Université de Bordeaux

Initiation à la Recherche
et/ou Développement

Cahier des charges

Thomas Faux
Charlotte Héricé
Typhaine Paysan-Lafosse
Joris Sansen

M. Jean-ChristopheTaveau

2011-2012

CBMN - UMR 5248
B8 AVENUE DES FACULTÉS
F-33402 TALENCE CEDEX

Table des matières

Chapitre 1

Contexte

1.1 Sujet

Le laboratoire de Chimie et Biologie des Membranes et Nanoobjets de bordeaux (CBMN [1]) est un laboratoire de recherche public composé de douze équipes de recherche dont l'équipe *Architecture des Complexes Membranaires et processus cellulaires* (ACMPC).

Cette équipe, dirigée par O. Lambert s'intéresse à l'architecture de complexes membranaires sur des structures de type protéine-liposome. C'est dans ce cadre d'étude que les chercheurs travaillent avec un *cryo-microscope électronique à transmission* (cryo-MET) afin d'obtenir des micrographes des structures étudiées puis de les analyser.
De plus, dans le cadre de leur partenariat avec une entreprise pharmaceutique, ils préparent des échantillons de virus pour observations au cryo-MET et analyse. Cependant la nouvelle génération des appareils d'observations permettent une collecte de données à haut débit, avantage non négligeable mais qui pose le problème du traitement des données collectées.
L'automatisation du traitement des données devient alors nécessaire, l'analyse manuelle des données récupérées étant extrêmement fastidieuse et chronophage.

Dans le cadre de leurs recherches et pour la partie qui nous concerne, l'analyse correspondra au traitement des images récupérées du cryo-MET et plus particulièrement au *picking* c'est-à-dire à le piquage des particules présentes sur les micrographes obtenus.

1.2 Objectifs

Notre objectif consistera donc en la création donc en l'implémentation d'une méthode de picking automatisé des particules sur les micrographes. Les échantillons qui nous serviront de tests seront de deux types :

- des micrographes d'échantillons de virus, de forme ronde, qu'il faut sélectionner afin de pouvoir déterminer leurs nombre et tailles,
- des micrographes de protéines membranaires, de forme pyramidale, qu'il faudra aussi sélectionner.

Ce procédé de piquage sera implémenté sous la forme d'un plugin ImageJ [2] qui servira de plate-forme de picking : il proposera à l'utilisateur d'utiliser des algorithmes de picking pré-installés ou d'en ajouter de nouveaux. Cela permettra de simplifier et de centraliser l'accès aux outils de sélection que l'on peut implémenter sur ImageJ.

Le logiciel ImageJ a été choisis car c'est un logiciel de traitement et d'analyse d'images développé en Java ™ [1] par le Nation Institute of Health (NIH).
Ce logiciel est "open-source" [2], multi-plateforme et bien connu de la communauté scientifique car initialement conçu pour des applications biomédicales. Il s'est peu à peu démocratisé dans d'autres domaines pour sa facilité d'utilisation et les possibilités de développement qu'il offre.
En effet, il est possible de développer soit-même et assez facilement des plugins, que ce soit en Java ™ ou en JavaScript [3].

1. Java ™ est un langage orienté-objet développé par Oracle [3]
2. son code source est en accès libre et est modifiable(licence GNU-GPL)
3. JavaScriptest un langage de programmation de script orienté objet à prototype [4]

Chapitre 2

État de l'existant

2.1 Détection à l'aide de références

Les approches initiales concernant la piquage de particules furent basées sur des références faisant appel à un modèle. Cette technique est utilisée pour trouver de petites parties d'une image en la comparant à un modèle. Celui-ci peut être obtenu soit à partir d'une structure tridimensionnelle connue si elle est utilisable soit par sélection d'une particule servant d'exemple dans le micrographe. L'algorithme détermine la meilleure correspondance entre la cible et le modèle pour pouvoir le localiser dans l'image.
La détéction par un modèle de référence est aujourd'hui plus performante, elle a été améliorée en tenant compte des changements d'efficacité de la variance locale dans l'espace de Fourier et de nouvelles approches qui utilisent différentes méthodes pour incorporer des modèles de bruit générique, ceci afin de pouvoir mieux gérer le bruit.

