



# Analyse d'image scientifique le monde ImageJ

Daniel.Sage@epfl.ch, EPFL-STI, Biomedical Imaging Group (BIG), responsable des développements logiciels



*ImageJ is a Java-based image processing and image analysis software for scientific images. ImageJ is in the public domain. It has made an important impact in biomedical imaging, becoming the de facto reference. Its open architecture has enabled the creation of hundreds of plugins, which have sometimes integrated the most advanced image-processing techniques.*

ImageJ est un logiciel Java de traitement d'images et d'analyse d'images scientifiques qui est dans le domaine public. ImageJ a un impact important en imagerie biomédicale, devenant *de facto* une référence. Son architecture logicielle ouverte a permis la réalisation de centaines de plugins intégrant parfois les techniques de traitement d'images les plus avancées.

## Fiche descriptive

ImageJ /Fiji		
<b>Domaine</b>		
◆ Traitement d'images et analyse d'images		
<b>Licence</b>	<b>langue</b>	<b>version</b>
◆ ImageJ: domaine public	◆ anglais	◆ 1.46f
◆ Fiji: GPL v2		
◆ ImageJ2: pas encore défini		
<b>Autres alternatives libres</b>		
◆ CellProfiler		
◆ BioImageXD		
◆ ICY		
<b>Alternatives non libres</b>		
◆ Metamorph		
◆ Imaris		
<b>Sites Web</b>		
◆ ImageJ: <a href="http://rsb.info.nih.gov/ij/">rsb.info.nih.gov/ij/</a>		
◆ Fiji: <a href="http://fiji.sc">fiji.sc</a>		
◆ ImageJ2: <a href="http://developer.imagej.net/">developer.imagej.net/</a>		
<b>Plates-formes</b>		

ImageJ est un fascinant logiciel de traitement d'images et d'analyse d'image scientifique [1], autant par ses fonctions intrinsèques que par le riche ensemble de fonctions délivrées sous forme de *plugins*. Derrière une interface quelque peu désuète, ImageJ cache des fonctions simples d'accès, directes et puissantes pour l'analyse d'image. Ce logiciel s'utilise dans tous les domaines scientifiques, bien qu'un accent particulier soit mis sur l'imagerie biomédicale. Dans un esprit totalement altruiste, ImageJ a été déposé dans le domaine public. ImageJ possède un mécanisme simple pour la création de *plugins*. Des centaines de traiteurs d'images, de programmeurs, de chercheurs avertis et d'utilisateurs un peu débrouillards ont réalisé des *plugins* et ont mis leurs contributions en ligne.

## Que fait ImageJ ...

Passer en revue toutes les fonctionnalités déborderait largement du cadre de cet article. En effet, ImageJ dispose d'un catalogue complet d'opérations standards de traitement des images. Si jamais une opération venait à manquer, alors on trouvera bien quelqu'un qui l'aura programmée et mise à disposition.

ImageJ est capable de lire un grand nombre de formats d'images [2], soit par ses fonctions natives (y compris le format médical Dicom), soit en utilisant un *plugin* spécialisé comme la très complète librairie BioFormats (licence GPL) [3]. Il y a même des *reverse engineering plugins* qui permettent de décoder certains formats de fichiers propriétaires, tenus secrets pour des raisons mercantiles.

