**2019**年秋季 图像处理与分析 第八讲作业

学院：人工智能学院 学号：201928014629008 姓名：牛李金梁

编程语言：Python3.7

必要依赖：numpy libopencv matplotlib

运行要求：3个题目写在了同一个py文件里，每个题目弹出一个展示框。运行时在命令行中输入python test8.py，每弹出一个窗口将其关闭即开始运行下一个题目。第二题运行的比较慢。

问题**1** 编写一个图像直方图均衡化程序

程序中编写了calHist()和histequal4e()2个函数。

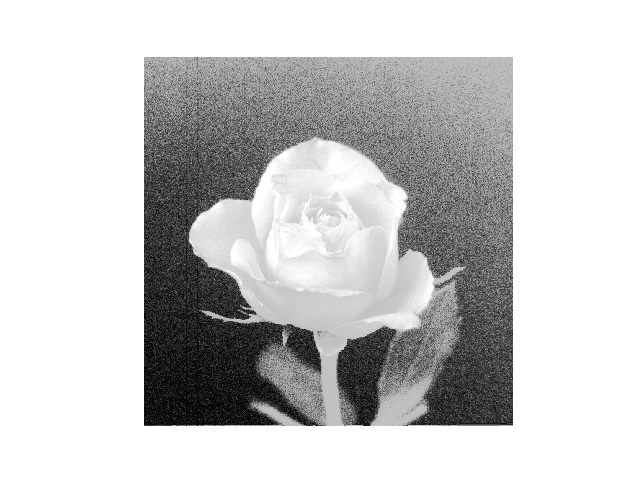
calHist()完成的是对图像进行直方图统计，返回的是一个256维的数组，对应的是图像在[0, 255]灰度级上像素的个数。编写这个函数的目的主要是为了方便绘制一个图像的灰度直方图。

histequal4e()进行的是直方图均衡化，首先调用calHist()统计直方图，之后将这个数组归一化，再对概率密度进行累计，从而得到均衡化之后的直方图，然后再将概率映射到8位灰度上以进行显示。

图片包含 屏幕截图

描述已自动生成

通过与下图，调用的OpenCV中cv2.equalHist()函数的结果进行比较，发现本函数在结果上没有问题。



问题**2** 读入清晰图像，加上椒盐噪声，采用有选择保边缘平滑法对图像进行平滑

本问题编写的函数有：add\_noise()，padding()，smooth\_one\_pixel()，def smoother()。

add\_noise()是用来增加椒盐噪声的函数。首先根据信噪比确定向图像中加入多少噪点，共进行这些次循环。每次循环中，随机得到图像的一个像素，然后按1/2的概率对其赋0或255。

padding()是用来进行填充的函数。零填充：按照图像和卷积核的大小定义一个全是0的矩阵，并将原图拷贝到中央，这便实现了零填充。像素复制填充：在上述矩阵基础上，以第一个不全为0的行填充矩阵上部的0，再以第一个不全为0的列填充矩阵左部的0，这样也把左上角填充成了离其最近的元素。

