M4201Cin TP 4

Filtrage et débruitage

2 mars 2020 – B. COLOMBEL

Table des matières

Les réponses aux questions des exercices 1 à 3 seront écrites dans l'éditeur Scinote dans dans 3 fichiers différents (1 par partie).
Les réponses autres que des fonctions Scilab seront commentées. Ces trois fichiers seront nommés :

`Nom_Prenom-TP4-exo?.sci`

et déposés dans le dossier prévu à cet effet dans AMETICE (module M4201Cin, TP4)

Si le module SIVP de **Scilab** est bien installé, vous verrez au démarrage :

```
Initialisation :
  Chargement de l'environnement de travail

SIVP - Scilab Image and Video Processing Toolbox
        load macros
        load gateways
        load help
        load demos
```

Sinon, installez le :

Applications » Gestionnaire de modules - ATOMS » Traitements des images » SIVP

Cliquer sur installer puis redémarrer **Scilab**.

⚠ Pour simplifier, nous n'utiliseraons que des images **renormalisées** c'est-à-dire que l'on a multiplié les valeurs de chaque pixel por qu'elles soient comprises entre 0 et 1.
Il suffit pour cela, si l'image est quantifiée sur b bits, de diviser l'image par 2^{b-1} (qui est la valeur maximale possible pour un pixel).

1 Bruiter une image

Lors de l'acquisition, de la transmission ou de la compression d'une image, il peut apparaître de nombreuses dégradations. Un des domaines principaux en traitement d'image consiste à traiter et corriger ces dégradations pour obtenir une image de meilleure qualité. On s'interesse ici à deux types de dégradations fréquemment rencontrées dans les images :

— Le **bruit additif**, qui affecte tous les pixels de l'image. Dans ce TP, nous considérerons un **bruit blanc additif Gaussien**, de moyenne nulle et de variance σ^2. Il s'agit d'un modèle fréquemment utilisé en première approximation pour modéliser le bruit d'acquisition et de lecture (si l'on ne dispose pas d'un modèle plus raffiné). Le bruit Gaussien affecte à la fois les basses et les hautes fréquences. Il est caractérisé par sa variance σ^2 : plus σ^2 est élevé, plus l'image est dégradée.

```
-->image = imnoise(image0, 'gaussian', m, v)
//Applique un bruit additif gaussien
// de moyenne m et de variance v
```

où le bruit suit une loi gaussienne d'espérance m et de variance `v`. Par défaut `v = 0,04`.

— Le **bruit impulsionnel**, n'affecte que certains pixels de l'image. Dans ce TP, nous considérerons un **bruit poivre et sel**, qui est une dégradation de l'image sous la forme de pixels noirs et blancs répartis au hasard. Ce bruit est dû soit à des erreurs de transmission de données, soit à la défaillance d'éléments du capteur CCD, soit à la présence de particules fines sur le capteur d'images. On le caractérise par le pourcentage p de pixels modifiés : plus p est élevé, plus l'image est dégradée.

```
--> bruitbin = grand(image0,'bin', 1, p);
--> image = max(image0, 255*bruitbin);
```

ou plus simplement en utilisant la fonction `imnoise` de SIVP :

```
--> image = imnoise(image, 'salt & pepper', p); //avec sivp
```

Exercice 1 :

1. Ouvrir l'image `lena.pgm` et la stocker dans la matrice `X1`.
2. Appliquer sur l'image `X1` un bruit blanc Gaussien de variance $\sigma^2 = 0{,}01$ et stocker le résultat dans une matrice `X2`. Afficher sur la même figure l'image originelle et l'image bruitée. Faire varier σ^2 et commenter.
3. Appliquer sur l'image `X1` un bruit poivre et sel avec un pourcentage $p = 0{,}05$ de pixels modifiés et stocker le résultat dans une matrice `X3`. Afficher sur la même figure l'image originelle et l'image bruitée. Faire varier p et commenter.
4. Afficher sur une même figure les images `X1`, `X2` et `X3`. Comparer les effets des deux dégradations et commenter.

2 Filtrage spatial

La fonction `imfilter` permet de faire une convolution d'une image par un « filtre » défini par une matrice F, cette matrice pouvant être créée à partir de la fonction `fspecial`.

Par exemple :

```
--> F = fspecial('average', 3); //filtre moyenneur
--> image_filtree = imfilter(image0, F); //filtrage
```

ou encore :

```
--> F = fspecial('gaussian',3); //filtre gaussien
```

Exercice 2 :

1. Reprendre les images `X1`, `X2` et `X3` précédemment définies. Pour l'image `X2` on prendra $\sigma^2 = 0{,}01$ et pour `X3`, on prendra $p = 0{,}05$.
2. Appliquer un filtre moyenneur de taille 3×3 sur l'image `X2` et stocker le résultat dans `Y2`. Afficher sur la même figure `X1`, `X2` et `Y2`. Le bruit a-t-il été atténué ?
3. Appliquer un filtre médian de taille 3×3 sur l'image `X3` et stocker le résultat dans `Y3`. Afficher sur la même figure `X1`, `X3` et `Y3`. Le bruit a-t-il été atténué ?