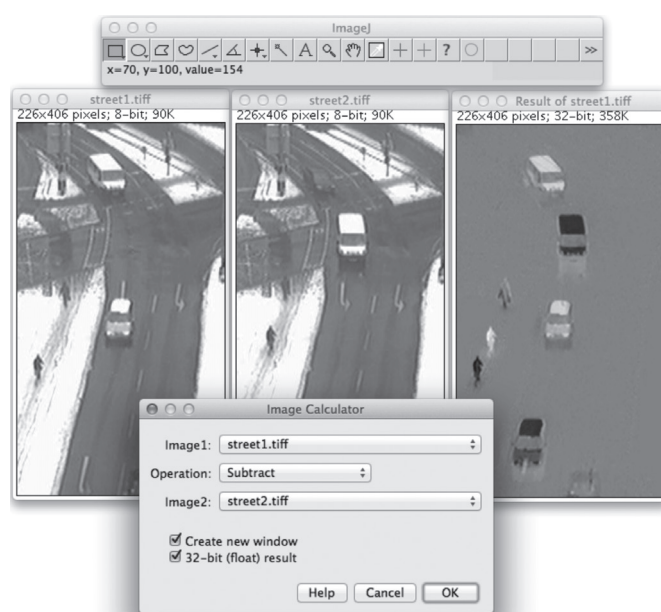


fig. 1 – illustration de la fonction Image Calculator. L'image résultat (à droite) est la soustraction des deux images (à gauche et au centre) prises à quelques secondes d'intervalle. Cette soustraction met clairement en évidence les objets en mouvement.

ImageJ possède une large panoplie d'opérations usuelles de traitement d'images dites *pointwise* et de transformation d'histogramme. Dans ce cadre, une fonction **Image Calculator** qui au premier abord semble banale s'avère en pratique très utile et intuitive. Elle permet de combiner des images pixel-à-pixel avec des opérations, exactement comme on pourrait le faire avec des nombres (fig. 1).

Les opérations de filtres dans le domaine spatial font partie de ImageJ, aussi bien les filtres linéaires classiques que les filtres morphologiques. Pour la segmentation d'images, la fonction préférée des utilisateurs est **Analyze Particles** qui, après binarisation, va extraire tous les composants connexes (objet) d'une image et caractériser leurs formes et leurs dimensions.

Un avantage de ImageJ est la possibilité d'appliquer certains traitements dans une troisième dimension. Celle-ci peut indifféremment représenter le temps pour les séquences d'images (films), ou une dimension spatiale supplémentaire (*z-stack*) pour les données volumétriques qui sont courantes en imagerie biomédicale.

### ... et ce que ImageJ ne fait pas

Dans le monde des logiciels d'imagerie numérique, les genres sont souvent confondus et il n'est pas inutile de mentionner ce que ne fait pas ImageJ.

- Acquérir des images, excepté pour les caméras compatibles TWAIN, FireWire.
- Visualiser des images en 3D: bien que ImageJ puisse tirer parti de certains *plugins* de rendu 3D, ils demeurent néanmoins moins efficaces et moins interactifs que les logiciels spécialisés.
- Retoucher une photo ou un montage. Photoshop ou GIMP sont meilleurs pour cette tâche.
- Produire une analyse statistique. Typiquement, dans une chaîne complète d'analyse d'images, les résultats bruts sont extraits par ImageJ et stockés sur fichiers. Ils sont ensuite repris par un logiciel d'analyse de données comme MS Excel/Open Office Calc, Matlab/Octave ou R.
- Résoudre directement votre problème d'analyse automatique d'images. Il est plutôt rare qu'une tâche d'analyse d'images entre simplement dans une catégorie standard. Il faut alors recourir au développement d'un programme spécialisé par macro ou par *plugin* et c'est là qu'intervient la programmabilité de ImageJ.

### Développement atypique

Jusqu'à aujourd'hui, le développement de ImageJ a reposé sur les épaules d'un seul homme, Wayne Rasband du *National Institutes of Health*. En 1997, Wayne Rasband avait déjà un logiciel d'analyse d'images très populaire mais tournant uniquement sur Macintosh: NIH Image. Il fait alors le pari —risqué à l'époque— de le porter en Java. Ceci deviendra ImageJ. Grâce à Java, ImageJ devient naturellement un logiciel multi-plateforme, ce qui va en décupler les usages.

