Traitement d'Images Numériques :une introduction avec le logiciel ImageJ et le langage Java

Tout le matériel nécessaire se trouve soit à l'URL http://www.math-info.univ-paris5.fr/~lomn/Cours/CV/TI/Material/Tutorial/ ou /Material/Data (site web 1) soit à l'URL http://rsb.info.nih.gov/ij/ (site web 2). Nous travaillerons sous Linux même si le logiciel fonctionne identiquement sous Windows et évidemment MacOS. Nous utiliserons le logiciel imageJ développé par le NIH. Ce logiciel est un petit laboratoire portable pour tester, expérimenter et même coder en Java. La version maintenue actuelle et plus complète s'appelle FiJi but « Fiji is just imageJ ». Il y a la distribution de base et ensuite on peut ajouter des plugins dans le répertoire plugins sous la forme de classe java en pseudo-code sous la forme de fiichiers avec l'extension *.class*. Quand on lance imageJ il charge automatiquement l'ensemble des plugins *.class* présent dans ce répertoire.

Prendre le logiciel imageJ en main

D'abord, il faut l'installer. L'archive *ijXXX.zip* fournie sur le site web 1 peut suffire. La télécharger. La dézipper (commande *unzip* ou *gunzip* sous Linux). Pour lancer le logiciel, se mettre dans le répertoire créé où se trouve l'archive *.jar* puis lancer la commande

$java –jar ij.jar

L'interface graphique est assez intuitive. Consultez rapidement :

- http://rsb.info.nih.gov/ij/docs/pdfs/ImageJ.pdf
- http://rsb.info.nih.gov/ij/docs/pdfs/examples.pdf etc. l'ensemble des pdfs est intéressant d'ailleurs.

Cette installation suppose qu'un JRE (Java RunTime Environment est installé sur votre machine). Il vient généralement par défaut avec les systèmes d'exploitations. Toutefois, si votre machine n'en possède pas se référer à http://rsb.info.nih.gov/ij/docs/install/linux.html et à la version plus complète qui contient un JRE (http://rsb.info.nih.gov/ij/download/linux/ij148-linux64.zip). Pour les futurs développeurs, un environnement tel que Java Development Kit (JDK) est conseillé (http://www.oracle.com/technetwork/java/javase/downloads/index.html). Cet environnement fournira notamment la librairie permettant (facilement) de compiler des plugins imageJ en Java pour les modifier. Nous verrons cela par la suite.

A noter que nous travaillons avec imageJ mais la version améliorée du logicielle (actuellement maintenue) et plus dédié encore aux Sciences de la Vie (visualisation 3D etc.) s'appel *Fiji*. Mais … *Fiji is just ImageJ*.

Ici vous trouverez une FAQ http://fiji.sc/Frequently_Asked_Questions qui peut s'avérer utile. En résumé pour travaillé avec imageJ, il faut a minima un JRE. Avec un JDK on peut faire du développement (coder, compiler) et donc faire évoluer le logiciel assez facilement.

Astuce : pour vérifier la présence de jre, jdk et leur localisation sur votre système utiliser les commandes Unix : soit ***$locate *jre**** soit ***$find /usr -name *jre**** par exemple et à adapter.

On va essentiellement travailler sur les images *Samples* proposés par imageJ en ligne. Il faut donc *a priori* que le réseau internet fonctionne correctement. Sinon, certaines essentielles seront accessibles sur le site 1.

Q0. Sur l'image ***blobs.gif***, comptez le nombre approximatif de cellules. Pour cela :

- Binarisez
  → ***Process/Binary/Make binary*** ou ***Image/Adjust/threshold*** (interactif). Testez les deux
  puis utilisez ***Analyze/Analyze Particles*.** (jouez un long moment avec les options etc.)