2.2 Piquage de particules sans références

2.2.1 Détection des bords et transformée de Hough

Les difficultées majeures rencontrées dans les techniques basées sur la détection des contours sont dues à la compléxité de détecter les bords des particules lorsqu'il y a un bruit de fond important sur les images issues de la Microscopie à Transmission Electronique.
Cette technique est basée sur la détection des arêtes ainsi que sur l'application de la transformée de Hough [5]. Cette méthode permet de détecter la présence de formes comme des lignes, des cercles ou encore des ellipses.
Dans la transformée de Hough [6], chaque ligne correspond à un vecteur à deux paramètres : Θ (angle) et ρ (norme du vecteur). Cette technique est basée sur la transformation de toutes les lignes possibles qui passent par un point, c'est-à-dire en calculant la valeur de ρ pour chaque Θ, on obtient alors une sinusoïde unique appelée espace de Hough. Si les courbes associées à deux points se coupent, l'endroit où elles se coupent dans l'espace de Hough correspond aux paramètres d'une droite qui relie ces deux points.

La détection de formes s'apparentant à des cercles ou des axes circulaires implique la localisation de tous les centres possibles et de trouver le rayon pour chaque centre ciblé.
Si le rayon du cercle est déjà connu, les coordonnées probables du centre du cercle sont d'abord stockées dans un fichier. Elles sont ensuite recherchées dans l'image afin de déterminer les points indiquant la localisation du centre des particules circulaires.
Inversement, si le rayon des particules n'est pas connu, un seul paramètre en plus est nécessaire. Il suffit d'enregistrer, non pas un seul point pour chaque pixel de contour mais une ligne constituée de plusieurs points le long du contour de la particule dans ce plan.

D'autre part, pour détecter des particules de formes irrégulières, l'approche décrite précédemment peut également être utilisée mais la transformée de Hough devra alors être remplacée par la transformée de Hough Généralisée [7]. Cette nouvelle méthode repose sur la modification de la transformée de Hough en utilisant le principe de l'identification à partir d'un modèle de référence. Cette modification permet également d'utiliser la transformée de Hough pour la détection d'objets non caractérisés à l'aide d'une équation analytique.
Tout d'abord, la méthode requiert la sélection de deux paramètres : la localisation du point à l'aide d'un modèle de forme. Ensuite, une distance adéquat est bougée dans différentes directions ρ afin d'arriver au point L. Pour détecter les particules orientées différemment ou qui ne seraient pas à la même échelle par rapport à une même forme, l'ajout de deux paramètres est nécessaire : l'orientation et l'echelle.

2.2.2 DoGLFC et classement par affinité

Cette technique est basée sur l'utilisation de l'algorithme DoGLFC [1] complété par l'algorithme de classement par affinité [8].

Le DoGLFC est basé sur l'algorithme DoG Picker du Scripps Institute [9], une méthode rapide qui permet la segmentation de particules. Après l'application de l'algorithme de Différence de Gauss (DoG), on obtient une cartographie de points similaire à celle de la méthode utilisant un modèle de référence.
Cet algorithme requiert un paramètre ajustable unique ou un jeu de paramètres basé sur le rayon de la particule ou un ordre de grandeur du rayon. L'exécution de cet algorithme renvoie une liste de trois paramètres décrivant la localisation de la particule (les coordonnées x et y) et la taille du pic.
L'algorithme DoGLFC sélectionne les particules (ou objets) potentielles de taille déterminée, ceci a un pouvoir discriminatif moindre par rapport à une technique basée sur une référence, tel que l'utilisation d'un modèle de référence, mais elle ne nécessite pas d'avoir beaucoup d'informations pour que la référence soit utilisable comme modèle.

Pour améliorer le rendement lors du piquage des particules, un nouvel algorithme semi-supervisé, le classement par affinité, peut être appliqué.
L'algorithme a besoin de trois paramètres d'entrée : un jeu d'images, la taille maximale de chaque vecteur et deux jeux d'indices indiquant quelle fenêtre doit être utilisée comme référence positive ou négative.
Lorsque l'algorithme a fini de se dérouler, on obtient une note de classement de chaque fenêtre où ce dernier est le plus proche de celui de la fenêtre où se trouve la particule ciblée.