Dès sa conception, ImageJ a été construit avec une architecture logicielle capable de recevoir des *plugins*, un concept assez novateur à l'époque. Le succès ne s'est pas fait attendre. De nombreux développeurs se sont attelés à la création de *plugins* souvent

pour leur propre usage et habituellement, ils les mettent à disposition gratuitement. Toutefois, en l'absence de coordination et de contrôle de qualité, on trouve de tout dans le monde ImageJ: des *plugins* au sommet de l'état de l'art dans leur domaine aux *plugins* de quelques lignes non documentés souffrant parfois de bugs. D'ailleurs, le principal souci de l'utilisateur est de se retrouver dans la jungle des *plugins*. Tout de même, il convient de souligner la courageuse tentative de documentation entreprise par le portail ImageJ [4].

Ce développement communautaire non hiérarchisé, a tout de même permis à des centaines de *plugins* de devenir très populaires, à tel point que l'on peut considérer ImageJ comme une coquille réceptacle de *plugins*. Certains d'entre eux sont si évolués qu'ils sont plus grands que ImageJ lui-même, à l'instar de TrakEM2 [5].

### J comme Java

Vous l'aurez compris, l'étrange J de ImageJ provient du langage de programmation Java dans lequel ImageJ est écrit. En traitement d'images, un éternel débat oppose les partisans de C/C++ à ceux de Java. En pratique, Java se présente comme un excellent langage pour l'imagerie, rapide, régulier dans sa structure, capable de manipulations graphiques, même s'il présente quelques défauts dans sa gestion de mémoire. Java est riche en bibliothèques de tous genres; grâce à sa facilité d'apprentissage, il satisfait autant le programmeur chevronné qu'une personne qui ne serait pas un pur programmeur.

Malgré les évolutions permanentes de Java, ImageJ est toujours compatible avec les toutes premières versions du langage. Cette volonté de stabilité explique l'interface dépouillée d'ImageJ.

Quelques passerelles existent entre ImageJ/Java et d'autres environnements: vers MySQL, vers Javascript, vers Python, vers ITK, vers R avec Bio7 [6], vers Matlab avec MIJ [7] et vers des produits commerciaux comme le logiciel de visualisation Imaris de Bitplane/Andor.

### Programmation: macros ou plugins

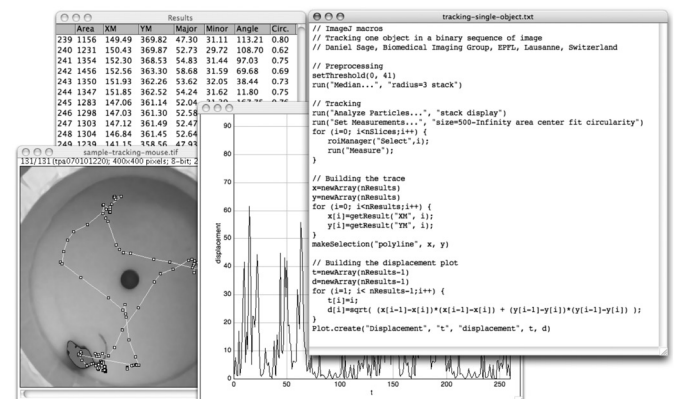


fig. 2 – exemple d'une macro ImageJ. Avec une dizaine de lignes de code seulement, ce programme peut faire le tracking d'une souris de laboratoire et tracer ses déplacements, dans des conditions particulièrement favorables: une seule cible toujours visible et bien contrastée.

Dans ImageJ, les macros assurent le premier niveau de programmation en permettant d'automatiser une séquence de tâches de traitement d'images. Les macros offrent la possibilité d'appeler des fonctions de ImageJ (fig. 2). L'apprentissage du langage macro

est aisé car les appels des fonctions sont enregistrables dans des macros—à la façon Excel. Depuis quelques mois, ImageJ reconnaît aussi Javascript comme langage de macro.

Même s'il existe des macros très avancées, les limites de la programmation par macro sont vite atteintes dans le cadre d'un travail professionnel, pour des raisons de faible performance, d'interaction homme-machine limitée et pour le génie logiciel très restreint des macros.

