Centre Universitaire de Mila

Institut Sciences et Technologies — 1ière Année Master STIC / I2A

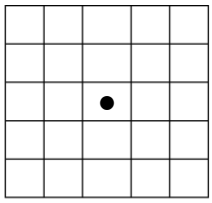
Département Mathématiques et Informatique — Mila le 18-12-2022

Interrogation en traitement d'image

I) PARTIE COURS/TD

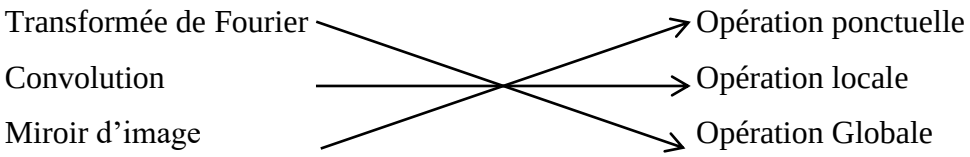
EXERCICE 1 (10 points):

1) Le nombre de bit pour enregistrer une image est $b = M \times N \times K$ avec $K = \log_2(L)$ **(VRAI).**
2) Une image a une résolution spatiale de K bits et une résolution de tons de gris de $M \times N$ pixels **(FAUX).**
3) Les formats les plus simples de fichiers images sont les images avec compression **(FAUX).**
4) Donner les distances euclidiennes ente le pixel central (●) et tous les pixels de l'image suivante :



$2\sqrt{2}$	$\sqrt{5}$	2	$\sqrt{5}$	$2\sqrt{2}$
$\sqrt{5}$	$\sqrt{2}$	1	$\sqrt{2}$	$\sqrt{5}$
2	1	0	1	2
$\sqrt{5}$	$\sqrt{2}$	1	$\sqrt{2}$	$\sqrt{5}$
$2\sqrt{2}$	$\sqrt{5}$	2	$\sqrt{5}$	$2\sqrt{2}$

5) Donner le type d'opération appropriée pour les fonctions d'images suivantes :



6) Donner l'algorithme accéléré de la transformation linéaire d'une image.

Utilisation d'une LUT (Look Up Table)

```
/* Initialisation de la LUT */
Pour i=0 à 255
    LUT[i] = 255*(i-min)/(max-min);
/* Transformation d'histogramme */
Pour x=1 à N
    Pour y = 1 à M
      g(x,y) = LUT[f(x,y)];
```

7) Calculer la transformée de Radon $g(\rho, \theta)$ de la fonction $f(x, y)$ définie comme suit:

$$f(x,y) = \begin{cases} A & Si\ x^2 + y^2 \leq r^2 \\ 0 & Sinon \end{cases}$$

Résultat :

$$g(\rho,\theta) = \begin{cases} 2A\sqrt{r^2 - \rho^2} & Si\ |\rho| \leq r \\ 0 & Sinon \end{cases}$$

II) PARTIE TP

EXERCICE 2 (10 points):

1) Donner le programme Python qui permet de calculer la somme suivante en utilisant deux méthodes différentes: (1) la boucle "for", (2) opérations sur les vecteurs:

$$Somme = -\frac{1}{2!} + \frac{2^2}{3!} - \frac{3^2}{4!} + \ \ldots\ldots\ldots - \frac{999^2}{1000!}$$

2) Donner le résultat du programme suivant:

f =

120	70	90	76	150
0	0	150	200	170
130	189	200	200	170
140	145	132	255	388
34	44	123	34	34

```
from scipy.signal import convolve2d

mon_filtre = np.ones((3, 3)) / (3**2)

nouvelle_image = convolve2d(f, mon_filtre)
```

Solution :

1)

a) **Boucle « for »:**

```
def factorielle(n):
    fact=1
    for i in range(1, n+1):
        fact = fact * i
    return fact
S1 = 0
for i in range(1,3):
    S1 = S1 + pow(i,2)/factorielle((i+1))
print(S1)
```

b) **Opérations sur les vecteurs :**

```
from scipy.special import factorial
import numpy as np
v1 = np.arange(1,1000)
v2 = np.arange(2,1001)
c1 = np.power(v1,2)
c2 = factorial(v2)
d = c1/c2
S2 = np.sum(d)
print(S2)
```

2)

21.11	47.78	65.11	92.89	66.22
56.56	105.44	130.56	156.22	107.33
67.11	120.67	163.44	207.22	153.67
75.78	126.33	146.89	170.67	120.11
40.33	68.67	81.44	107.33	79