L3 SI

Traitement d'image
TP N° 1 : filtrage linéaire

A/ Problème

On désire comparer les performances de deux filtres linéaires : le filtre de la moyenne et le filtre gaussien. L'image utilisée pour les différents tests est une image bruitée synthétique dont les paramètres (variance du bruit …) sont totalement maîtrisés.

Les performances sont évaluées à l'aide du critère des moindres carrés suivant :

$$\sigma^2 = \frac{1}{NM} \sum_{i=1}^{N} \sum_{j=1}^{M} \left(I(i,j) - I_f(i,j) \right)^2 \qquad (1)$$

où I et I_f sont respectivement les images non bruitée et bruitée puis filtrée.

1/ A quelle grandeur correspond le paramètre σ^2 ?

B/ Manipulation

Pour les besoins de cette séance, il faut importer les modules suivants :

```
>>import numpy as np
>>import cv2 as cv
>>from matplotlib import pyplot as plt
```

1/ Chargement et affichage de l'image non bruitée.
A partir de l'interpréteur python de votre choix, charger l'image stockée sur le DD :

```
>>im = cv.imread('photophore.tif', cv.IMREAD_GRAYSCALE)
```

Cette commande permet de créer en mémoire un objet **`ndarray`** du module **`numpy`**, identifié par la variable **`im`** et correspondant à l'image en niveau de gris.

Nous utiliserons le module pyplot pour afficher cette image.

```
>>plt.imshow(im, cmap='gray')
>>plt.show()
```

2/ Réalisation de l'image synthétique.

L'image **`im`** constitue la référence non bruitée à partir de laquelle nous allons construire l'image **`imb`** en ajoutant à cette image un bruit gaussien d'écart type **`et = 5`**.
Pour des raisons de simplicité de manipulation, il est recommandé de charger spécifiquement la fonction **`randn()`** du module numpy :

```
>>from numpy.random import randn

>>et = 5
>>imb = im + et*randn(im.shape[0], im.shape[1])
```

Visualiser les deux images et constater les effets du bruit pour différentes valeurs de **`et`**.

n.b. : Pour afficher les deux images simultanément, il est possible de faire apparaître une nouvelle fenêtre dans laquelle sera affichée la nouvelle image à l'aide de **`plt.figure()`** :

(*affichage première image*)
plt.figure()
(*affichage deuxième image*)

3/ Filtre moyen

On se propose de comparer deux filtres de la moyenne de tailles respectives 3x3 et 5x5.

Réaliser les deux masques des filtres. S'assurer que le filtre sont bien normalisés (de poids 1) de telle sorte que le niveau moyen de l'image ne soit pas affecté.

```
>>flt1 = np.ones((3,3)) / 9
>>flt2 = np.ones((5,5)) / 25
```

En utilisant l'opérateur de convolution du module OpenCV, réaliser le filtrage par les deux filtres ainsi créés :

```
>>imf1 = cv.filter2D(imb, -1, flt1, borderType=cv.BORDER_CONSTANT)
>>imf2 = cv.filter2D(imb, -1, flt2, borderType=cv.BORDER_CONSTANT)
```

Visualiser les différentes images filtrées. Que constate-t-on ?

4/ Filtre gaussien

Sur le même principe on réalise le filtrage à l'aide de deux filtres gaussiens de taille 3x3 et 5x5.
Rappel : L'expression de la fonction de Gauss (non normalisée) est donnée par la relation :

$$f(x,y)=\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)$$

Pour réaliser un filtre Gaussien de taille nxn il est nécessaire de procéder comme suit :

Construire à l'aide de la fonction **mesgrid()** du module **numpy** deux masques X et Y qui contiennent respectivement les coordonnées x et y du pixel considéré.

Exemple : l'instruction

```
>>X, Y = np.meshgrid(range(-1,2), range(-1,2))
```

construit les masques **X** et **Y** qui contiennent respectivement les coordonnées x et y d'un filtre 3x3 :

```
X = array([[-1,  0,  1],          Y = array([[-1, -1, -1],
           [-1,  0,  1],                     [ 0,  0,  0],
           [-1,  0,  1]])                    [ 1,  1,  1]])
```

A partir de ces masques on construit le filtre normalisé :

```
>>fgauss = np.exp(-(x*x + y*y) / (2*sigma*sigma))
>>fgauss = fgauss/sum(sum(fgauss))
```

Réaliser les 2 filtres (normalisés) en s'assurant que la taille n du filtre soit impaire et soit égal à 6 σ. Filtrer l'image test, visualiser les résultats.

5/ Comparaison des performances.
Calculer les performances des filtres en utilisant l'équation (1). Classer les 4 filtres en fonction de leur performance respective.