**Artificial Intelligence based Image Processing (AIIP)**

(mai.nguyen-verger@cyu.fr)

Créer un répertoire de travail portant le nom « AIIP » et sous-répertoires suivants (avant de programmer) :

root/ AIIP /Document
/Programmes
/Rapport

Conseils pratiques **:**
Tout programme commence par :

**%** Titre du programme
**%** Auteur

et les commandes de Matlab suivantes :

***clear all*** ;**%** pour effacer toutes les variables issues de programmes précédemment exécutés.
***close all*** ;**%** pour fermer toutes les figures ouvertes précédemment.

**********

## Partie 1. Création des image numériques

### Partie 1(a). Création des images binaire et d'intensités (image en niveaux de gris)

**1)** Créer un triangle blanc inférieur sur un fond noir de taille 10 x10 pixels (Fig. 1).

**2**) Crée un triangle blanc en haut dans un fond noir de taille NLxNC (avec NL=NC=100) et un triangle blanc en bas dans un fond noir de taille NL x NC, puis les visualiser (Fig. 2a, 2b)

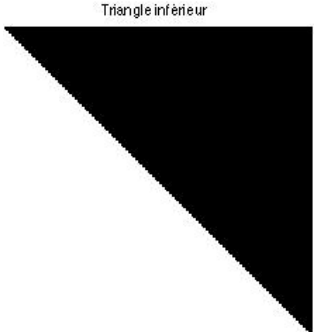



Fig. 1


Fig. 2a

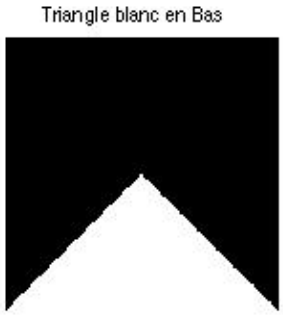



Fig. 2b

**3)** Créer une image binaire alternée de dimension NxN avec N=100 (les pixels dont le numéro est impair ont la valeur 0, pair la valeur 1). Fig.3

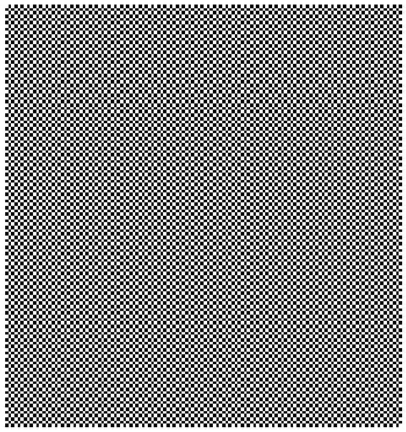


Fig. 3

**4)** Créer une image d'intensités (en niveaux de gris) dégradée en ligne d'un rectangle de taille 10 x 8 (Fig.4)

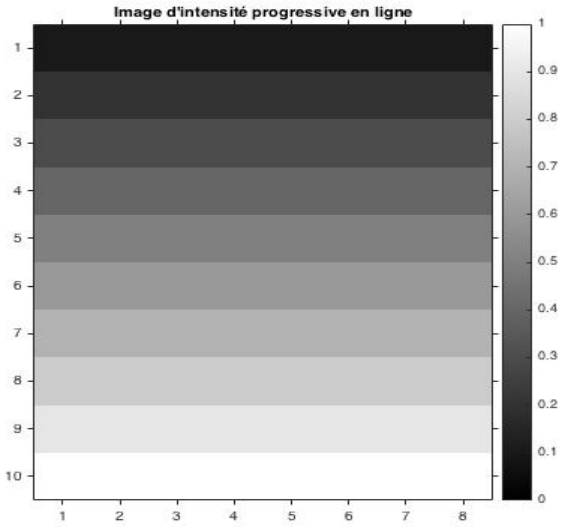



Fig. 4

**5)** Créer une image, de dimension NxN **(avec N=50, puis N=100),** de niveaux de gris progressifs en fonction de position de pixel (la valeur de chaque pixel dépend de la position de ce pixel dans la matrice de NxN). Fig. 5

Attention à la normalisation : les niveaux de gris doivent être entre [0,1] ou bien [0, 255].

