TrakEM2 workshop

Albert Cardona

1 Introduction

2 Motivation

Our current work in neuroscience requires detailed anatomical descriptions of dendrites and axons (the wires), and synapses (the contact points). While we can inspect most wiring details at light microscopy with specific color labels, synapses and very fine processes so far lay under the sensitivity and resolution thresholds. The electron microscope, on the other hand, enables high-resolution imaging of such structures, but with severe costs: sections are not registered to each other; imaging itself induces non-uniform, non-linear deformations; and the sheer amount of image data obtained becomes unmanageable for a human operator.

To address these and related issues, we have developed TrakEM2: an ImageJ plugin that virtualizes access to very large data sets, and makes its analysis feasible.

The purpose of this workshop is to provide an overview of TrakEM2 capabilities, and to illustrate them with specific examples of everyday use.

2.1 Overview of TrakEM2 features

TrakEM2 provides the means to:

1. Seamlessly work with terabyte-sized data sets.
2. Create montages from a collection of tiles.
3. Register montages to each other (i.e. to align serial sections).
4. Precisely segment structures such as neurites across multiple registered sections.
5. Sketch neurites (and other structures) with simplified tubes and spheres.
6. Perform measurements such as double dissectors for synapse density estimation, and for areas, lengths and volumes.
7. 3D visualization of segmented elements and image volumes via Bene Schmid's 3D Viewer plugin.
8. Annotate with floating text labels (and search them with regular expressions).

9. Compare tubular structures (such as neurites or blood vessels) for identification purposes relative to reference data.

TrakEM2 works also with confocal stacks, and any sort of images supported by ImageJ.

3 How to install TrakEM2

- Via Fiji: Fiji is Just ImageJ, batteries included. There are Fiji packages available for all main operating systems (at http://pacific.mpi-cbg.de/wiki/index.php/Downloads) and also via *git* (a distributed version control system) as explained in the *how to* webpage (`http://www.ini.uzh.ch/~acardona/howto.html#fiji_install`). With *git* its easier to stay up to date with new releases. Fiji includes a JVM and java3d.
- In a regular ImageJ installation: download all required jar files as listed in `http://www.ini.uzh.ch/~acardona/howto.html#install_trakem2`, and install java3d (already included by default in MacOSX).

4 TrakEM2: an overview

4.1 The project

A TrakEM2 project consists of all image transformations, annotations, segmentations and displays. Images themselves are not stored; only the file path (i.e. their location in the file system).

4.2 The XML file

When saving a project, TrakEM2 generates an XML file containing all the relevant information to reopen itself. It is **strongly** recommended to save the XML file at the root folder of all images. In this fashion, the XML file will contain only *relative file paths* to the image files. Then the entire parent folder may be moved across computers and file systems without risk of breaking file paths to images.

The XML file itself, despite its occlusive verbosity, may be edited manually when needed. A typical case is when files have been added whose file paths could not be made relative to the XML file itself, and which then may fail to be found when moving the project to a different computer.

4.3 The displays, the layers and the layer sets

A **layer** is what ImageJ calls a **slice**. In each layer, any number of images are hosted, along with floating text labels and segmentations. Each layer has a Z coordinate and a thickness, independent of any other layers.

A **display** is simply an interactive canvas for viewing and editing the contents of a layer: drag images, register them, add new images by drag and drop, put floating text labels, etc. Within a display any number of layers may be browsed –just like slices in a stack.

Table 1: Correspondences between ImageJ and TrakEM2 containers. Note that TrakEM2 containers are far more flexible (see text).

ImageJ	TrakEM2
slice	layer
stack	layer set
ImageWindow	Display

All layers belong to a **layer set**, a substitute for a stack: a layer set may contain layers with repeated Z coordinate and unequal thickness, and each layer in the layer set may contain any number of nested layer sets –so your transmission electron microscopy serial sections may logically be embedded in a light-imaged histological section.

4.4 The trees: Template Tree, Project Tree, Layer Tree

The main TrakEM2 window shows 3 trees side by side, which provide access to all data stored within TrakEM2. From right to left (see figure 1):

- **The Layer Tree**: from the root layer set, lists all layers and any nested layer sets that these may have. From here, with a selection or multiple selection (with shift), via right-click, the layer's Z coordinate, thickness and title may be edited.
- **The Project Tree**: shows the root node –the project itself–, and all abstract nodes and segmentation nodes. So your segmentations need not be an unmanageable list, but a tree with collapsible branches showing, for example, abstract nodes like tissue, neuron, soma, dendritic tree, synapse ... and segmentation objects like *area list*, *ball*, *pipe* and *dissector* assigned to any of these abstract nodes.

