FILTRAGE SYNTAXIQUE, LINEAIRE ET CONVOLUTION
(BAS-NIVEAU)

Toujours dupliquer une image avant de la traiter (**Image→** *duplicate*). Penser aussi parfois à **Edit→***Undo* (lequel fonctionne *au plus* une fois). Toujours essayer de visualiser l'aspect linéaire du traitement en fonction de l'espace de représentation.

Quelques notes concernant ImageJ :

Le logiciel ImageJ provient de la réécriture dans le langage Java du logiciel libre de traitement d'images NIH-Scion Image (National Institute of Health, USA). Ainsi multi-plateforme, cette nouvelle version reste complètement libre, y compris son code source : de nombreux plugins ont été développés par des utilisateurs et sont disponibles via internet. Toutes les informations pour télécharger et utiliser le logiciel ImageJ se trouvent sur : http://rsb.info.nih.gov/ij/ , http://rsb.info.nih.gov/ij/docs/index.html , http://rsb.info.nih.gov/ij/docs/menus/plugins.html , http://rsb.info.nih.gov/ij/plugins/index.html . Une version où les aspects visualisation 3D et traitement 3D sont plus naturellement intégrés a été récemment proposée sous le nom de Fiji: http://fiji.sc/ .

Avertissement :

Le logiciel travaille sur des images codées sur 8 bits et n'affiche que des résultats entiers positifs. De temps en temps, en fonction de la version installée, le menu a pu légèrement changer et l'implémentation un peu évolué (en mieux normalement). A vous d'être flexible et agile.

Remarque 1 : Par défaut, cette version du logiciel adopte la **convention** habituelle pour les images en niveaux de gris : les niveaux sombres représentent des valeurs faibles proches du niveau 0 (noir), et les pixels clairs des valeurs élevées proches du niveau 255 (blanc).
Par contre, pour les images et opérations binaires est gardée la convention originelle du logiciel, alors très inhabituelle (sauf pour les radiologues) : les objets sont identifiés en noir par des pixels à 0, et se détachent du fond qui lui est rempli en blanc de pixels à 255, au lieu de respectivement blanc à 255 et noir à 0 dans la convention classique. Pour se rapprocher de cette dernière (objets blancs, **mais** à 255 sur fond noir à 0), éventuellement changer dans **Edit→*Options*→*Colors*…** les options *Foreground* à *white* et *Background* à *black*, et dans **Process→*Binary*→*Options*…** cocher la case *Black Background*.

FORMATS D'IMAGES

ImageJ permet d'importer des fichiers images de formats différents que le format *tiff* usuel. Par exemple, le format *raw* - pas d'information d'en-tête dans le fichier, seulement les informations de pixels ligne par ligne. Ces fichiers sont facilement visualisables si on connaît la hauteur et la largeur de l'image. Des images tiff codées sur 16 bits doivent être importées, mais ImageJ lira les dimensions de l'image à partir de l'en-tête du fichier.

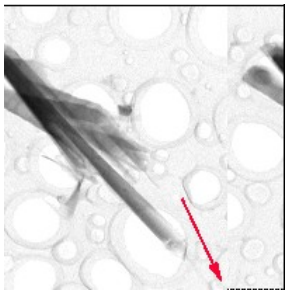
Lire un format TIFF (procédure normale)

- **File -> Open** menu, select **fib_tiff8.tiff.**
- L'image a été stockée avec une *LUT* inversée. La *LookUpTable* est une partie de l'en-tête *TIFF*. C'est une table de correspondance associant l'ensemble des niveaux de gris stockés dans le fichier à une autre dynamique de visualisation. **Image -> LookUpTable ->Invert. Testez les autres LUT.**

Importer un fichier RAW

Le fichier **fib_tiff8.raw** est la même image enregistrée au format Raw (brute)– cad sans aucune information d'en-tête. Elle ne peut pas être ouverte avec ImageJ (pas d'en-tête informant sur la largeur, hauteur...), mais elle peut être importée.