Les *plugins* constituent le niveau avancé de programmation [8, 9]. Ils sont écrits en Java et permettent d'accéder non seulement aux fonctions de ImageJ mais également à tout le langage Java et à l'ensemble de ses bibliothèques. En particulier, il permet de créer des interfaces utilisateurs (GUI) qui peuvent interagir avec les données image (fig. 3). Il est également possible d'accélérer certaines opérations en faisant appel au *multithreading* de Java. Bien entendu, l'écriture de *plugins* s'adresse à des personnes qui ont une base de programmation suffisante en Java. L'accès aux sources documentées de ImageJ facilite le développement de *plugins*.

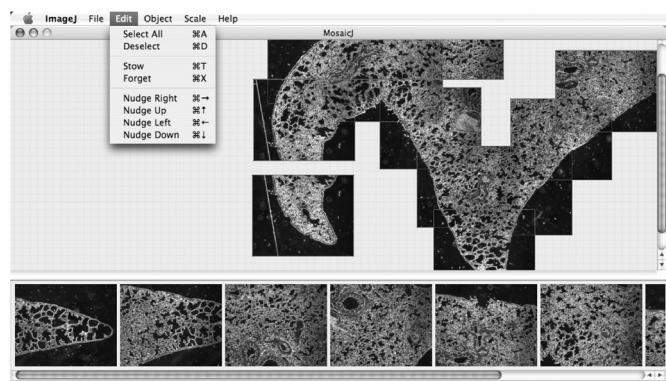


fig. 3 – MosaicJ (Philippe Thévenaz) est un plugin pour l'assemblage semi-interactif d'une mosaïque d'images. L'alignement précis des images est automatisé en recourant au plugin TurboReg [18]

## ImageJ et ses distributions

L'architecture logicielle ouverte et la licence domaine public a stimulé la créativité des développeurs et a permis l'émergence de différentes distributions qui rencontrent un grand succès dans leurs domaines respectifs.

### ImageJ, le classique

La version ImageJ du NIH est la plus légère. Pour un Mac, la version actuelle 1.46f ne pèse que 5.5 Mo, autant dire qu'il faut moins d'une minute entre l'intention d'utiliser ImageJ et l'ouverture de la première image sur ImageJ. Pas de librairie exotique à installer, pas de DLL ! Tout ImageJ est contenu dans un seul répertoire que l'on peut déplacer. La version pour Windows vient avec une machine virtuelle Java (JVM ou JRE); ainsi, tout marche après quelques clics.

### µManager, le pro pour l'acquisition

µManager est un logiciel *open source* (licence BSD / Lesser GPL) pour le contrôle et l'automatisation des tâches d'un microscope [10]. Combiné avec ImageJ, il offre une large gamme de pilotes (*drivers*) pour les microscopes les plus communs, pour les caméras scientifiques les plus courantes et pour les actionneurs motorisés de platine de microscopes.

### Fiji, Fiji is just ImageJ – batteries included

Fiji est un logiciel *open source* sous licence GPL, entièrement basé sur ImageJ. Il est hébergé sur un GIT version *control* par une petite communauté de développeurs passionnés [11]. On y retrouve tout ce qui fait ImageJ, complété par un ensemble cohérent et documenté de *plugins* dédiés aux sciences du vivant et par de nombreux langages de *scripting*, tels Javascript, Jython, Ruby, ... Fiji est certainement la distribution la plus aboutie d'ImageJ en termes de fonctionnalités, de développement communautaire et de documentation (fig. 4).