  Via right-click menu, any node's title may be edited, and the node (and all its nested children) measured or deleted.
- **The Template Tree**: purely abstract, this tree is a declaration of intentions: what abstract nodes should there be, and what kind of children may they have. See figure 1 for an intuitive view. Only template nodes and their relationships as specified in this tree may be instantiated in the Project Tree.

  Via right-click menu, new nodes may be added, existing ones deleted or renamed, and the whole or a subset of the tree (depending upon calling the menu on the root node or any other node, respectively) may be exported as a .dtd file for reuse in new TrakEM2 projects to be created from this template.

4.5 The displays

A TrakEM2 **display** (Fig. 4.5) is a regular ImageJ canvas with expanded capabilities. The canvas has virtual dimensions, editable anytime independently of the images, and which range from 0 to Double.MAX_VALUE in width and height (way more than you'd ever use).

The display is the main TrakEM2 interface. Image files may be imported into it by drag and drop, or by means of the contextual menu (right-click, or control-click in MacOSX). Objects other than images,

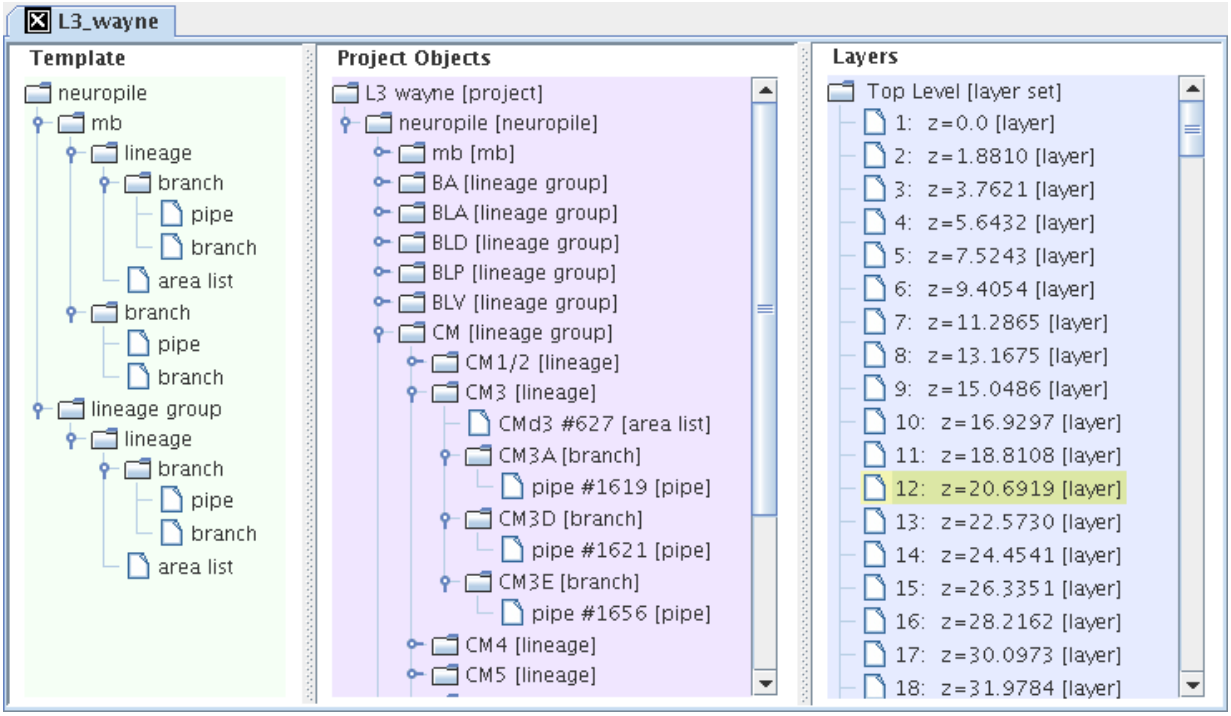

Figure 1: **The three trees**, as an interface for editing and visualizing the three internal TrakEM2 data structures.

such as segmentations (area list, pipe, ball, profiles) and measurements (all ImageJ ROIs, dissectors), are also edited and visualized within the display.