- **File -> Import** menu. Utiliser ces réglages: 199x202 – 8 bits
- Importer le fichier avec une largeur modifiée d'1 pixel. (Les images sont enregistrée en listant les pixels dans le sens de lecture usuel)



Si on met un offset de décalage de **40** bytes (ou octets) tout en conservant les bonnes dimensions de l'image 199x202, on montre l'effet de wrapping:

- Utiliser l'outil de sélection ligne pour mesurer la longueur de cette droite en pointillées – qui est longue de **40** pixels. Dans le cas de cette image, l'offset devrait être de 0.

Images TIFF codées sur 16 bits

Les valeurs de pixels sont alors compris entre 0 et 65535, au lieu de 0-255).

- En utilisant la pipette, vérifiez cela sur l'image **galaxie.tif.**
- Comment améliorer le contraste pour voir les détails (Menu **Process**).

ANALYSE STATISTIQUE D'UNE IMAGE

Modification de dynamique et égalisation d'histogramme:

- lire l'image **enhance_me.tif.** Calculer son histogramme (*Histogram* dans **Analyze**, ou raccourci clavier Ctrl+H). Y repérer en particulier la moyenne et la dynamique i.e. les niveaux min et max (*Mode* indique le niveau de gris le plus représenté dans l'image, avec entre parenthèses le nombre des pixels correspondants).
- effectuer une correction de la dynamique de visualisation de l'image en utilisant l'opérateur *Enhance Contrast* du menu **Process** : laisser les options *Normalize* et *Equalize Histogram* non cochée. Indiquer le pourcentage de points que l'on veut saturer : le logiciel calcule alors la dynamique de l'image (niveaux de gris min et max) et semble répartir les points non saturés équitablement entre 0 et 255.

Attention : si l'option *Equalize Histogram* aiguille sur une opération complètement différente (l'égalisation, cf. plus bas), *Normalize* permet l'application optionnelle du résultat. Ainsi, si *Normalize* n'est pas cochée, le réhaussement linéaire de contraste intervient seulement dans la visualisation (cf. **Image→*Color*→*Show LUT*** avant et après) tandis que l'image n'est pas changée numériquement : les histogrammes avant et après sont identiques. Si l'on désirait que le changement soit réellement pris en compte en dur dans l'image résultante (que la palette de niveaux de gris soit appliquée à l'image dans le fichier qui la stocke), il faudrait cocher l'option *Normalize*.

- Utiliser le menu **Image→*Adjust*→*Brigthness/Contrast***. Cet outil permet de visualiser la table de correspondance des niveaux de gris en superposition avec l'histogramme de l'image (mieux que *Show LUT* donc). Il offre de plus de multiples possibilités de modifier la LUT interactivement puisque sont observables les effets directs à la visualisation. Jouer sur la luminosité, le contraste et les niveaux extrêmes. Que se passe-t-il quand on **Apply** l'histogramme.
- Comparer le profil de surface correspondant entre l'image réhaussée et non réhaussée.
  Remarque : en somme, cocher l'option *Normalize* dans *Enhance Contrast* revient à cliquer sur *Apply* dans *Brigthness/Contrast*, ou encore à **Image→*Lookup Tables*→*Apply LUT***.
- Comparer les résultats obtenus et leurs histogrammes respectifs, après rehaussement de contraste affine *Enhance Contrast* (appliqué en cochant l'option *Normalize*), et après *Equalize Histogram*.
- Afficher l'histogramme et la LUT de l'image **chromosomeX.tif.**
- Appliquer à l'histogramme de l'image une transformation linéaire avec . A quelles conditions sur *a* et *b* rehausse-t-on son contraste ?