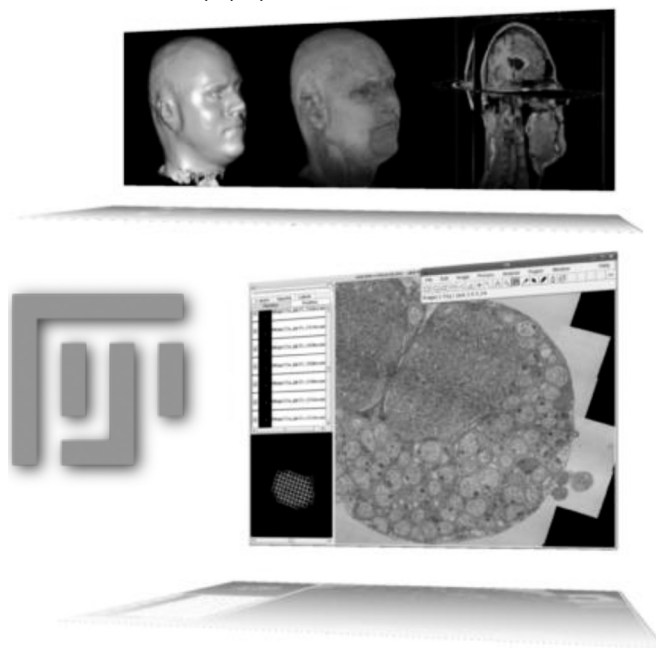


fig. 4 – montage illustrant la distribution Fiji de ImageJ. En haut, une reconstruction d'une image IRM avec le plugin 3D Viewer (Benjamin Schmid). En bas, segmentation d'une image de cerveau d'un drosophila avec le plugin TrakEM (Albert Cardona).

### Autres projets avec un coeur ImageJ

**Bio7** est un logiciel *open source* sous licence GPL/EPL dédié aux sciences de la terre et de l'environnement. Il démontre que ImageJ est suffisamment polyvalent pour être utilisé en dehors de l'imagerie biomédicale [6].

**SalsaJ** est une *spin-off* de ImageJ consacrée à l'astronomie.

**ImageJ for Microscopy** est une distribution populaire d'ImageJ avec des *plugins* spécifiques pour le microscope. Toutefois, la maintenance de cette distribution a été interrompue.

**ImageFlow** est une tentative de construire un environnement de développement pour ImageJ basé sur la programmation visuelle.

## ImageJ, l'indispensable compagnon de bioimagerie

Depuis les récentes découvertes des protéines fluorescentes et l'apparition de nouvelles générations de microscopes haute-résolution 3D nous sommes dans une nouvelle ère, celle de l'imagerie de cellules vivantes [12]. Cette capacité de différencier *in vivo* des composants cellulaires et leurs interactions à l'échelle moléculaire révèle la complexité de la machinerie de la cellule. La bioimagerie est la source de données hyper-volumétrique souvent jusqu'à 5D, les trois dimensions spatiales, la dimension *multichannel* et le temps.



Dans ce nouveau secteur en permanente évolution, les logiciels commerciaux ont surtout brillé par leur absence. ImageJ s'est imposé naturellement comme le logiciel ouvert et il demeure l'outil privilégié pour l'analyse d'images en sciences du vivant.

À titre d'exemple, on peut citer quelques applications où ImageJ/Fiji se démarque:

**Colocalisation:** observation des recouvrements dans différents canaux fluorescents (*plugins*: JACoP, Colocazion);

**Cytométrie:** extraction des cellules ou des composants nucléaires en 2D ou 3D (*plugins*: TrakEM2, *Image-based Tool for Counting Nuclei*);

**Segmentation couleur:** segmentation basée sur la couleur (*plugins*: Threshold Colour, Color Segmentation, Color Inspector 3D);

**Tracé de neurone:** reconstitution de la morphologie neuronale (fig. 5) (*plugins*: NeuronJ, NeuriteTracer);

**Super-resolution by photoactivated localization:** localisation de molécules fluorescences photo-activées (PALM) permettant de gagner un ordre de grandeur dans la résolution spatiale (*plugins*: Octane, QuickPALM).

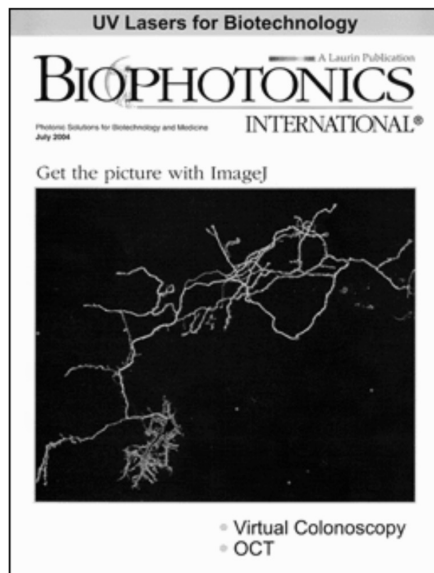


fig. 5 - NeuronJ (Erik Meijering) est un plugin pour le tracé interactif de l'arbre dendritique d'un réseau de neurones