A display shows only one layer at a time, but offers a scroll bar (and the mouse scroll wheel) for browsing through layers (just like ImageJ browses through slices in a stack).

Standard ROIs are used inside the display for measuring (lengths, areas, angles). Support for proper pixel measurement with ROIs is currently under development.

4.6 Navigation

The hand and glass tools are used for panning and zooming respectively, like on any other ImageJ image window. But also, in accordance to standard means of navigation present in other software (such as Inkscape), the middle mouse button is used for panning, and control+scroll wheel (apple+scroll wheel in MacOSX) for zooming (the latter offers position-invariant zooming relative to the mouse pointer, very useful for zooming in several orders of magnitude without losing the desired landing area).

4.7 Selections

Selections are also managed through displays. A selection is different than an ImageJ ROI: not a **r**egion **o**f **i**nterest, but a group of one or more objects over which operations such as translation, rotation, scaling, deletion, contrast correction, transparency adjustment, measurements, etc., may be performed.

- Click on an image or other object to select it. If more than one lay under the mouse, a menu pops up to select one.
- Shift+click to select multiple objects.
- Shift+click an object in a multiple selection to set it as active (notice its white frame, as opposed to selected but non-active pink frame).
- Shift+click a selected and active object to deselect it.
- Draw a ROI and then right-click and choose "Selection - Select all under ROI" for idem.

4.8 Transforming objects: rotate, translate, scale

On selected objects, pushing 't' will bring up a transformation box for rigid deformation: scale, translate (by dragging the whole box) and rotate relative to a draggable floating pivot. The contextual menu will have options to apply (enter), cancel (escape) and specify a transformation.

4.9 The 3D viewer

The primary means of 3D visualization consists of Bene Schmid's 3D Viewer. To visualize an object in 3D, select it in a display and then right-click and pick "Show in 3D". From the Project Tree, any child objects,

