提取圖像(有noise)

Noise處理方法: 和鄰近的pixel取平均值, 如:3x3 or 5x5

先讀取圖像的大小 sharp

根據大小建立一個3x3都為1的距陣和圖像每點相乘

排列這9個數並將中間的數取出並替代3x3卷積核的值

跑完整個圖像達到降噪效果

import numpy as np  
import cv2  
import copy  
  
  
def medFilter(img, imgS, filS):  
 temp = []  
 imgTempf = copy.deepcopy(img)  
 for k in range(imgS[2]): # use k as index of image to determine which pixels need to be filtered  
 for i in range(imgS[0]): # use i as index of image to determine which pixels need to be filtered  
 for j in range(imgS[1]):  
 imgTempf = conv(imgTempf, imgS, filS, temp, i, j, k)  
 return imgTempf  
  
  
def conv(imgOrignal, imgSS, filSS, tempp, ii, jj, kk):  
 imgNew = copy.deepcopy(imgOrignal)  
 for l in range(filSS[0]): # if 3x3 kernel, l = 0, 1, 2  
 if ii + l < imgSS[0]:  
 for m in range(filSS[1]):  
 if jj + m < imgSS[1]:  
 tempp.append(imgOrignal[ii + l][jj + m][kk]) # save pixel value of image  
 else:  
 break  
 else:  
 break  
 temppSort = sorted(tempp) # sort the pixel  
 num = int(len(temppSort) / 2) # to find the middle index of pixel  
  
 if (ii + 1 < imgSS[0]) and (jj + 1 < imgSS[1]):  
 imgNew[ii + 1][jj + 1][kk] = temppSort[num] # put the middle index of pixel value to the core of filter location of image  
 tempp.clear()  
 temppSort.clear()  
 return imgNew  
  
  
image = cv2.imread('mario.jpg')  
imgTemp = copy.deepcopy(image)  
arr = np.array(imgTemp)  
imgSize = arr.shape  
  
fNum = 3  
kernel = []  
for i in range(fNum):  
 kernel.append([1] \* fNum)  
  
fil = np.array(kernel)  
filSize = fil.shape  
print('kernel: ', kernel)  
print('filterSize: ', filSize)  
  
imgTemp2 = medFilter(imgTemp, imgSize, filSize)  
  
print('imageSize: ', imgSize)  
print('end')  
cv2.imshow('Before', image)  
cv2.imshow('After', imgTemp2)  
cv2.waitKey(0)