### ImageJ à l'EPFL

Des dizaines de laboratoires de l'EPFL utilisent quotidiennement ImageJ, comme d'autres instituts ou d'autres entreprises. ImageJ est très répandu dans les plates-formes de microscopie, comme celle de la Faculté SV, la plate-forme technologique de bio-imagerie et imagerie optique (PTBIOP).

Dans le laboratoire d'imagerie biomédicale (BIG, *Biomedical Imaging Group*) du Prof. M. Unser, nous avons pris le virage ImageJ en l'an 2000, en premier lieu pour répondre aux besoins de l'enseignement. Nous avons mis en place une structure simple et didactique pour enseigner le codage des algorithmes usuels de traitement d'images en nous appuyant sur une classe `ImageAccess` que nous avons développée [13]. Dans le cadre des cours de *Master, Image Processing I et II*, nous organisons des séances de laboratoire qui se déroulent sur trois heures et en temps limité. Chaque année, ce sont presque cent-cinquante étudiants au premier semestre

et cinquante étudiants au second semestre qui suivent ces cours. Tous ces étudiants sont amenés à implémenter individuellement des algorithmes en Java et à les tester sur des images.

En observant les performances, la stabilité et la convivialité d'ImageJ, nous avons compris que ImageJ offrait également un excellent environnement de travail pour développer de nouveaux algorithmes de traitement d'images et qu'il pouvait servir de plate-forme d'échange pour nos collaborations avec les chercheurs en sciences du vivant.

Au BIG, nous avons porté plusieurs nouveaux algorithmes sous forme de *plugins* (fig. 6) sous licence *citationware*. Certains ont été développés par des doctorants, d'autres par des étudiants [14].

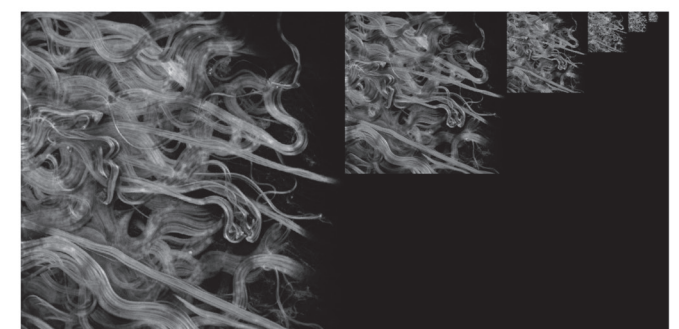
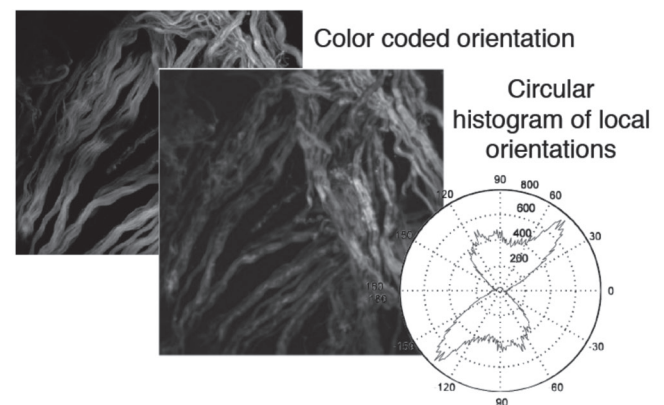
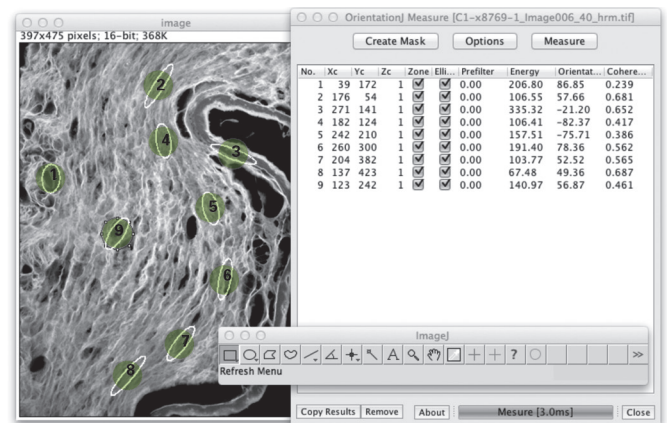


