输出在附录文件夹中

分析：在高斯噪声情况下，高斯模糊能取得相对比较好的结果；在椒盐噪声情况下，中值模糊能取得比较满意的结果（中值模糊能直接取中值而忽略突变的影响）

不论在什么情况下，均值模糊效果都一般，这是由于均值模糊忽视高斯噪声正态分布的性质，且不会忽视椒盐噪声的影响

import cv2 as cv

import numpy as np

def clamp(pv):

*#\*防止溢出*

    if pv > 255:

        return 255

    elif pv < 0:

        return 0

    else:

        return pv

def gaussian\_noise(image):

*#\*增加高斯噪声*

    height, width, channels = image.shape

    for row in range(0, height):

        for col in range(0, width):

            s = np.random.normal(0, 15, 3)

            b = image[row, col, 0]

            g = image[row, col, 1]

            r = image[row, col, 2]

            image[row, col, 0] = clamp(b + s[0])

            image[row, col, 1] = clamp(g + s[1])

            image[row, col, 2] = clamp(r + s[2])

def sp\_noise(image, prob):  *#prob为占比，一般设为0.05*

*#\*增加椒盐噪声*

    height, width, channels = image.shape

    sum\_points = height \* width

    noise\_points = int(sum\_points \* prob)

    for i in range(noise\_points):

        row = np.random.randint(0, height - 1)

        col = np.random.randint(0, width - 1)

        if np.random.random() < 0.5:

            image[row, col] = 0

        else:

            image[row, col] = 255

img = cv.imread('d:\Code\python3\OpenCV\day2picture\SuperSmashBros.jpg')

sp\_noise(img, 0.05)

*#gaussian\_noise(img)*

*#img = cv.GaussianBlur(img, (5, 5), 0)  #高斯矩阵长宽都为5，标准差为0*

*#img = cv.blur(img, (3, 3))*

img = cv.medianBlur(img, 3)

cv.imwrite('day2picture/mBlur\_spNoise.jpg', img)

cv.imshow('demo', img)

cv.waitKey(0)

cv.destroyAllWindows()