

**数字图像处理实验报告**

实验 三

题 目 图像增强

学 院 计算学部

专 业 计算机科学与技术

学 号 22B903037

学 生 刘建

任 课 教 师 姚鸿勋

哈尔滨工业大学计算机科学与技术学院

2022年秋季

实验三 报告

1. 实验内容（contents）
2. 熟悉图像直方图操作和空域滤波的思想，熟悉二值图像处理的思想
3. 编程实现图像的空域滤波

• 编程实现直方图均衡化和归一化操作

• 编程实现给图像加高斯噪声和椒盐噪声

• 编程实现中值滤波和均值滤波，思考其快速算法实现原理，对上述的含噪声图像进行处理

• 尝试编程实现基于小波变换的图像去噪算法实现

• 实验报告中体现不同高斯噪声和椒盐噪声的去噪效果比较，并采用PSNR和SSIM来进行评价

1. 编程实现图像的某种模板算子操作，例如sobel边缘检测等算子，理解模板操作
2. 编程实现基本的二值图像处理操作：阈值化处理和半色调处
3. 实验目的（purposes）

完成实验内容，学习掌握相关知识

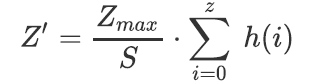
1. 实验设计、算法和流程(Design, algorithm and procedure)
2. 直方图均衡化和归一化操作

直方图均衡化是使图像直方图变得平坦的操作。直方图均衡化能够有效地解决图像整体过暗、过亮的问题，增加图像的清晰度。

def hist\_equal(img,z\_max=255)：

构造函数hist\_equal，传入单通道图和灰度最大值，返回均衡后的图像。

其中S是总的像素数，Zmax是像素的最大取值(8位灰度图像为255)，h(i)为图像像素取值为 i 及 小于 i 的像素的总数。



1. 给图像加高斯噪声和椒盐噪声

我们构造函数GaussiNoist(image ,mean, sigma)对图像image加上高斯噪声，其中mean和sigma分别是高斯噪声的均值和方差。在该函数中，我们利用函数np.random.normal生成均值为mean、方差为sigma、大小为image的形状的高斯噪声，并将这个噪声加到image上，并将图像的像素剪裁到[0,255]范围内。

我们构造函数SaltPapperNoist(image, prob)，以概率prob对图像image加上椒盐噪声。在该函数中，我们用image中的像素点个数乘上prob计算加入的椒盐噪声像素点个数noisenumber，接着随机的选择image中的noisenumber个像素点，随机的赋值为0或255。

1. 中值滤波快速算法

我们构造函数middleFilter(image , size=3)，实现对图像image进行窗口大小为size的中值滤波快速算法。在这个函数中，我们先对image进行pad为size的整数倍，接着对image中每个像素计算它的size\*size领域中的中值：对于第一个像素点计算领域窗口中的像素直方图Hs，接着对这个领域像素值进行排序，选择中间顺序的像素值middle代替这个位置原先的像素值，并计算领域中小于middle的像素个数nms；接着以位移为1移动遍历整个image，每次位移都更新Hs：对位移后离开领域的像素点，对其像素值L，在直方图Hs中减去1，如果L小于middle，nms-1；对位移后加入领域的像素点，对其像素值L，在直方图Hs中加上1，如果L小于middle，nms+1； 设置m\_index=size\*size/2表示领域中间像素值的顺序索引，接着比较m\_index和nms，如果nms<m\_index,则试探nms+Hs[middle]是否小于m\_index，如果小于则令nms = nms+Hs[middle], middle=middle+1，继续比较，直到nms+Hs[middle]大于等于m\_index，则当前领域的中值为middle；如果nms>m\_index,则令middle=middle-1，nms=nms-Hs[middle]，直到nms<=m\_index，则当前领域中值为middle；如果nms=m\_index，则middle为当前领域中值，用middle代替当前的像素值。

1. 均值滤波快速算法实现

我们构造函数AverageFilter(image, size=3)，实现对image进行领域大小为size的均值滤波快速算法。同样我们先将image图像pad到size的整数倍，接着便利图像中的每个像素点。对于第一个像素点计算size\*size领域内像素值的和sum，接着计算sum/(size\*size)来代替当前的像素值，接着位移为1遍历图像，对于位移后离开领域的像素值从sum中减去，对位移后加入领域的像素值加到sum上，然后用sum/(size\*size)来代替当前的像素值。这样既完成均值滤波