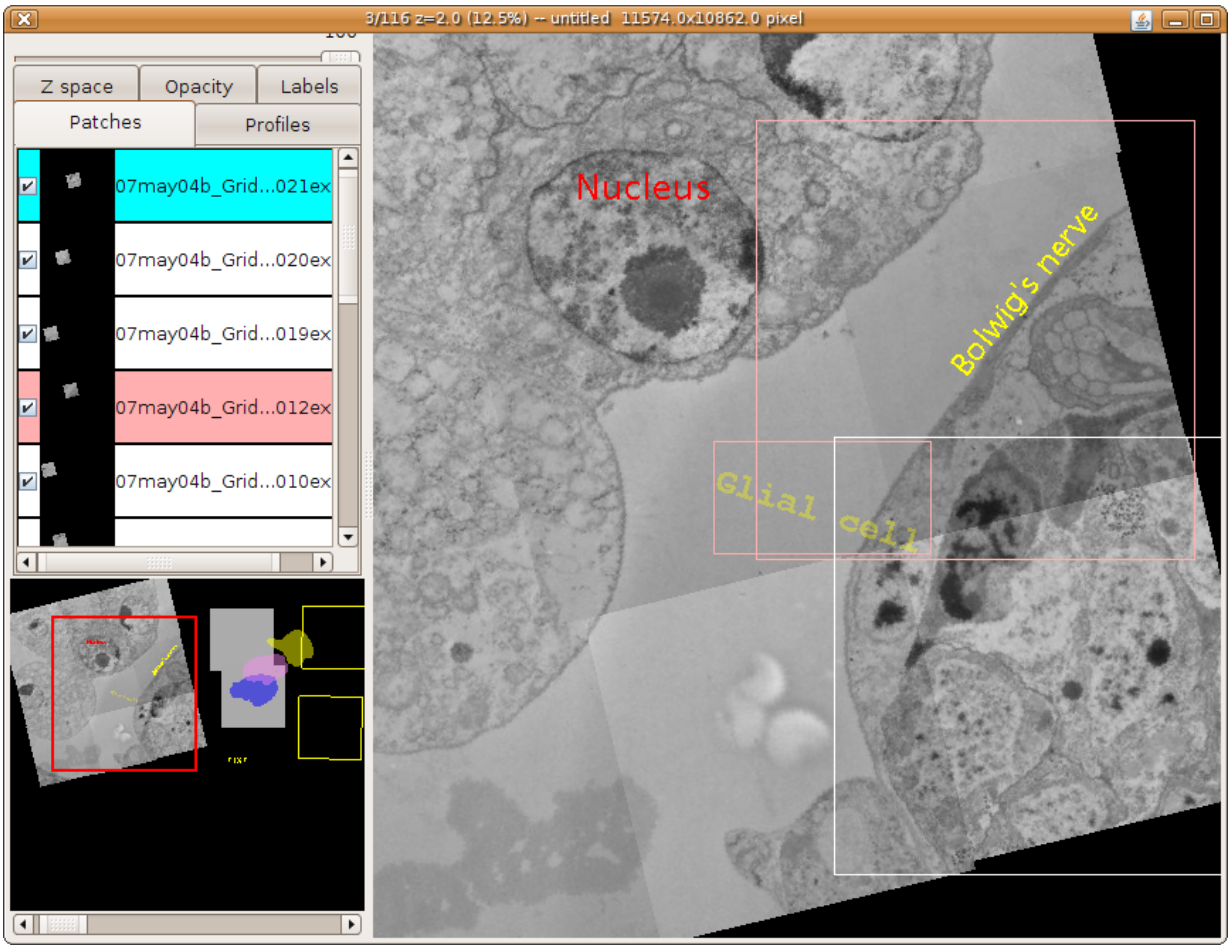

Figure 2: **A TrakEM2 Display**, showing 9 images in a layer, where 2 images and one floating text label (set to 30% transparency) are selected (*pink* and *white* frames; *white* is the active one – note the corresponding *pink* and *blue* coloration of the object panels *on the left*). The Navigator (*bottom left*) paints a red frame to indicate the area currently displayed in the canvas (*right*).

recursively, will also be shown if they are not hidden (when hidden, a notice is printed into ImageJ's log window).

Changes in the transparency of an object from a Display will be effective in the 3D Viewer as well.

To navigate a 3D Viewer, select the hand tool and then:

- Drag to rotate the view.
- Shift+drag to pan the view.
- Scroll wheel to zoom.
- Mouse over an object to see its name printed in ImageJ status bar.

5 Creating, saving and opening TrakEM2 projects

5.1 Creating a TrakEM2 project

In Fiji, go to menu "File - New - TrakEM2". Choose either blank (a completely new and empty project), or from template; the latter offers a file dialog to choose a .dtd or TrakEM2 .xml file containing a Template Tree. You can generate .dtd files from any subset of an existing Template Tree from its contextual menu.

From a standard ImageJ installation, identical options are present under "Plugins - TrakEM2" menu.

5.2 Save and Save As

Saving a TrakEM2 project is as simple as pushing 's' key on a Display or on the Project Tree. Otherwise right-click on a Display and select "Project - Save".

To "Save As", right-click a Display and select the menu "Project - Save As". The current .xml file path of the project will change to that of the newly chosen one. So subsequent calls to 's' or "Save" will overwrite that new .xml file.

5.3 Opening a TrakEM2 project

In Fiji, as simple as drag and drop of the .xml file onto ImageJ status bar. Otherwise, adjust your HandleExtraFileTypes or use the menu "Plugins - TrakEM2 - Open Project".

6 Importing images

- Importing a single image: drag and drop an image from a directory into the canvas. Alternatively, right-click on the canvas and select "Import / Import image..."