L'utilisation de DoGLFC complété par le classement par affinité permet d'extraire rapidement les particules de l'image et d'éliminer avec précision les particules correspondant à de la contamination ou du bruit de fond.

2.3 Perceptron

Un perceptron est une sorte de réseau neuronal artificiel. Dans son état le plus simple, il représente un système de classification binaire/linéaire.

1. Difference Of Gaussian Local Fast Correlation

Ce programme a la particularité d'être capable d'apprendre des concepts, ce qui signifie qu'il peut apprendre afin de répondre par vrai ou faux à des données qui lui sont soumises, grâce à la présentation répétée de plusieurs exemples d'étude. Il a déjà été testé sur des images binaires pour la détection de formes ou de contours mais pas sur des images en niveaux de gris ou sur des problèmes de reconnaissance de modèles visant à sélectionner des particules. Le réseau neuronal n'a pas non plus été exploité comme un outils de sélection automatique de particules mais plusieurs recherches ont conclu qu'il pourrait être utilisé pour l'élimination de faux-positifs. [10].

Chapitre 3

Analyse des besoins

3.1 Besoins fonctionnels

Le plugin

- Au lancement, les images seront préalablement chargées sur ImageJ, et l'utilisateur aura le choix entre plusieurs algorithmes de picking qui lui proposeront les différents traitements applicables afin d'optimiser le piquage,
- l'interface proposera un mode de prévisualisation afin de vérifier les changements apportés à l'image avant de les appliquer au stack,
- l'affichage sera clair et succinct, et sera constitué d'une liste de boutons, scrollbars, et autres paramètres modifiables (rayons de filtres, etc),
- les résultats seront affichés dans un tableau, et visualisables sur le stack afin de vérifier le bon déroulement du processus,
- l'utilisateur pourra modifier les sélections (ajouter et retirer les pointeurs),
- il devra également être possible d'implanter simplement de nouveaux algorithmes dans l'interface.

Les algorithmes

- Traitements et segmentation des micrographes issus de cryo-MET,
- piquage automatique des particules depuis les micrographes,
- le format de sortie est un jeu de coordonnées *x,y* associé à la position de l'image dans le stack, le tout enregistré dans un fichier au format *.csv*.

3.2 Besoins non fonctionnels

Le plugin

- L'implémentation de la plateforme sera faite en Java ™,
- nous utiliserons les API [1] graphiques de la bibliothèque *swing*,
- le programme devra être le plus simple d'utilisation possible afin de le rendre accessible à tous,
- le code devra être suffisamment clair, explicite et commenté afin de permettre aisément l'implémentation de nouveaux algorithmes.

1. Application Programming Interface

Les algorithmes

- Les algorithmes seront testés avec l'outil macro d' ImageJ et implémentés en Java ™,
- le temps de traitement des données devra être le plus rapide possible afin de pouvoir gérer de grands jeux de données,

Dans l'optique de démocratiser l'utilisation de notre interface et afin de permettre l'extension de ce plugin avec d'autres algorithmes, nous placerons notre projet sous une licence GPL [2].
Le projet devra être terminé vers mi-mai.

2. General Public License [11]

Chapitre 4

Choix et explications

Comme expliqué précédemment, afin de réaliser notre projet, plusieurs choix doivent être faits :

- tout d'abord le choix des langages utilisés,
- et ensuite la ou les méthodes de piquage que nous implémenterons.

4.1 Langages choisis

Nous avons fait le choix de commencer notre projet avec une phase de prototypage en utilisant un langage interne propre à ImageJ, que nous avons préféré au langage JavaScript. Cela nous permettra de faire rapidement différents tests afin d'éprouver la méthode utilisée (robustesse, fiabilité, efficacité).
La seconde phase consistera, si le code nous semble satisfaisant et suffisamment performant, en sa traduction en Java ™ qui est le seul langage utilisable pour développer un plugin pour ImageJ.