1. 基于小波变换的图像去噪算法

构造函数DWT（image），对图像image进行基于小波变换的图像去噪。在该函数中，先对象进行小波变换，返回LL，（LH，HL，HH），四组数据，接着对（LH，HL，HH）部分的数据，按照一个阈值thresh进行比较，如果小于thresh则设置为0，最后对修改后的数据LL，（LH，HL，HH）进行小波逆变换并返回。

1. 实现sobel边缘检测算则。

构造sobel(image)函数对image图像用sobel算子进行边缘检测。函数中，对image中的每个像素点，对像素点的3\*3领域中，用模板和乘以这个领域的像素值，分别计算竖直方向和水平方向的梯度值，将2个梯度值的均值平方和代替该像素点。

1. 阈值化处理 和 半色调处理

构造函数threshold(image,thresh)，对图像image利用阈值thresh进行阈值化处理。该函数中，对图像中的每个像素点，和thresh进行比较，如果小于thresh则设置为0，如果大于thresh则设置为1.

构造函数randomHalfColor(image)，对图像image进行半色调处理。该函数中，将image图像先转换为灰度图像，接着对图像中的每个像素点生成一个随机值，如果该像素点的像素值大于这个随机值则设置为1，否则设置为0。

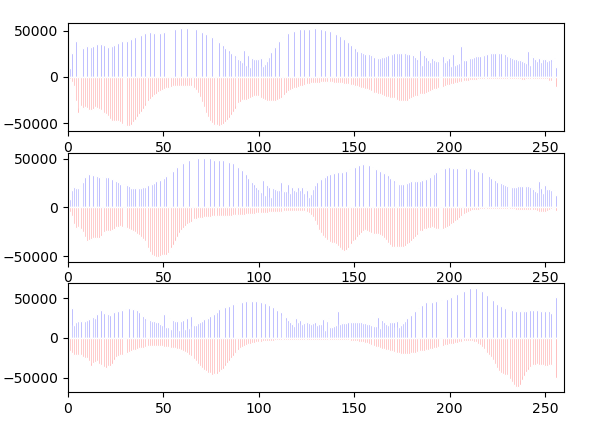
1. 实验结果(results)