fig. 6 - montage illustrant l'analyse directionnelle d'images. En haut, un outil mesure l'orientation locale des fibres. Au centre, le plugin OrientationJ donne la distribution des orientations des fibres de collagène. En bas, le plugin MonogenicJ représente l'information d'orientation dans différentes échelles obtenues par une décomposition en ondelettes orientable (steerable wavelets). Images provenant du laboratoire LHCT de l'EPFL

**PureDenoising:** débruitage multi-canaux basé sur des transformées en ondelettes (*wavelets*).

**DeconvolutionLab:** outil pour la déconvolution 3D en microscopie qui rivalise avec les produits commerciaux.

**PSFGenerator**: création de *Point-Spread Function* 3D pour la microscopie.

**EDF**: extension de profondeur de champ pour microscopie à champ large.

**TurboReg**, **StackReg**, **MosaicJ**: alignement (*registration*) d'images et de séquences d'images.

**SteerableJ**, **OrientationJ**, **MonogenicJ**: analyse directionnelle par des ondelettes orientables.

**Snakuscul**, **Ovuscul**: famille de contours actifs (*snakes*) pour la segmentation.

**SpotTracker**: suivi (*tracking*) de particules individuelles dans des images bruitées.

### 2012, l'année du renouveau pour ImageJ

Wayne Rasband est désormais à la retraite. Avec sa générosité habituelle, il continue à maintenir ImageJ et à faire évoluer ImageJ en y intégrant les contributions de développeurs. Maintenant, tous les regards se tournent vers l'équipe de ImageJDev [15] qui s'est constituée en 2010 autour d'un projet fédérateur de développement de la nouvelle génération d'ImageJ, nommé ImageJ2. L'ambition est de reconstruire toute l'architecture logicielle ImageJ, y compris le modèle des données, tout en garantissant une presque parfaite rétro-compatibilité (*backward compatibility*) avec les *plugins* d'ImageJ. C'est un défi de taille, car la représentation des images aux multiples facettes (3D, 4D, 5D, multi-canaux, virtualisation) doit satisfaire autant l'utilisateur par sa rapidité que le développeur par sa simplicité.

2012 voit émerger un tout nouveau logiciel Java d'analyse d'images, ICY, spécialisé en bioimagerie [16]. ICY a défini un modèle homogène des données pour la bioimagerie. Il s'appuie sur la librairie VTK pour offrir un rendu efficace d'images 2D et 3D. Des tentatives de communication avec les *plugins* ImageJ sont en cours de développement.

Enfin, l'aspect logiciel d'analyse est désormais reconnu comme une infrastructure à part entière dans le domaine de l'imagerie biomédicale; en conséquence, il fait l'objet d'un *workpackage* du projet européen **Euro-BioImaging**. Notre laboratoire organisera le 30 avril et le 1er mai 2012 un workshop [17] consacré aux logiciels Java d'analyse d'images biologiques pour tenter de dessiner les contours de l'après ImageJ. Si vous avez réalisé un *plugin*, c'est une belle occasion de venir à Barcelone pour le présenter !

