USTHB- Faculté d'Informatique
Module : Traitement et Analyse d'Images
M1 IV

Série TP N° 3

(Filtres et Convolutions)

Objectifs :

Ce TP a pour but de vous familiariser avec les concepts de base du traitement d'images, En suivant les exercices de ce TP, vous apprendrez à :

- Comprendre le principe de la convolution d'image.
- Explorer les fonctions de convolution d'OpenCV.
- Appliquer différents filtres sur une image.

Chaque partie est composée d'exercices à résoudre. Suivez les instructions et répondez aux questions pour chaque exercice.

1. Convolution avec OpenCV

Instructions :

La convolution est une opération mathématique qui permet de transformer une image en appliquant un noyau (ou filtre) qui modifie les pixels de l'image. Elle est couramment utilisée pour des opérations telles que le floutage, le débruitage et la détection des contours.

Chaque filtre est représenté par une matrice appelée "noyau" qui est appliquée sur chaque pixel de l'image pour produire une nouvelle valeur de pixel en fonction de la moyenne, de la pondération ou de la médiane des pixels environnants.

1. Recherchez dans la documentation OpenCV la fonction permettant d'appliquer une convolution (ou filtre) à une image sur le lien suivant : https://docs.opencv.org/4.x/
   Indice : vous le trouverez dans le module **Image processing**
2. Créer un nouveau script python sur **vscode**
3. Enregistrer le fichier sous le nom : **TAI_TP3.py**
4. Téléchargez l'image jointe à ce fichier (lena_noise.jpg) et placez-la dans le même répertoire que votre script.
5. Importez les bibliothèques nécessaires : cv2, matplotlib.pyplot et numpy
6. Definisser les filtres suivants:
   - Filtre moyen
   - Filtre median
7. En consultant la documentation OpenCV appliquer une convolution en utilisant les différents filtres sur l'image lena_noise.jpg.
8. Faites varier la taille de noyaux (filtres) afficher les résultats pour K=3, 5, 7, 11
9. Créer un noyau quelconque et appliquer le sur l'image lena_noise.jpg

2. Convolution « from scratch ».

Voici le code Python suivant :

USTHB- Faculté d'Informatique
Module : Traitement et Analyse d'Images
M1 IV

Série TP N° 3

(Filtres et Convolutions)

```python
import cv2
import numpy as np
img = cv2.imread("image.jpg", cv2.IMREAD_GRAYSCALE)

kernel = np.array([[1, 0, -1],
                   [0, 1, 0],
                   [1, 0, -1]])

def convolutionW(img, kernel):
    h, w = img.shape
    kh, kw = kernel.shape

    new_h = h - kh + 1
    new_w = w - kw + 1
    output = np.zeros((new_h, new_w), dtype=float)


    for i in range(new_h):
        for j in range(new_w):
            region = img[i:i+kh, j:j+kw]
            output[i, j] = region * kernel

    return output

result = convolutionW(img, kernel)

# Normalisation pour affichage
result = cv2.normalize(result, None, 0, 255, cv2.NORM_MINMAX)
result = np.uint8(result)

cv2.imshow("Result", result)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

1. Que fait ce code?
2. Quelles sont les erreurs présentes dans ce code ?
3. Corriger le code afin d'appliquer une convolution (n'oubliez pas d'inverser le filtre dans les deux axes)

$$(f \star g)(t) \stackrel{\text{def}}{=} \sum_{x=-\infty}^{\infty} f(x)g(t-x)$$

On souhaite maintenant améliorer la fonction de convolution précédente en ajoutant un padding autour de l'image pour conserver les mêmes dimensions en sortie.

Pour un noyau de taille $k \times k$ appliqué sur une image de taille $M \times N$, le padding nécessaire pour conserver la même taille est :

Série TP N° 3

(Filtres et Convolutions)

$$p = \frac{k-1}{2}$$

Ainsi :

Le nombre de pixels ajoutés sur chaque bord est p.
La nouvelle taille de l'image devient : $(M+2p)\times(N+2p)$

4. Utilisez la fonction NumPy suivante pour créer un padding de zéros autour de l'image : **padded = np.pad(img, ((p, p), (p, p)), mode='constant', constant_values=0)**
5. Modifiez le code précédant pour appliquer la convolution avec padding.
6. Appliquez la convolution paddée sur lena_noise.jpg

3. Filtre Gaussien.

1. Recherchez dans la documentation OpenCV la fonction permettant d'appliquer un flou gaussien.

Le filtre gaussien utilise une distribution gaussienne pour créer un noyau de convolution.

Le noyau gaussien dépend de la taille de la fenêtre et du paramètre σ (écart-type). Le noyau, est calculé en utilisant la formule suivante :

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

2. Ecrire une fonction **filtre_Gaussien** qui retourne le noyau de convolution gaussien et prend en paramètre d'entrée (σ, et la taille du noyau)
3. Afficher les valeurs du filtre gaussien pour K=3 et σ =1
4. Appliquer le filtre gaussien sur l'image donnée à l'aide de la fonction convolution définie dans l'exo précédant avec σ =1 et (K=3)
5. Afficher l'image filtrée
6. Appliquer différentes tailles de noyau et différents sigma (par exemple, 5x5 et 7x7) et la valeur de σ=0.5 et, σ=1, σ=2.
7. Afficher les résultats.