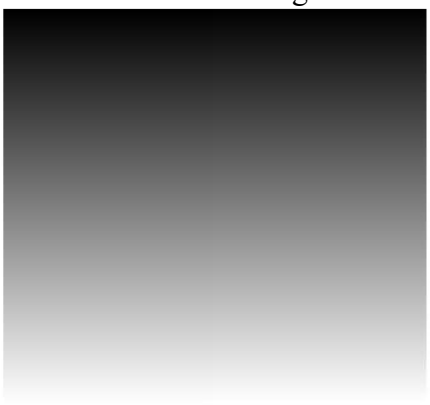


Fig. 5

## Partie 1(b). Image Couleur

Rappel du dosage des couleurs (mesures psychovisuelles) :

Rouge (r=1, g=0, b=0); Vert (r=0, g=1, b=0); Bleu (r=0, g=0, b=1);
Gris (r=g=b=0.5); Jaune (r=1, g=1, b=0); Violet (r=1, g=0, b=1);
Violet (r=1, g=0, b=1); Blanc (r=1, g=1, b=1); Orange (r=1, g=0.5, b=0);
Vert sombre (r=0, g=0.5, b=0)
Bleu-vert (r=0, g=0.6, b=1) ;

**6)** Création des images couleurs **RGB** :

(Rappel sur la Représentation **RGB** : créer trois matrices de couleur suivantes :

```
R= […] ;
G=[…] ;
B=[…] ;
```

puis les concaténer dans une matrice à trois dimensions I par :

```
I( : , : , 1)= R ;
I( : , : , 2)= G ;
I( : , : , 3)= B ;
```

**ou bien** utiliser la fonction de la concaténation **cat**() de Matlab **:**

```
I=cat(3,R,G,B)      % commande pour la concaténation de trois matrices R, G, B
```

)

**6.1**) Création de l'image couleur suivante :

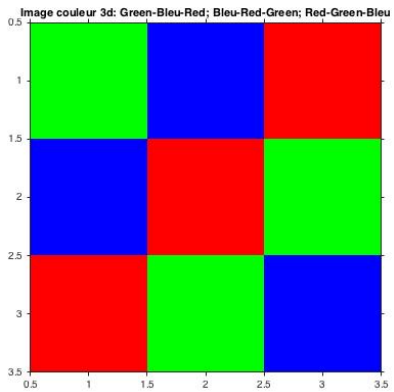



Fig. 6

**6.2)** Créer une image couleur dont l'affichage donne le résultat suivant :

| | | |
|---|---|---|
| jaune | violet | vert sombre |
| orange | blanc | jaune |
| rouge | bleu | Vert |

**6.3)** Modifier les matrices R, G, B pour obtenir l'image suivante :

| | | |
|---|---|---|
| jaune | bleu | rouge |
| vert | blanc | jaune |
| violet | vert sombre | orange |

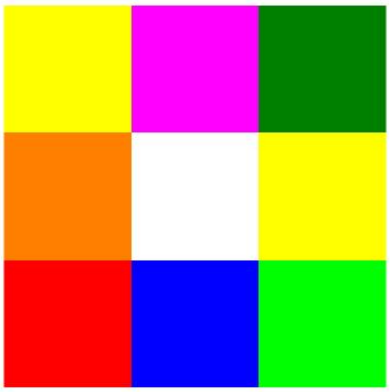


Fig. 7a

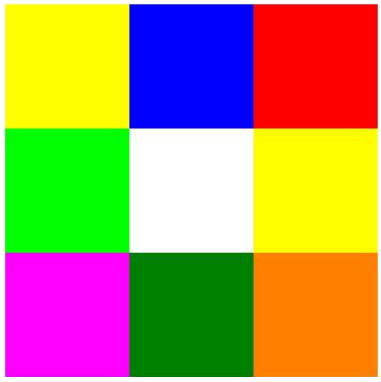


Fig. 7b

**7)** Création des images couleurs **indexées**

(Rappel sur la Représentation **indexée** des images couleurs :
Créer :
- une table de couleurs appelée *map*=[…] qui est une matrice ***nx3*** (où ***n*** est le nombre de couleurs dans l'image) et
- une matrice des index de la même taille de l'image *I*=[….] qui contient les nombres entiers compris entre ***1*** et ***n***, chaque entier joue le rôle d'un index relatif à la couleur dans la table de couleurs, et
- afficher l'image par ***imshow****(I,map).*
)

*map*=[….] ;
*I*=[….] ;
**imshow**(*I*, *map*) ;

**7.1)** Créer la matrice *I* et la table de couleur *map* pour obtenir l'image dans Fig.7a

**7.2)** Modifier *I* et *map* pour obtenir l'image dans Fig.7b

**7.3)** Modifier la table de couleurs comme ci-dessous et afficher de nouveau l'image. Observer le résultat obtenu et expliquer le.

*map*=[0.4 0.2 0.4 ; 0 1 0 ; 1 0 0] ;

## Partie 2. Histogramme des niveaux de gris d'images et transformations d'histogramme

1) Écrire les **algorithmes** du CALCUL, de l'ETIREMENT et de l'EGALISATION de l'histogramme des niveaux de gris d'une image de taille NLxNC et de 256 niveaux de gris entre 0 (noir) et 255 (blanc).

2) Écrire un **programme** qui calcule et affiche les histogrammes des images dans les Fig.1, Fig. 2a, Fig. 2b de la Partie 1.
Tracer une courbe en testant trois fonctions différentes: *plot*(x,y), *stem*(), *bar*().

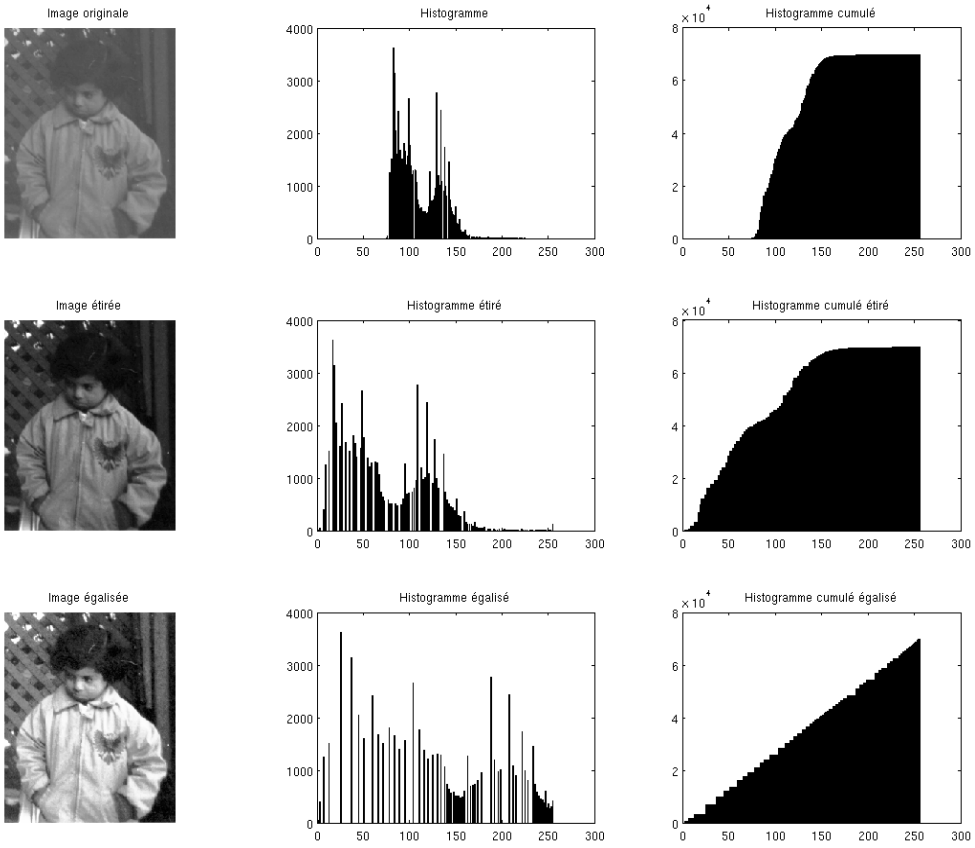
3) En utilisant l'image de test dans le fichier 'histo_imageTest_entree_3D.png' se trouvant dans le répertoire «Images», Écrire les **programmes** qui

- Calcule l'histogramme de l'image originale, affiche l'histogramme et l'histogramme cumulé.

- Modifie les intensités des pixels de l'image pour **étirer et égaliser** l'histogramme, puis afficher les histogrammes et les histogrammes cumulés de l'image étirée et de l'image égalisée.

- Commenter les histogrammes obtenus.

Les résultats sont présentés en colonnes (l'image à gauche, son histogramme et son histogramme cumulé à droite) avec les intitulés des images et les valeurs des axes. Exemples :



## Partie 3. Réduction de bruit (Débruitage)

Rappel sur le filtrage :

- Filtre médian :

Ce filtre affecte à un pixel la valeur médiane des intensités dans son voisinage. Autrement dit, le filtre médian retient une valeur d'intensité telle que dans le voisinage, il y ait d'autant d'intensités supérieures que d'intensités inférieures à cette valeur.
Exemple : un voisinage de 3x3 pixels dans une image :

$$I_{originale} = \begin{pmatrix} 70 & 50 & 30 \\ 40 & \mathbf{20}^* & 70 \\ 60 & 50 & 10 \end{pmatrix}$$

Le filtre médian range par ordre croissant les intensités du voisinage :

10 20 30 40 **50** 50 60 70 70

Il affecte au pixel central (qui valait initialement 2*) l'intensité qui se trouve au milieu du rangement ci-dessus, c'est à dire **5,** donc le résultat est : $I_{mediane} = \begin{pmatrix} 70 & 50 & 30 \\ 40 & \mathbf{50}^* & 70 \\ 60 & 50 & 10 \end{pmatrix}$

- Filtre moyenneur :

Ce filtre remplace le niveau de gris du pixel central par la moyenne des niveaux de gris des pixels autour du pixel central. Pour cela on peut utiliser un masque ***h*** rectangulaire du type (8-connexe) et convoluer l'image originale avec le masque :

$$\boldsymbol{h} = \frac{1}{9}\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix},$$

le résultat est : $I_{moyenne} = \begin{pmatrix} 70 & 50 & 30 \\ 40 & \mathbf{44}^* & 70 \\ 60 & 50 & 10 \end{pmatrix}$

La **taille** du masque est un paramètre variable. Plus la taille du masque est grande plus l'effet du filtrage est fort. Exemple : la taille du masque rectangulaire : 3x3, 5x5, 9x9.

Afin de **conserver la dynamique** des niveaux de gris, il faut que la somme des éléments/coefficiens du masque soit égale à 1.
Pour que le filtrage soit isotopique (même effet dans toutes les directions), le voisinage considéré devra avoir une forme circulaire (filtre circulaire, filtre conique).

L'inconvénient du filtre moyenneur est l'introduction de flou, les contours sont donc dégradés.

- Filtre moyenneur pondéré (filtre Gaussien) :

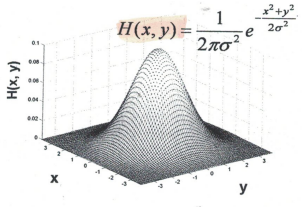



FIG. III.1 – Distribution Gaussienne 2D *avec un écart type* $\sigma = 1.25$ *et une moyenne* $\mu = 0$

Formule Gaussienne 2D : $\boldsymbol{h(x,y)} = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$

Formule de l'écart type : $\boldsymbol{\sigma = 0.3\left(\frac{n}{2} - 1\right) + 0.8}$ avec $n=n_x = n_y$

$$\boldsymbol{h_{3x3}} = \frac{1}{16}\begin{pmatrix} \mathbf{1} & \mathbf{2} & \mathbf{1} \\ \mathbf{2} & \mathbf{4} & \mathbf{2} \\ \mathbf{1} & \mathbf{2} & \mathbf{1} \end{pmatrix} \quad \Rightarrow \quad I_{gaussien} = \begin{pmatrix} 70 & 50 & 30 \\ 40 & \mathbf{42}^* & 70 \\ 60 & 50 & 10 \end{pmatrix}$$

Rappel sur Matlab:

1) Les fonctions utiles:

***size***(), ***abs***() pour calculer la valeur absolue, ***sum***(), *sum(sum())*, ***zeros***(),

opération matricielle terme par terme: **.*** (point étoile pour la multiplication terme par terme), ./ (point barre oblique), ***imnoise***() pour ajouter des bruits, ***fspecial***(), etc..

2) Attention à **l'effet de bord** : on peut ajouter les zéros autour de l'image avant d'effectuer les opérations.

3) Les niveaux de gris d'images sont entre [0,255] ou entre [0,1] avec noir (0) et blanc (255 ou 1).

4) Les opérations dans les images sont réalisées souvent avec des données de type « double » (exemple : log). S'assurer que les opérations s'effectuent sur une image I de ce type (en utilisant *double(I)* par exemple).

5) Changement de type des valeurs d'une image :
***uint8( ), uint16( )*** : entiers non signés 8, 16 bits
***double( )*** : double précision
***logical( )*** : binaire

