|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Université de Badji-Mokhtar Annaba  Département d’informatique  2020-2021 | | **Exercice 3** | | | Promotion : Master I - IATI  Cours : Prétraitement |
| Soit I une image représentée par la matrice de la figure 1. Soit I’ l’image dégradée de I représentée par la matrice de la figure 2. L’objectif de ce problème est de rapprocher I’ de I.  On dit que deux images sont proche si leur mesure de similarité Ms est inférieure au seuil S=0.05. | | | | | |
| Figure 1 : Matrice I | |  | Figure 2 : Matrice I’ | | |

Ms est définie par :

1. Calculer l’histogramme de l’image I
2. Calculer l’histogramme de l’image I’
3. Compléter le code python ci-contre qui calcul la mesure de similarité Ms

def Calcul\_Ms(h1,h2):

s11=s12=s21=s22=0

for i in range(19):

s11+=h1[i]

s12+=………………………

for i in range(………,25):

…………………………………

…………………………………

ms=(abs(s11-s12)+abs(s21-s22))/N

return …………………..

1. Donner la valeur de la mesure de similarité entre I et I’
2. Appliquer le moyenneur (3x3) à l’image I’ et récupérer l’image Imoy
   * 1. Calculer l’histogramme de l’image Imoy
     2. Donner la mesure de similarité entre I et Imoy
3. Appliquer le filtre Max (3x3) à l’image I’ et récupérer l’image Imax
   * 1. Calculer l’histogramme de l’image Imax
     2. Donner la mesure de similarité entre I et Imax
4. Appliquer le filtre médian (3x3) à l’image I’ et récupérer l’image Imed
   * 1. Calculer l’histogramme de l’image Imed
     2. Donner la mesure de similarité entre I et Imed
5. En comparant les mesures de similarités déduire la méthode de filtrage la plus performante

**Solution**

1. hist I= [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 2., 5., 3., 0., 0., 0., 0., 0., 2., 8., 6., 8.]
2. hist I’=[0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 2., 4., 3., 0.,0., 0., 0., 1., 2., 8., 6., 8.]
3. code python

def Calcul\_Ms(h1,h2):

s11=s12=s21=s22=0

for i in range(19):

s11+=h1[i]

s12+=**h2[i]**

for i in range(19,25):

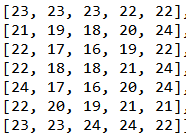
**s21+=h1[i]**

**s22+=h2[i]**

ms=(abs(s11-s12)+abs(s21-s22))/N

return **ms**

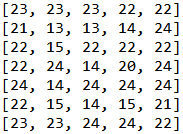
1. s= 0.05714285714285714
2. Imoy =



hist(Imoy)= [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 2.,2., 3., 3., 3., 4., 7., 5., 6.]

s(I,Imoy)= 0.22857142857142856

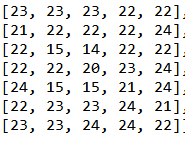
1. Imax=



hist(Imax)= [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 2., 4., 3., 0.,0., 0., 0., 1., 2., 9., 5., 9.]

s= 0.11428571428571428

1. Imed=



hist(Imed)= [ 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,0., 1., 3., 0., 0., 0., 0., 1., 3., 12., 8., 7.]

s= 0.4

1. Aucune méthode n’est bonne

**Code Python**

import numpy as np

from PIL import Image

from pylab import \*

I=np.array([[23, 23, 23, 22, 22],

[21, 13, 13, 14, 24],

[22, 15, 23, 22, 22],

[22, 14, 14, 12, 24],

[24, 14, 24, 24, 24],

[22, 15, 14, 15, 21],

[23, 23, 24, 24, 22]])

P=np.array([[23, 23, 23, 22, 22],

[21, 13, 13, 14, 24],

[22, 15, 23, 22, 22],

[22, 12, 14, 20, 24],

[24, 14, 24, 24, 24],

[22, 15, 14, 15, 21],

[23, 23, 24, 24, 22]])

N=I.size

def hist(im):

l,h =im.shape

h\_r=np.zeros([25])

for i in range(l):

for j in range(h):

h\_r[im[i,j]] += 1

figure()

plt.plot(h\_r)

return h\_r

def similarite(h1,h2):

s11=s12=s21=s22=0

for i in range(19):

s11+=h1[i]

s12+=h2[i]

for i in range(19,25):

s21+=h1[i]

s22+=h2[i]

s=(abs(s11-s12)+abs(s21-s22))/N

return s

def convolve(s,k):

b=np.copy(s)

for i in range(1,s.shape[0]-1):

for j in range(1,s.shape[1]-1):

b[i,j]=int((1/9)\*((s[i-1,j-1]\*k[0,0])+(s[i-1,j]\*k[0,1])+(s[i-1,j+1]\*k[0,2])+(s[i,j-1]\*k[1,0]) +(s[i,j]\*k[1,1]) +(s[i,j+1]\*k[1,2])+(s[i+1,j-1]\*k[2,0])+(s[i+1,j]\*k[2,1])+(s[i+1,j+1]\*k[2,2])))

return b

def MinMax(s):

bmax=np.copy(s)

bmin=np.copy(s)

for i in range(1,s.shape[0]-1):

for j in range(1,s.shape[1]-1):

voisin=np.array([s[i-1,j-1],s[i-1,j],s[i-1,j+1],s[i,j-1],s[i,j+1],s[i+1,j-1],s[i+1,j],s[i+1,j+1]])

mini=np.min(voisin)

maxi=np.max(voisin)

if (s[i,j]<mini)or(s[i,j]>maxi):bmin[i,j]=mini

if (s[i,j]<mini)or(s[i,j]>maxi):bmax[i,j]=maxi

return bmin,bmax

def Med(s):

bmed=np.copy(s)

for i in range(1,s.shape[0]-1):

for j in range(1,s.shape[1]-1):

voisin=np.array([s[i-1,j-1],s[i-1,j],s[i-1,j+1],s[i,j-1],s[i,j],s[i,j+1],s[i+1,j-1],s[i+1,j],s[i+1,j+1]])

voisin.sort()

bmed[i,j]=voisin[4]

return bmed