- Présenter à l'écran les images avant et après transformation ainsi que les histogrammes correspondants.
- Proposer une méthode automatique par seuillage d'histogramme pour mesurer la superficie totale occupée par les grains dans l'image **blobs.tif**.
- Enregistrez le plugin **Otsu_Thresholding** dans le répertoire de Plugins. Fermez imageJ et relancez-le. Appliquez successivement le seuillage automatique basique de **Process/Binary/** et celui plus intelligent de **Plugins/Otsu.**

DÉTECTION DE CONTOURS ET DERIVATION

La détection de contours est utilisée pour segmenter des images, déterminer et reconnaître des objets dans le cas où la texture n'a pas d'importance (en reconnaissance de formes par exemple). Dans une image, les contours correspondent à des fortes discontinuités (sauts de niveaux de gris) lorsque l'on suit une ligne tracée sur l'image.

- En raisonnant sur une telle ligne d'image présentant un contour (10 20 25 100 110 90 40 20 10, par exemple), réfléchir aux poids qu'il faudrait affecter à un masque 1x3 (1D), pour faire ressortir le contour (discontinuité) et affaiblir les zones homogènes. Remarquer que le sens de la discontinuité est pris en compte (passage clair → foncé ou foncé → clair).
- Réfléchir à la construction d'un masque 3x3 permettant de détecter les discontinuités horizontales, verticales, puis obliques (45°). On pourra tester les masques obtenus sur une image de détails **de clown.gif** avec le menu **Process/Filter/Convolve**
- En tenant compte du fait qu'ImageJ sature les valeurs négatives ou supérieures à 255, construire plusieurs masques 3x3, permettant d'extraire successivement les contours dans toutes les directions. Comment combiner ces images pour obtenir le résultat final : image de contours dans toutes les directions ?
- Appliquer ce traitement sur les images **boat.tif.**

On peut remarquer que de nombreuses discontinuités sont obtenues alors que certaines ne sont pas significatives (ce ne sont pas de vrais contours).

- Quelle opération effectuer ensuite afin de ne garder que les « vrais contours » de l'image ? Proposez des filtrages fréquentiels pour extraire les contours.

A travers la commande **Process**→*Find Edges*, ImageJ met en œuvre un détecteur classique donnant directement les contours dans toutes les directions : le détecteur de Sobel. Tout comme le détecteur de Prewitt (que l'on peut construire de la même façon par application de deux noyaux de convolution directionnels puis calcul de la racine carrée de la somme des carrés des 2 images résultantes), il forme une première étape dans la détection finale des contours.

- Ajouter du bruit sur l'image **boat.tif.**, dupliquer 2 fois l'image bruitée obtenue et appliquer un détecteur de contours (Prewitt ou Sobel). Comparer ce résultat à ceux obtenus après un prétraitement de l'image bruitée par un filtre de lissage 3x3, respectivement par un filtre médian 3x3.

RESTAURER EN UTILISANT LA FFT (Déconvolution)

Source : http://www-sop.inria.fr/ariana/personnel/Nicolas.Dey/DeMiTri/arc.php Les méthodes de déconvolution en imagerie microscopique requièrent en général une connaissance précise de la fonction d'appareil (PSF) qui caractérise le microscope dans divers contextes expérimentaux et qui définit les dégradations introduites par les différents éléments de l'ensemble système optique / échantillon. Plusieurs approches sont possibles pour obtenir cette fonction. Dans une première approche, de type expérimental, la PSF de l'instrument est caractérisée grâce à la visualisation de préparations de microbilles de latex calibrées enrobées des chromophores utilisés dans les études, qui sont chacune considérée comme un point source et dont l'image permet d'obtenir la PSF. Une deuxième approche consiste à calculer une PSF théorique à partir d'un modèle mathématique qui représente au mieux l'ensemble système optique / échantillon. Une troisième approche consiste à réaliser la déconvolution à l'aveugle, c'est à dire à partir des données elles-mêmes et sans besoin de donner en entrée la PSF. Le grand avantage de ce dernier type de méthode est de calculer non seulement les images restaurées mais aussi la PSF qui correspond à l'expérience. Les algorithmes actuellement les plus répandus sont la déconvolution haute résolution par un algorithme de type aveugle par critère MLE (``Maximum Likelihood Estimation'').