**Programmation : Suppression de bruits par les filtres médian et moyenneurs**

**1)** Débruitage par filtre médian de l'image suivante déjà traitée dans la séance d'Histogramme :

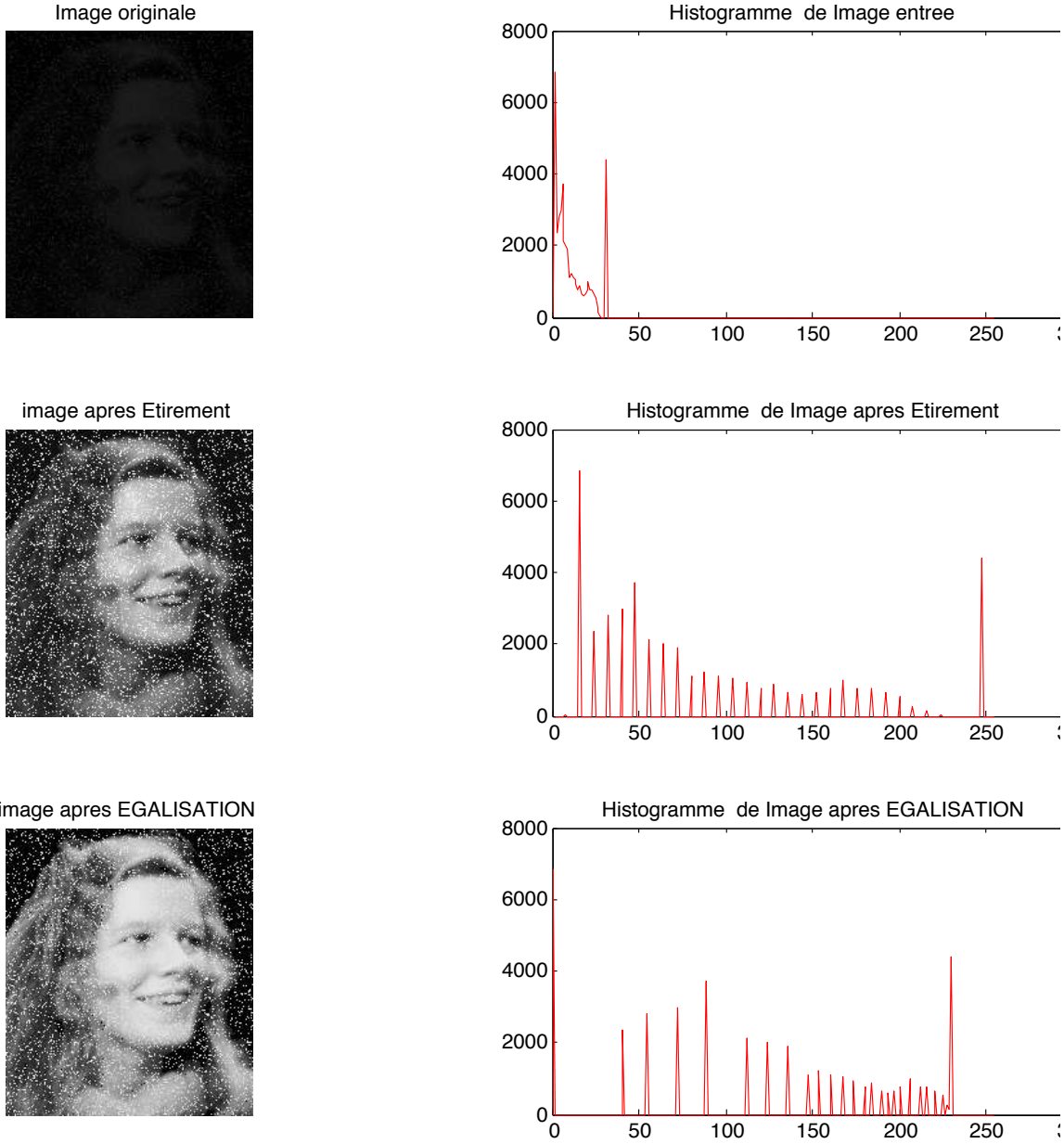



Fig.1 Images originale et après l'étirement et l'égalisation de l'histogramme

Fig. 2 Image débruitée

**2)** Filtres médian et moyenneurs

1) Lire le fichier d'image (l'image de test pour cette séance est 'lena.gif', ou 'cameraman.tif').

2) Ajouter à l'image originale le bruit de type gaussien de moyenne nulle et de variance 0.04 (4% de la valeur max), et de type 'salt & pepper'. (fonction ***imnoise****(I, 'gaussian/ salt & pepper', mean, var)* ). Afficher les deux images ainsi bruitées.

3) Appliquer sur les deux images bruitées un filtre moyenneur rectangulaire de taille 3x3, afficher et commenter les résultats obtenus.

4) Appliquer sur les deux images bruitées le «filtre médian », afficher et commenter les résultats obtenus.

5) Comparer 3) et 4).

Exemple :

Image+bruit gaussien

Image+bruit gaussien+ Filtre Moyenneur

Image+bruit impulsionnel

Image+bruit impulsionnel+ Filtre Median

6) Commenter sur l'atténuation du bruit et la qualité (ou la détérioration des détails) en fonction de la largeur du filtre moyenneur, puis le choix de la largeur adaptée ou optimale.

Exemple :

Image originale

Rayon = 5

Image originale

Rayon = 10