我们利用如下的彩色图像进行实验：



4.1 直方图均衡化和归一化

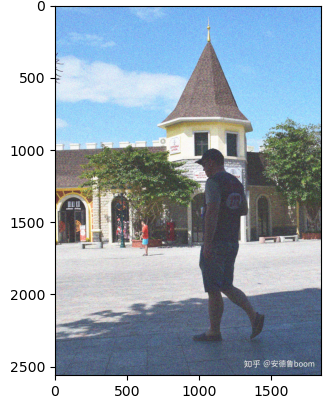
下图是3个channel的直方图，上面蓝色时直方图均衡化后的结果，下图是原图。



下图均衡化后的图像。



3.1 给图像加上高斯噪声后结果：



给图像加上椒盐噪声：



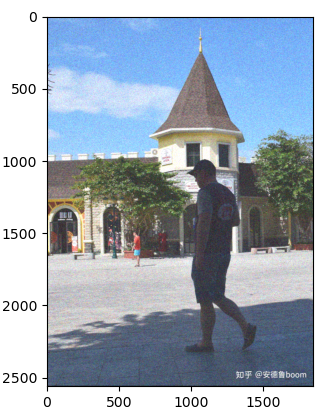
3.1

加了高斯噪声后的PSNR值：13.118434697889844

和原图的SSIM值：0.8980606784119786

加了椒盐噪声后的PSNR值：12.93754837162445

和原图的SSIM值：0.9896543376572053  
中值滤波 对 高斯噪声图片

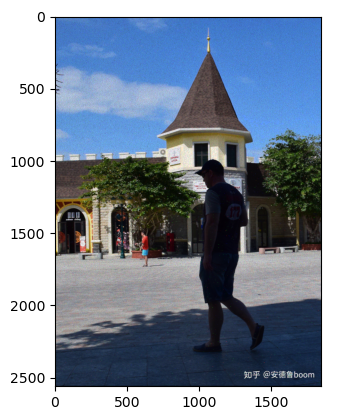


PSNR值为12.867705999936534

和原图的SSIM值：0.9176535986361243

可以看到和原图的相似度增加了

中值滤波 对 椒盐噪声图片



PSNR值：13.163057307025937

和原图的SSIM值值0.9935990321427134

可以看到用中值滤波对椒盐噪声图片的处理比均值滤波要好了很多，图片和还原度很高。

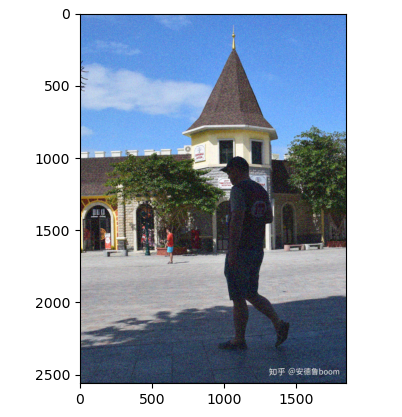
均值滤波 对 高斯噪声图片



PSNR值：14.056084901203933

和原图的SSIM值0.9002891767931177

均值滤波 对 椒盐噪声图片



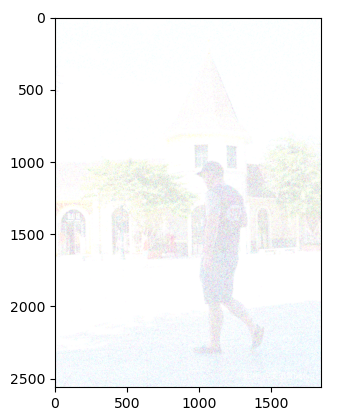
PSNR：14.008914558420935

和原图的SSIM:0.9901423481845606

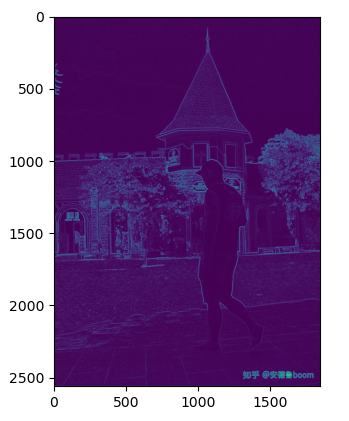
由于是滤波方法，从PSNR上来看，去躁滤波的结果很多反而更差，但是从视觉上来看，均值滤波对加了高斯噪声的图像去躁效果最好，中值滤波对加了椒盐噪声的图像去噪效果最好。

3.1 基于小波变换的图像去噪算法：（DWT）

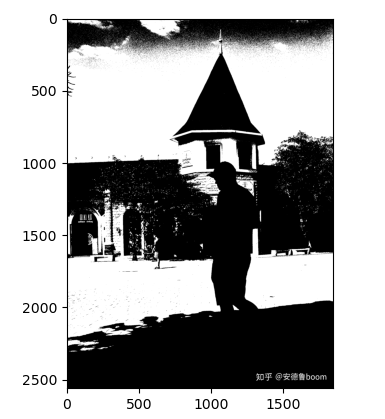
对加了高斯噪声的图像进行基于小波变换的图像去噪，结果如下：



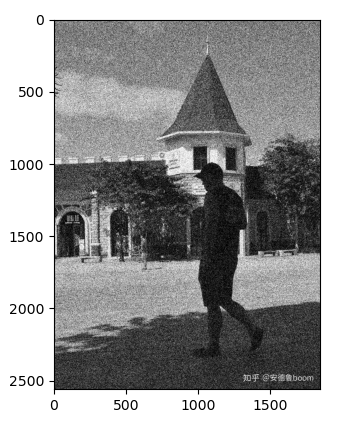
3.2 利用sobel算子进行边缘检测结果为：



3.3 阈值化处理结果：



半色调处理结果：



***Digital Image Processing***

**Mark sheet of experimentation**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| No | Student No. | Name | Score of Part 1  (code) | Score of Part 2  (content) | Score of Part 3  (result) | Score of Part 4  （report） | Total Score |
| 1 |  |  |  |  |  |  |  |

Signature: Date: