import numpy as np

import matplotlib.pyplot as plt

import matplotlib.image as mpimg

# leitura da imagem

imagemOrignal = mpimg.imread('../image\_(2).jpg')

# aplicação da media alfa cortada numa area selecionada

def filtroMediaAlfaCortada(area, size, corte):

# soma todos os termos na area

soma = np.sum(area)

# caso maior ou igual ao size é mantido como um filtro de mediana

if(corte > size\*\*2-1):

return soma

# caso contrário é convertido num intermediario ou em um filtro de

# media aritimetica

else:

return soma/(size\*\*2-corte)

# aplicação da media harmonica area selecionada

def filtroMediaHarmonica(area, size):

# soma todos os termos invertidos na area,

# somando um para evitar divisão por zero

inverted = 1/(area+1)

soma = np.sum(inverted)

# se for infnito o loop verifica qual o valor zero dentro da area analisada e

# aplica 1/(area+1) neste caso em especial

if(soma == float('Inf')):

soma = 0

for line in area:

for pixel in line:

if(pixel != 0):

soma = soma + 1/pixel

else:

soma = soma + 1/(pixel+1)

return (size\*\*2)/(soma+1)

def aplicaMediaAlfaCortada(imagemOriginal, corte=7, size=12):

# se estabelece um limite para as fronteiras das áreas analisadas durante o loop

limit = int(size/2)

X = imagemOriginal.shape[0]

Y = imagemOriginal.shape[1]

imagemReturn = [[]]

aux = 0

for j in range(limit, int(X)-limit) :

for i in range(limit, int(Y)-limit):

aux += 1

if(aux%100000 == 0):

print(100\*aux/((int(X)-limit)\*(int(Y)-limit)))

# se estabelece a área por meio dos limites atribuidos previamente e

# da posição atual do loop

area = imagemOriginal[(j-limit):(j+limit),(i-limit):(i+limit)]

pixel = filtroMediaAlfaCortada(area, size, corte)

# acrescenta o pixel resultado do calculo de mediaAlfaCortada na linha da

# imagem que será retornada equivalente a imagem original

imagemReturn[j-limit] = imagemReturn[j-limit] + [pixel]

if(j<X-limit-1):

imagemReturn += [[]]

return np.array(imagemReturn, dtype=np.uint8)

def aplicaMediaHarmonica(imagemOriginal, size=10):

# se estabelece um limite para as fronteiras das áreas analisadas durante o loop

limit = int(size/2)

X = imagemOriginal.shape[0]

Y = imagemOriginal.shape[1]

imagemReturn = [[]]

aux = 0

for j in range(limit, int(X)-limit) :

for i in range(limit, int(Y)-limit):

aux += 1

if(aux%100000 == 0):

print(100\*aux/((int(X)-limit)\*(int(Y)-limit)))

# se estabelece a área por meio dos limites atribuidos previamente e

# da posição atual do loop

area = imagemOriginal[(j-limit):(j+limit),(i-limit):(i+limit)]

pixel = filtroMediaHarmonica(area, size)

# acrescenta o pixel resultado do calculo de mediaAlfaCortada na linha da

# imagem que será retornada equivalente a imagem original

imagemReturn[j-limit] = imagemReturn[j-limit] + [pixel]

if(j<X-limit-1):

imagemReturn += [[]]

return np.array(imagemReturn, dtype=np.uint8)

# ploting

(fig, (ax1, ax2)) = plt.subplots(1,2)

ax2.imshow(aplicaMediaHarmonica(imagemOrignal), cmap='gray')

ax1.imshow(imagemOrignal, cmap='gray')

plt.show()