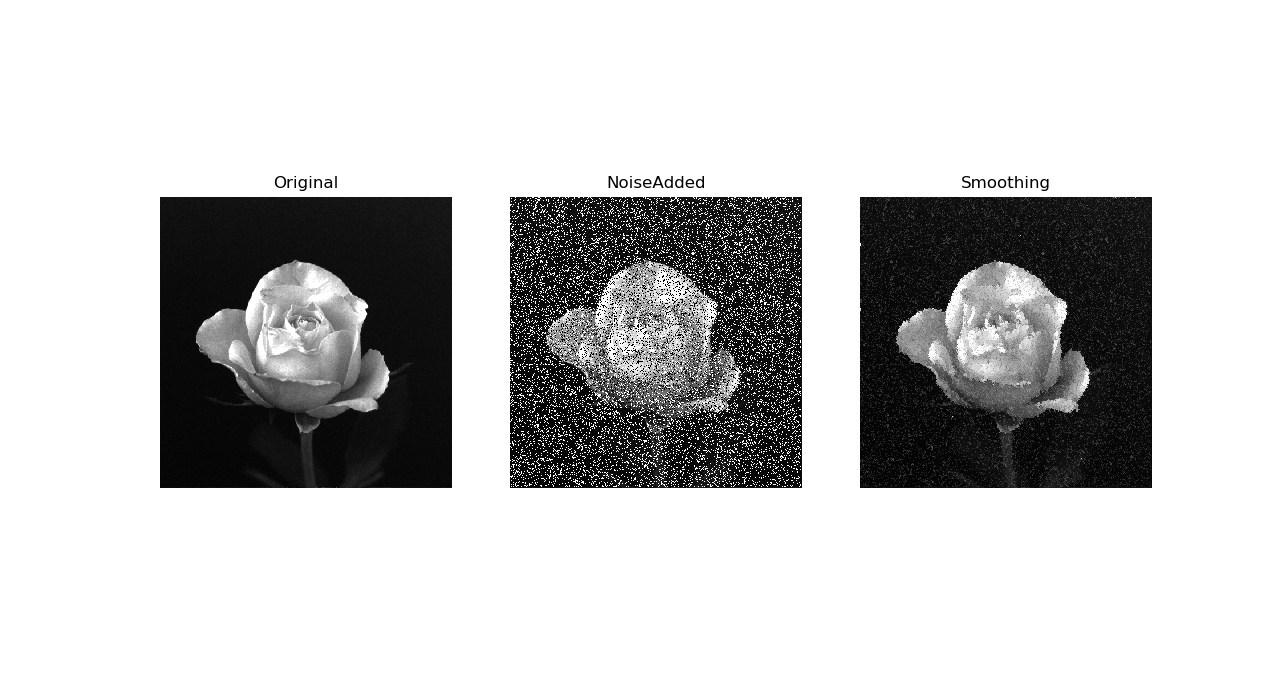
smooth\_one\_pixel()是用来对图像的某个像素进行平滑的函数，其参数为该像素的(5, 5)邻域。首先利用中间的3\*3正方形的方差和均值作为初始值。

之后先计算4个五边形掩模，可以知道，每个五边形掩模是该区域中心点加上4个方向上的6个点，因此可以进行循环分别来取这6个点。预先定义好循环边界的表，之后对每一个掩模，先计算方差是否更小，如果是的话再计算其均值，并更新最小值。

之后计算4个六边形掩模。观察到六边形是4个3\*3的正方形减去角上的2个点，因此预先设计一个表格来遍历这4个正方形。又观察到，被减去的点无论是哪个六边形掩模都不会取到，因此用这个作为判别条件即在每个掩模计算时减去2个点。

最后返回最小方差对应的均值即是该像素平滑后的值。

smoother()是对图像进行平滑的函数，调用的是padding()进行填充，然后对每个像素调用smooth\_one\_pixel()进行平滑即可。结果如下：



问题**3** 编写一个程序完成拉普拉斯增强

和本问题相关的共有4个函数：padding()，flip180()，twodConv()以及map2Uint8()。

padding()和问题2是相同的。

flip180()是用来将矩阵旋转180°，从而实现卷积操作。首先将矩阵展成向量，再倒序排列，然后reshap成原先的大小，即完成了180°旋转。

twodConv()是卷积函数。首先调用padding()进行填充，对卷积核进行旋转180°。之后对填充后的矩阵进行卷积操作，从矩阵中按照卷积核大小复制出来一个卷积区域，再把这个区域和卷积核的卷积结果填充到结果矩阵中。

map2Uint8()是用来把数据映射到[0, 255]区间内。由于拉普拉斯增强会产生负值，因此需要将其映射到8位灰度范围内。具体做法是每个像素减去全局最小灰度值来正值化，然后除以全局最大灰度值来归一化，再乘上255进行映射。

实验结果如图：



调用OpenCV中的Laplacian(I, cv2.CV\_16S, 3)进行比较，其结果如下，发现自己编写的程序该结果正确。