Afin de pouvoir quantifier la qualité du débruitage, on va utiliser une mesure objective appelée **Peak Signal to Noise Ratio** (PSNR) et définie par :

$$\mathrm{PSNR} = 10 \log_{10} \left(\frac{R^2}{\frac{1}{MN} \sum_{m=1}^{M} \sum_{n=1}^{N} \left[x^o(m,n) - x^d(m,n) \right]^2} \right)$$

où x^o et x^d sont respectivement les images d'origine et débruitée et où R est la dynamique de l'image (valeur maximale possible pour un pixel).

Remarque. Cette métrique est très largement utilisée pour évaluer les méthodes de compression et de débruitage d'images. Si le PSNR est utile pour mesurer la proximité de l'image débruitée par rapport à l'original au niveau du signal, il ne prend pas en compte la qualité visuelle de reconstruction et ne peut être considéré comme une mesure objective de la qualité visuelle d'une image.

4. Calculer le PSNR pour les deux simulations précédemment réalisées. Sachant qu'on considère en général qu'un excellent débruitage offre un PSNR d'au moins 20 dB, les résultats vous semblent-ils logiques ?
5. Tester les 10 filtres suivants sur `X2`, puis sur `X3`. Lequel donne les meilleures performances sur `X2` ? sur `X3` ?
   (a) Filtre moyenneur : 3×3, 5×5 et 7×7
   (b) Filtre Gaussien de taille 15×15 : $\sigma_h = 2$, $\sigma_h = 1{,}5$, $\sigma_h = 1$ et $\sigma_h = 0{,}5$
   (c) Filtre médian : 3×3, 5×5 et 7×7

$$\begin{array}{ccc} \text{Domaine spatial} & & \text{Domaine fréquentiel} \\ \boxed{\text{Image originale}} & \xrightarrow{\mathrm{TF}} & \boxed{\text{Image spectre}} \\ \Big\downarrow i * h & & \Big\downarrow I \times H \\ \boxed{\text{Image transformée}} & \xleftarrow{\mathrm{TF}^{-1}} & \boxed{\text{Image spectre transformée}} \end{array}$$

où i est l'image originale ; h le filtre spatial ; $I = \mathrm{TF}(i)$ et $H = \mathrm{TF}(h)$

FIGURE 1 – Filtrage fréquentiel

3 Filtrage d'une image : domaine fréquentiel

Le filtrage linéaire consiste en un produit de convolution dans le domaine spatial, ce qui correspond à une multiplication dans le domaine spectral (voir figure 1 page 4).

On s'interesse donc souvent à la réponse fréquentielle d'un filtre pour savoir notamment quelles fréquences il va amplifier, quelles directions privilégiées il va mettre en évidence, etc... En particulier, en observant la transformée de Fourier du masque de convolution (éventuellement complété par des zéros), on arrive à observer le comportement fréquentiel du filtre. Tout comme la transformée de Fourier d'une image classique, on peut représenter la réponse fréquentielle en échelle linéaire ou en échelle logarithmique.

Exercice 3 : On réutilise ici la matrice `X1` de Léna.

1. (a) Générer un masque h_1 correspondant à un filtre moyenneur de taille 3×3.
   (b) Écrire un script qui permet d'afficher sur un même graphique :
   — L'image originelle (en haut à gauche)
   — Le spectre de l'image originelle en échelle linéaire (en haut au milieu gauche)
   — Le spectre de l'image filtrée en échelle linéaire (en haut au milieu droit)
   — La réponse en fréquence du filtre en échelle linéaire (en haut à droite)
   — L'image filtrée (en bas à gauche)
   — Le spectre de l'image originelle en échelle logarithmique (en bas au milieu gauche)
   — Le spectre de l'image filtrée en échelle logarithmique (en bas au milieu droit)
   — La réponse en fréquence du filtre en échelle logarithmique (en bas à droite)
   (c) Quel effet le filtre a-t-il sur le spectre ? S'agit-il d'un filtre passe-bas, passe-haut, etc. ?
   Met-il en évidence des directions particulières ?
2. Refaire la même expérience avec un filtre moyenneur de taille 5×5 et de taille 7×7 et commenter.
3. Refaire la même expérience avec un filtre Gaussien de taille 15×15 et d'écart type $\sigma_h = 2$, $\sigma_h = 1{,}5$ et $\sigma_h = 1$. Commenter.