Week4

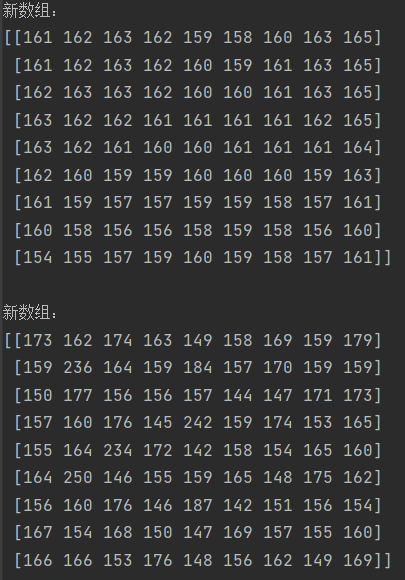
使用语言：python

**源码：**

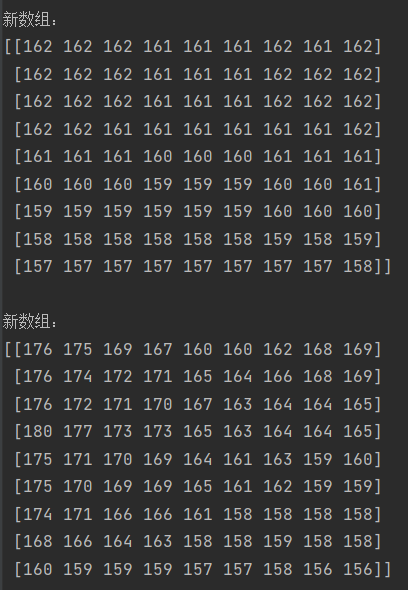
import cv2 as cv  
import numpy as np  
import matplotlib.pyplot as plt  
  
# 转换为灰度图像  
imgG1 = cv.imread("image/lena.jpg", 0)  
cv.imwrite("image/lenaG.jpg", imgG1)  
imgG2 = cv.imread("image/images.jpg", 0)  
cv.imwrite("image/imagesG.jpg", imgG2)  
  
  
# 定义得到前9行与前9列交叉的81个像素的灰度矩阵（9X9）  
def get\_img\_array(grayImg):  
    new\_img\_array = np.zeros((9, 9), np.uint8)  
    for i in range(9):  
        for j in range(9):  
            gray = int(grayImg[i, j])  
  
            new\_img\_array[i, j] = np.uint8(gray)  
    print(f&apos;新数组：\n{new\_img\_array}\n&apos;)  
    return new\_img\_array  
  
  
# 输出G1,G2的前9行与前9列交叉的81个像素的灰度矩阵H1,H2（9X9）  
H1 = get\_img\_array(imgG1)  # H1  
H2 = get\_img\_array(imgG2)  # H2  
  
# 均值滤波  
G11 = cv.blur(imgG1, (5, 5))  # 可以更改核的大小  
G21 = cv.blur(imgG2, (5, 5))  
# 输出G11，G21的前9行与前9列交叉的81个像素的灰度矩阵H11,H21(9X9)  
H11 = get\_img\_array(G11)  # H1  
H21 = get\_img\_array(G21)  # H2  
  
# L3部分  
# 高斯滤波  
G12 = cv.GaussianBlur(imgG1, (5, 5), 0)  
G22 = cv.GaussianBlur(imgG2, (5, 5), 0)  
# 中值滤波  
G23 = cv.medianBlur(imgG2, 5)  
# 显示图像  
cv.imshow("LENA GRAY G1,NoiseGray G2", np.hstack([imgG1, imgG2]))  
cv.imshow("medianBlurG11,medianBlurG21", np.hstack([G11, G21]))  
cv.imshow("GaussianBlurG12,GaussianBlurG22,medianBlurG23", np.hstack([G12, G22, G23]))  
  
# 等待显示  
cv.waitKey(0)  
cv.destroyAllWindows()

**效果图和分析：**

**控制台：H1、H2**

****

**H11、H21**

****

**H1和H11的总体来看从第一排到第九排依次递减，行中中间的数字与两旁相比较小，变成H11后出现相同数字聚集在一起的现象**

**H2矩阵排列相对无序，H21左边的数字整体比右边的大**

**变化原因：均值滤波是指任意一点的像素值，都是周围 N \times M 个像素值的均值。其中5\*5的矩阵称为核（即G11 = cv.blur(imgG1, (5, 5))采用5\*5像素取均值），针对原始图像内的像素点，采用核进行处理，得到结果矩阵**

****

****

**G1变G11和G2变G21也和上诉H1变化类似**

****

**G12和G11相比，G12模糊相对均匀**

**高斯滤波实质上就是一种均值滤波，只是高斯滤波与基本均值滤波不同的是按照加权平均的，距离越近的点权重越大，距离越远的点权重越小。高斯滤波就是对整幅图像进行加权平均的过程，每一个像素点的值，都由其本身和邻域内的其他像素值经过加权平均后得到。高斯滤波让临近的像素具有更高的重要度，对周围像素计算加权平均值，较近的像素具有较大的权重值。**

观察G22与G23，中值滤波器平滑效果好，G23的噪声与G22相比少。

中值滤波是非线性的图像处理方法，在去噪的同时可以兼顾到边界信息的保留。