Le problème de la déconvolution d'images est inhérent à toute acquisition d'images réelles, qui sont dégradées par l'optique et l'électronique utilisées pour leur acquisition. Ce problème est de modélisation simple : $Y = h * X + n$, où Y est l'image acquise, X est l'image idéale que l'on cherche, qui est convoluée avec la fonction de floue h ou PSF et n est un bruit additif supposé blanc et gaussien. Nous considérons le problème en deux temps : d'abord le problème de l'estimation des dégradations (fonction de flou h et statistiques du bruit n), puis la déconvolution (trouver X, connaissant Y, h et les statistiques du bruit n). Les études antérieures menées depuis plusieurs années dans le projet Ariana en imagerie satellitaire ont montré qu'il était nécessaire de traiter ces deux estimations séparément, les modèles d'images (a priori injectés) les plus efficaces pour chacun des deux problèmes étant de nature différente. Pour l'estimation de la fonction de flou, il faut utiliser un modèle robuste d'images, générique à toutes les images, utilisant donc des caractéristiques universelles des images comme l'autosimilarité à travers les échelles. Pour la restauration, il faut au contraire un modèle plus fin, adapté aux détails de l'image considérée (contours, textures). Il n'est possible d'estimer les paramètres d'un tel modèle à partir de l'image floue et bruitée que si l'on connait les dégradations au préalable.

Concernant la restauration d'images connaissant les dégradations, il s'agit d'estimer X connaissant Y, h, σ où σ est l'écart type du bruit n. La reconstruction d'une image à partir d'une observation floue et bruitée est un problème mal posé, quel que soit le type d'acquisition (imagerie satellitaire ou biologique). Une simple inversion entraîne une amplification inacceptable du bruit. C'est pourquoi le problème est souvent régularisé lors de l'inversion, en imposant une contrainte de lissage à la solution recherchée.

Restauration d'image satellitaire

- Etudier le fichier exemple *FFT_deconv.pdf.*
- Sélectionner l'image **face.tiff**
- Récupérer la Transformée de Fourier : **Process -> FFT -> FFT** puis faire la même chose avec l'image **blur_circle.tiff**
- Revenir à l'image **face.tiff** originale et dégradez-la avec l'outil **Smooth**. Observez la FFT successivement.
- De même, faites subir des décalages et des rotations à votre image, et observez à chaque fois la FFT. Interprétez mathématiquement à chaque fois.
- Dégrader par le flou **face.tiff** avec **blur_circle.tiff**, en convoluant les deux images (FFT->FD Math)

- Comment atténuer l'effet de flou de l'image dégradée **faceX.tiff** si on la considérait comme une image microscopique nécessitant une opération de déconvolution dans le domaine fréquentiel (voir texte introductif). Faites des essais.
- Demander au professeur la fonction de « *blur* » correspondant à la « spread *function* » du microscope utilisé pour réaliser la bonne déconvolution.

FILTRAGE DE FORME - NON LINEAIRE – DE RANG (« HAUT NIVEAU »)

LES FILTRES MEDIANS

- Lire l'image **casserole**. Comparer les filtrages de type médian et de type moyenne (**Process→*Filters***).
- Lire l'image **NoisyMultBlobs.** Cette image contient un bruit impulsionnel de type *Salt and Pepper*. Comparer les résultats obtenus par filtrage de cette image bruitée en utilisant un filtre médian de taille 3 et un filtre de lissage linéaire de même taille. Expliquez la non linéarité du filtre médian. Même chose avec l'image **NoisyAddBlobs** souffrant d'un bruit additif gaussien.

MORPHOLOGIE MATHEMATIQUE

Nous allons travaillez sur des images binaires. Les objets sont normalement noirs sur fond blanc contrairement à la norme mathématique (voir remarque 1).