### Références

- [1] RASBAND, W. *ImageJ*, National Institutes of Health, Bethesda MD, USA. [imagej.nih.gov/ij/](http://imagej.nih.gov/ij/)
- [2] ABRAMOFF, M. D. et al. *Image Processing with ImageJ*, Biophotonics International 11, pp. 36-42, 2004.
- [3] LINKERT, M. et al. *Metadata Matters: Access to Image Data in the Real World*, Journal of Cell Biology 189, pp. 777-782, 2010. [loci.wisc.edu/software/bio-formats/](http://loci.wisc.edu/software/bio-formats/)
- [4] JAHNEN, A. *ImageJ Information and Documentation Portal*, CRP Henri Tudor, Luxembourg. [imagejdocu.tudor.lu/](http://imagejdocu.tudor.lu/)
- [5] CARDONA, A. *TrakEM2*. [fiji.sc/wiki/index.php/TrakEM2/](http://fiji.sc/wiki/index.php/TrakEM2/)
- [6] AUSTENFELD, M. *Bio7*. [bio7.org/](http://bio7.org/)
- [7] SAGE, D. et al., *MJ*. [fiji.sc/wiki/index.php/Miji/](http://fiji.sc/wiki/index.php/Miji/)
- [8] BURGER, W. and BURGE, M. J. *Digital Image Processing*, Springer Verlag, 2008. [www.imagingbook.com/](http://www.imagingbook.com/)
- [9] BAILER, W. *Writing ImageJ Plugins – A Tutorial*. [www.gm.fh-koeln.de/~konen/WPF-BV/tutorial-ImageJ\\_V1.71.pdf](http://www.gm.fh-koeln.de/~konen/WPF-BV/tutorial-ImageJ_V1.71.pdf)
- [10] EDELSTEIN, A. et al., *Computer Control of Microscopes Using µManager*, Current Protocols in Molecular Biology 14, 2010. [valelab.ucsf.edu/~MM/MMwiki/](http://valelab.ucsf.edu/~MM/MMwiki/)
- [11] SCHINDELIN, J. *Fiji Is Just ImageJ (Batteries Included)*. [fiji.sc](http://fiji.sc)
- [12] VONESCH, C. et al., *The Colored Revolution of Bioimaging*, IEEE Signal Processing Magazine 23, pp. 20-31, 2006. [bigwww.epfl.ch/publications/vonesch0601.html](http://bigwww.epfl.ch/publications/vonesch0601.html)
- [13] SAGE, D. and UNSER, M. *Teaching Image-Processing Programming in Java*, IEEE Signal Processing Magazine 20, pp. 43-52, 2003. [bigwww.epfl.ch/teaching/iplab/](http://bigwww.epfl.ch/teaching/iplab/)
- [14] EPFL-BIG members, *Available Algorithms for ImageJ and Fiji*. [bigwww.epfl.ch/algorithms.html](http://bigwww.epfl.ch/algorithms.html)
- [15] RUEDEN, C. and ImageJdev team, *ImageJ2*. [developer.imagej.net/](http://developer.imagej.net/)
- [16] Institut Pasteur, *Icy: The Open Source Community Software for Bio-Imaging*. [icy.bioimageanalysis.org/](http://icy.bioimageanalysis.org/)
- [17] Euro-Bioimaging Workshop, *Bioimage Analysis Software: Is there a Future Beyond ImageJ?* [bigwww.epfl.ch/eurobioimaging/](http://bigwww.epfl.ch/eurobioimaging/)
- [18] THÉVENAZ, P. et al., *A Pyramid Approach to Subpixel Registration Based on Intensity*, IEEE Transactions on Image Processing 7, pp. 27-41, January 1998. [bigwww.epfl.ch/thevenaz/turboreg/](http://bigwww.epfl.ch/thevenaz/turboreg/)



L'icône de démarrage d'ImageJ est la reproduction d'une photographie du microscope de Hartnack, qui date des années 1870. C'est un contraste assez saisissant que de double-cliquer sur cette icône d'un autre âge pour analyser des images provenant d'appareils à la pointe du progrès, comme des microscopes 3D confocaux à fluorescence capables de percevoir des protéines.



Article du FI-EPFL 2012 sous licence CC BY-SA 3.0 / D.Sage