageData[i*width_+j]=sum;
14.        }
15.        if(M==8){ //标准掩膜
```

```

16.         int sum;
17.         sum=bmpImg->imageData[(i-1)*width_+j]+
18.         bmpImg->imageData[(i)*width_+j-1]+
19.         bmpImg->imageData[(i)*width_+j+1]+
20.         bmpImg->imageData[(i+1)*width_+j]-
21.         bmpImg->imageData[i*width_+j]*3;
22.         if(sum<0)sum=0;
23.         else if(sum>255)sum=255;
24.
25.         bmp_l->imageData[i*width_+j]=sum;
26.     }
27. }
28. }

```

#### 四、实验环境及运行方法


本人使用 dev-c++ 5.11 版 C 语言工程运行代码，如果电脑上安装有 dev-c++，单击文件“bmpwork.dev”即可打开工程文件，点击编译运行的图标或按 f11 即可。



使用 vscode 运行的方法：打开程序所处文件夹（包含 bmp.h 、bmp.c 以及 main.c 文件），在终端输入“gcc bmp.c main.c -o main”，敲击 enter 键后即可编译生成 main.exe，再直接在终端输入“.\main”或在文件夹中打开 main.exe 即可运行程序得到结果。

**注意：**在运行程序前，保证待操作的 bmp 文件在对应的路径（sample 文件夹）中。

#### 五、实验结果展示

##### 1.均值滤波图像

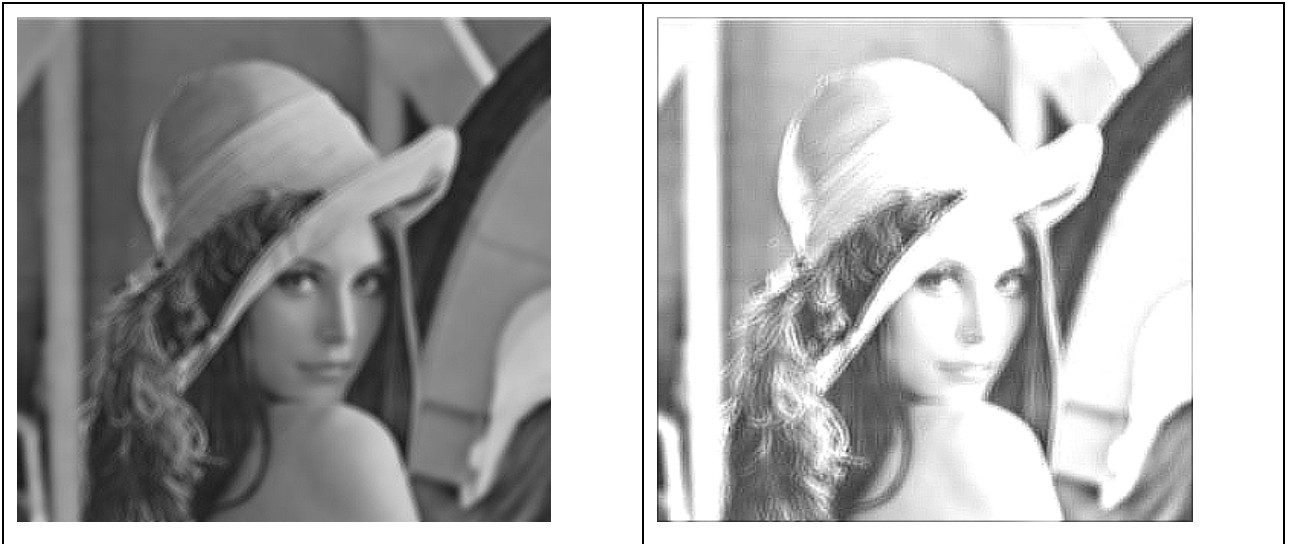
原图	
	

一次滤波	二次滤波
	

可以看出，经过滤波操作后，原图与噪声都变得模糊化了，图片的观感并不如原图清晰，但是噪声也不像在原图上的一样突出，经过二次滤波后效果更加显著，也更加模糊。原图的噪声肉眼可见与图片格格不入，经过几次处理后虽然模糊化了，但是噪声逐渐融入其余像素中，可以预见，再有几处理就可将噪声基本融入环境。

## 2.拉普拉斯图像增强

原图	
	
标准掩膜	扩大化掩膜



左侧为较小的标准掩膜处理结果，右侧为扩展掩膜处理结果。左侧图片相比原图变化较小，锐化效果不明显，反而比原图模糊；而右侧图片效果明显，亮度增大，部分细节突出，但是也有模糊的迹象。

## 六、心得体会

总有各种情况导致图片上出现不和谐的噪声点，滤波等操作就可以尽可能消除噪声，保留原图美感，而锐化在平时也并不罕见，突出图像细节信息（作业拍照上传 orz）是我们经常需要的功能。此次作业接触到了部分操作并亲自实现了它们，但是效果并不理想，或者说与各类现成的软件处理结果相比差距较大，这也是由于自身能力的局限，难以使用更好的算法，还需要加以学习。