import math  
import os  
import numpy as np  
from numpy import fft  
import cv2  
import matplotlib.pyplot as plt  
  
plt.rcParams[**'font.sans-serif'**] = [**'SimHei'**]  
plt.rcParams[**'axes.unicode\_minus'**] = False  
  
  
*# 仿真运动模糊*def motion\_process(image\_size, motion\_angle):  
 PSF = np.zeros(image\_size)  
 center\_position = (image\_size[0] - 1) / 2  
 slope\_tan = math.tan(motion\_angle \* math.pi / 180)  
 slope\_cot = 1 / slope\_tan  
 if slope\_tan <= 1:  
 for i in range(15):  
 offset = round(i \* slope\_tan) *# ((center\_position-i)\*slope\_tan)* PSF[int(center\_position + offset), int(center\_position - offset)] = 1  
 return PSF / PSF.sum() *# 对点扩散函数进行归一化亮度* else:  
 for i in range(15):  
 offset = round(i \* slope\_cot)  
 PSF[int(center\_position - offset), int(center\_position + offset)] = 1  
 return PSF / PSF.sum()  
  
  
*# 对图片进行运动模糊*def make\_blurred(input, PSF, eps):  
 input\_fft = fft.fft2(input) *# 进行二维数组的傅里叶变换* PSF\_fft = fft.fft2(PSF) + eps  
 blurred = fft.ifft2(input\_fft \* PSF\_fft)  
 blurred = np.abs(fft.fftshift(blurred))  
 return blurred  
  
  
*# 逆滤波*def inverse(input, PSF, eps):  
 input\_fft = fft.fft2(input)  
 PSF\_fft = fft.fft2(PSF) + eps *# 噪声功率，这是已知的，考虑epsilon* result = fft.ifft2(input\_fft / PSF\_fft) *# 计算F(u,v)的傅里叶反变换* result = np.abs(fft.fftshift(result))  
 return result  
  
*# 维纳滤波，K=0.01*def wiener(input, PSF, eps, K=0.01):  
 input\_fft = fft.fft2(input)  
 PSF\_fft = fft.fft2(PSF) + eps  
 PSF\_fft\_1 = np.conj(PSF\_fft) / (np.abs(PSF\_fft) \*\* 2 + K)  
 result = fft.ifft2(input\_fft \* PSF\_fft\_1)  
 result = np.abs(fft.fftshift(result))  
 return result  
  
  
def put(path):  
 img = cv2.imread(path, 1)  
 img = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY)  
 img\_h = img.shape[0]  
 img\_w = img.shape[1]  
 *# 进行运动模糊处理* PSF = motion\_process((img\_h, img\_w), 60)  
  
 blurred = np.abs(make\_blurred(img, PSF, 1e-3))  
 *# 逆滤波* res1 = inverse(blurred, PSF, 1e-3)  
 *# 维纳滤波* res2 = wiener(blurred, PSF, 1e-3)  
  
 *# 添加噪声,standard\_normal产生随机的函数* blurred\_noisy = blurred + 0.1 \* blurred.std() \* np.random.standard\_normal(blurred.shape)  
 *# 对添加噪声的图像进行逆滤波* res3 = inverse(blurred\_noisy, PSF, 0.2 + 1e-3)  
 *# 对添加噪声的图像进行维纳滤波* res4 = wiener(blurred\_noisy, PSF, 0.2 + 1e-3)  
 plt.subplot(2, 4, 1), plt.axis(**'off'**), plt.imshow(img, plt.cm.gray), plt.title(**' 原图'**)  
 plt.subplot(2, 4, 2), plt.axis(**'off'**), plt.imshow(blurred, plt.cm.gray), plt.title(**'运动模糊'**)  
 plt.subplot(2, 4, 3), plt.axis(**'off'**), plt.imshow(res1, plt.cm.gray), plt.title(**' 逆滤波'**)  
 plt.subplot(2, 4, 4), plt.axis(**'off'**), plt.imshow(res2, plt.cm.gray), plt.title(**' 维纳滤波(k=0.01)'**)  
 plt.subplot(2, 4, 5), plt.axis(**'off'**), plt.imshow(img, plt.cm.gray), plt.title(**' 原图'**)  
 plt.subplot(2, 4, 6), plt.axis(**'off'**), plt.imshow(blurred\_noisy, plt.cm.gray), plt.title(**'有噪声且运动模糊'**)  
 plt.subplot(2, 4, 7), plt.axis(**'off'**), plt.imshow(res3, plt.cm.gray), plt.title(**' 逆滤波'**)  
 plt.subplot(2, 4, 8), plt.axis(**'off'**), plt.imshow(res4, plt.cm.gray), plt.title(**' 维纳滤波(k=0.01)'**)  
  
 *# plt.savefig('3.new.jpg')* plt.show()  
  
  
*# 图像处理函数，要传入路径*put(**r'../images/arnold-b-cklin\_fighting-on-a-bridge.jpg'**)

参考链接：

https://blog.csdn.net/bingbingxie1/article/details/79398601