Le JavaScript, qui est un langage proche de Java ™, nous aurait permis de réaliser des essais de piquage et aurait facilité la création du plugin. Cependant, nous n'avons encore jamais travaillé avec celui-ci et son apprentissage serait trop chronophage.
Nous ne l'avons donc pas choisi pour l'implémentation du plugin et avons préféré nous orienter sur l'outil macro d'ImageJ.

En effet, ce logiciel possède un mode Recorder visant à conserver un historique de nos commandes. Cette fonction peut également servir lors de l'écriture de Macros ImageJ ainsi que pour l'automatisation des tâches. Grâce à celui-ci, la création de notre méthode de piquage pourrait se faire beaucoup plus rapidement. De plus, nous avons déjà utilisé cet outil au cours de notre formation mais il reste éloigné du Java ™ et du JavaScript.

Une fois l'algorithme de piquage testé et validé avec l'outil Macro d'ImageJ, nous allons l'adapter en langage Java ™ afin de pouvoir l'implanter dans notre plugin.
Le langage Java ™ est fréquemment utilisé, multiplateforme [1] grâce à l'utilisation d'une machine virtuelle et propose de nombreuses bibliothèques.
De plus, de nombreux logiciels proposent un mode de debogage pour ce langage, ce qui confère un avantage non négligeable étant donné le temps imparti.

1. compatible quel que soit le système d'exploitation

4.2 Algorithmes de piquage

Nous allons principalement nous orienter sur deux algorithmes de piquage :

- le premier est basé sur la détection des contours de particules par comparaison avec une image de base. Nous les comparerons donc à une image de cercle dont la taille variera afin de sélectionner les objets quelles que soient leurs tailles.
- le second comparera des particules avec une image de référence issu des images à analyser. Celle-ci sera orientée selon différents angles afin de détecter les particules d'intérêt quelle que soit leur orientation. Le modèle sera ensuite corrélé au stack d'images préparées et l'image résultante contiendra les pixels d'intensités maximales sur lesquels nous pourrons faire la sélection.

En supplément de ces deux algorithmes, nous en ajouterons sûrement deux autres : Détection par algorithme de Hough ainsi que l'algorithme de DoG [2].

Une fois ces algorithmes implantés dans l'interface, nous effectuerons quelques tests statistiques sur les résultats obtenus (positifs, négatifs, faux-positifs) afin d'éprouver leurs fiabilités.

2. Difference of Gaussian

Bibliographie

[1] Université de Bordeaux. Chimie et biologie des membranes et nanoobjets - umr 5248. `http://www.cbmn.u-bordeaux.fr`.

[2] National Institute of Health. National institute of health - imagej's website. `http://www.rsbweb.nih.gov/ij/`.

[3] Oracle. Java - object-oriented programming language. `http://www.oracle.com/technetwork/java/index.html`.

[4] Wikipedia. Javascript - a multi-paradigm language, supporting object-oriented, imperative, and functional programming styles. `http://en.wikipedia.org/wiki/Javascript`.

[5] Yuanxin Zhu, Bridget Carragher, Fabrice Mouche, and Clinton S. Potter. Automatic particle detection through efficient hough transforms. *IEEE Transactions on Medical Imaging*, Sept. 2003.

[6] Wikipedia. Definition and description of hough transform. `http://en.wikipedia.org/wiki/Hough_transform`.

[7] Wikipedia. Definition and description of generalized hough transform. `http://en.wikipedia.org/wiki/Generalised_Hough_transform`.

[8] Robert Langlois, Jesper Pallesen, and Joachim Frank. Reference-free particle selection enhanced with semi-supervised machine learning for cryo-electron microscopy. *Journal of Structural Biology*, Jun. 2011.

[9] Scripps Research Institute. Scripps research institute website. `http://www.scripps.edu/`.

[10] William V. Nicholson and Robert M. Glaeser. Review : Automatic particle detection in electron microscopy. *Journal of Structural Biology*, Feb. 2001.

[11] Inc. Free Software Foundation. The gnu general public license website. `http://www.gnu.org/copyleft/gpl.html`.

Projet Picking-Plugin