Q1. Ensuite, tentez localement de séparer deux cellules qui se touchent pour affiner le nombre de particules (utilisez pour cela les ***ROI : Region Of Interest*** qui permettent de n'appliquer des traitements en l'occurrence ici ***l'érosion Process/Binary/Erode*** que sur une partie de l'image définie avec la souris et plusieurs fois). Et recomptez automatiquement les cellules avec ***Analyze Particles***

Q2. Refaites les démarches en lissant auparavant l'image. (***Process/Smooth***)

Q3. Sur ***Boats.gif***, trouvez les contours de l'image. (***Process/Find Edges***)

Q4. Travaillez sur l'image ***EnhanceM***e et essayez d'y voir quelque chose. (***Process/Enhance_Contrast).***

Q5. Sur ***LineGraph***, essayez de ne garder que le texte.

Binarisez votre image (Process/Binary/Make Binary), Duppliquez-la (Image/Duplicate). Réalisez une ouverture sur la première image (Image/Binary/open), suivi d'une dilatation (Image/Binary/dilate), utilisez un filtre AND (Et logique) entre les deux images (Process/Image_Calculator). Là c'est de la morphologie mathématique puissante qui travaille sur la structure et la forme. Historiquement, imageJ considère la valeur noire comme la valeur maximale (255 en niveau de gris ou 1 en binaire) car le fond est en général blanc dans nos images habituels. Ce qui peut être une source de confusion pour le filtre ET logique. Notez que la valeur de l'intensité sous la souris est indiqué dans la fenêtre principale de imageJ.

Faire évoluer le logiciel : compiler et écrire du code Java

Etudier la documentation pour utiliser et écrire des plugins (http://fiji.sc/Introduction_into_Developing_Plugins). La partie essentielle vous est proposée dans le document *Plugins_Menu.pdf* sur le site 1.

Q1. Sur ***Dot-Blots.jpg***, il s'agit de trouver les cercles de rayon entre 10 et 20 pixels. Pour cela, trouvez le *plugin* lié à la transformée de Hough sur l'internet (développé par M. Hemerson Pistori) (voir doc d'accompagnement *hough_circle.html*: il y a des opérateurs comme détecteurs de contours à effectuer pour un résultat optimal ou sur mon site 1). L'installer dans votre logiciel c'est-à-dire télécharger le code dans le répertoire *plugins* de votre installation. L'utiliser.

Q2. Transformez la suite d'opérations de la question 1 en une Macro appelée ***Cercle20*** applicable à n'importe quelle image en niveaux de gris qui permettra d'automatiser l'extraction de cercles de rayon 20 pixels. Appliquez-la à la composante *Cr* de ***Mandrill*** (la composante *Cr* d'une image couleur est le canal dit de Chrominance rouge et peut être récupérer avec les plugins dans ***ColorSpaces.zip***).

Q3. Éditez le code ***Hough_Circles.java*** pour qu'il fasse toujours une recherche non limité en quantité de cercles de rayons 20 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) lors de l'exécution de imageJ.

$java –cp ij.jar:XXX/tools.jar ij.imageJ

où XXX est le chemin URL sur votre machine de l'archive tools.jar qui contient le compilateur java (javac).
De plus, en général, si vous créez un plugin à partir de rien, insérer dans le nom du fichier un « _ » permet au logiciel imageJ de charger automatiquement le nouveau plugin lors du lancement du logiciel.

Enfin vous remarquerez que

$javac Hough_Circle.java

ne fonctionnera pas car ce plugin fait appel à des librairies typiques de imageJ pour fonctionner (les import au début du fichier de code). Il faut donc souvent préciser le chemin d'accès à ces librairies si on veut compiler en dehors de imageJ des codes de plugins. Par exemple,

$javac -cp XXX/ij.jar Hough_Circle.java
pourrait fonctionner (cp : correspond à ClassPath,le chemin d'accès des archives jar nécessaires à la compilation).

ou
$javac -cp .:XXX/ij.jar Hough_Circle.java
pour ajouter le répertoire courant.

Une formule intéressante pour faire et de l'Unix et de la compilation :
$javac -cp .:`find * -name "*.jar" | tr "\n" ":"` Hough_Circle.java
Qu'est-ce que cela fait ?

A terme l'usage d'un IDE (Integrated Developement Environment) est conseillé comme Eclipse pour des projets Java. Il propose un environnement graphique de développement plutôt qu'en ligne de commande. Mais il demande un peu de temps à prendre en main.