Les opérateurs basiques en noir et blanc

- A partir des opérations de bases disponibles dans le logiciel et des images binaires proposées (comme **Bin_cat.tif**), calculez les opérations morphologiques suivantes : **Erosion, Dilatation, Ouverture, Fermeture, Top-hat**
- Comment obtiendriez-vous un gradient morphologique, c'est-à-dire des contours de l'image par combinaisons d'opérateurs morphologiques.
- Comment vérifier expérimentalement la relation de dualité des opérateurs (e.g. Erosion/Dilatation, Ouverture/Fermeture) ?
- En quoi sont-ce des opérateurs de filtrage non linéaire ?

En niveaux de gris : installer le plugin morphologie.zip

- Mêmes questions sur l'image **model.tif.**
- Comment récupérer la position des yeux dans **model_face.tif :** il faut utiliser le concept de fermeture pour atténuer le blanc des yeux puis soustraire les images à la manière d'un top-hat morphologique puis utiliser le seuillage interactif.
- Sur **LineGraph**, essayez de ne garder que le texte. Combinez une ouverture, une dilatation, l'inversion, les opérateurs logiques binaires en les images.
- Testez le Watershed sur **Bin_coffee.tif.** On pourra alors utiliser *Analyse_Particles* pour compter les cellules ou grains de cafés et obtenir des statistiques de surface par exemple.

Le langage des cygnes

Nous allons travailler sur l'image **cygnes.tif.**

- Amusez vous à dilater ou éroder en *Greyscale* les cygnes. Puis, après avoir binarisé l'image, à partir des opérateurs sur les Histogramme, Binaire, Morphologie, trouvez une suite d'opérateurs pour afficher à l'écran le nombre de cygnes volant dans le ciel.
- Utilisez le concept de reconstruction géodésique (*GreyScale_Reconstruct*) pour créer une image à seulement trois cygnes. Nous vous rappelons que vous aurez besoin d'une image de marqueurs initiaux en plus de l'image originale, qu'on peut obtenir par l'utilisation répétée d'érosions.

FAIRE ÉVOLUER LE LOGICIEL

Etudier la documentation pour utiliser et écrire des plugins (http://mtd.fh-hagenberg.at/depot/imaging/imagej/). La partie essentielle vous est proposée dans le document *Plugins_Menu.pdf*.

Q1. Sur ***DotBlots***, il s'agit de trouver les cercles de rayon 10 pixels. Pour cela, trouvez le plugin lié à la transformée de Hough sur l'internet (voir doc d'accompagnement). L'installer dans votre logiciel. L'utiliser : attention il faut prétraiter l'image avant de lancer le plugin (binarisation, nettoyage de forme, lissage, détection de contour, squelettisation).

Q2. Transformez la suite d'opérations de la question 1 en une Macro appelée ***Cercle10*** applicable à n'importe quelle image en niveaux de gris qui permettra d'automatiser l'extraction de cercles de rayon 10 pixels. Sauvez le fichier texte correspondant à la macro et convertissez également cette macro en un plugin codé sous la forme d'un fichier .java.

Compilez le plugin .java pour en faire un executable .class.

Appliquez cette suite de commande à la composante *Cr* de ***Mandrill (obtenue avec le plugin ColorSpace.zip)*** en tant que macro puis plugin.

Q3. Editez le code ***Hough_Circles.java*** pour qu'il fasse toujours une recherche limité en quantité à 10 cercles de rayons 10 pixels. Compilez-le pour créer le ***Hough_Circles.class*** correspondant et donc le nouveau plugin. Appliquez la macro précédente.

!!Remarque : si le logiciel ne trouve pas de compilateur Java, il faut rajouter l'archive tools.jar en ligne de commande (voir la doc d'installation : *java –cp ij.jar ;XXX/tools.jar ij.imageJ*).