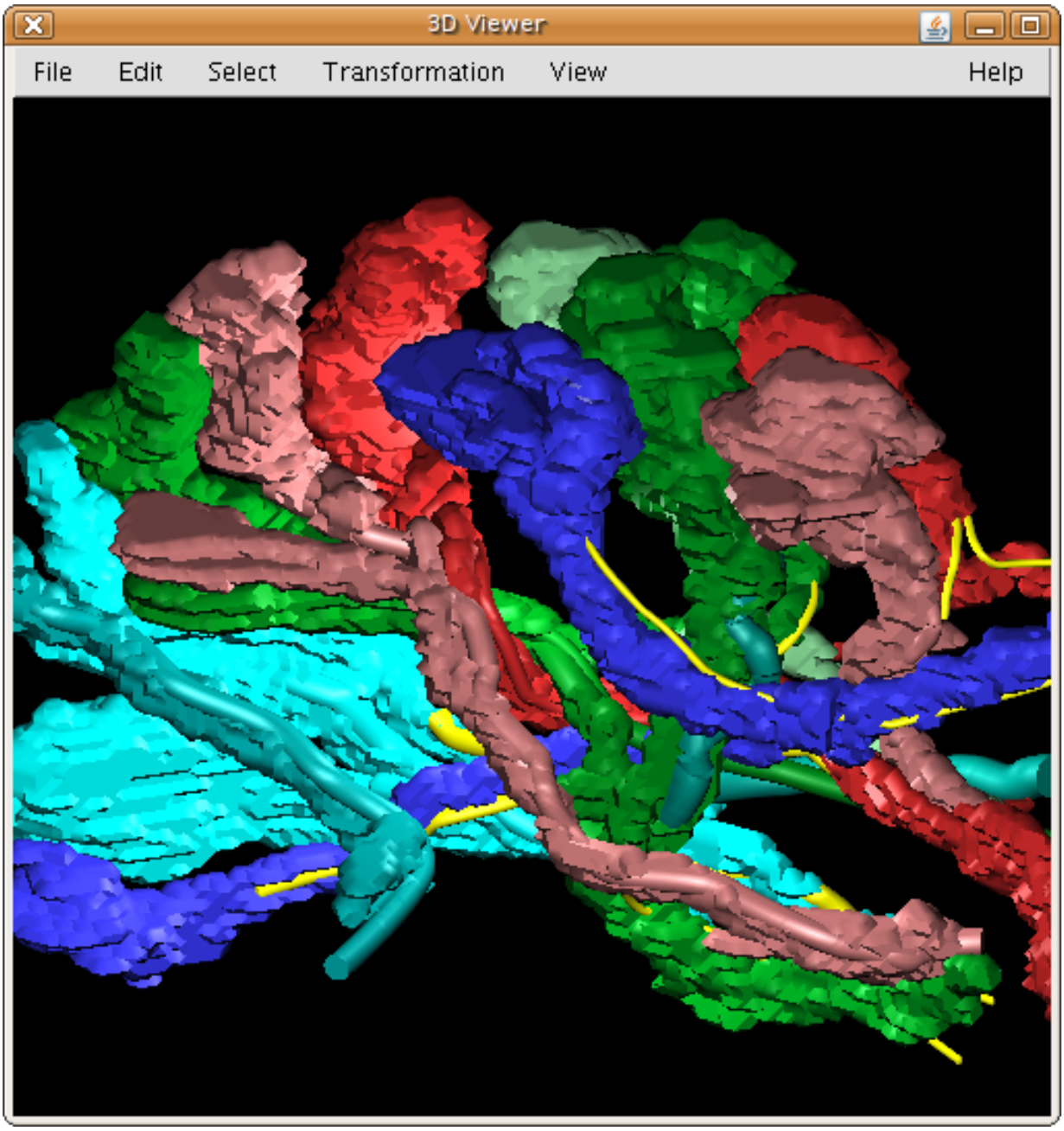

Figure 3: **The 3D Viewer**: An example of meshes (as generated by marching cubes from AreaList) and pipes on the 3D display. All other segmentations may also be displayed, namely profile lists (by CurveMorphing; see [1]) and balls (the latter as spheres).

- Importing a list of images as a montage: drag and drop a folder full of images into a canvas. Alternatively, right-click and select either "Import / Import grid..." or "Import / Import sequence as grid...". A dialog with placement and stitching options will popup in any case, including the ability to correct brightness and contrast across tiles.
- Importing a list of images into multiple layers: create a text file containing one image per row, with four columns: file path, X, Y and Z (the layer). All coordinates may be scaled via a dialog, when not in pixels. Columns may be separated by either a single space, a tab or a comma. TrakEM2 will automatically create new layers to accommodate any given Z value. To import from the text file, right-click on a Display and call "Import - Import from text file...".

  As an example, a montage of 4 tiles of 1024x1024 in a layer with Z coordinate zero, and an overlap of 100 pixels:

```
/path/to/image1.tif 0 0 0
/path/to/image2.tif 924 0 0
/path/to/image2.tif 0 924 0
/path/to/image4.tif 924 924 0
```

7 Registering images: automatic, semi-automatic and manual

Automatic registration is available for all image importing methods. In addition, images may be registered within an open Display any time. Three registration modes are show below.

7.1 Automatic registration

Select a two or more images and then choose "Montage" from the contextual menu. Options are provided to tweak the settings of the underlying SIFT [2] feature extraction[1].

7.2 Semi-automatic registration or snapping

Choose an image and select "Snap" from the contextual menu. A good attempt is made to stitch it to the best possible option among any images with which it intersects.

7.3 Manual registration

The space bar will toggle the transparency of the active, selected image to 50% and back to 100%. Then drag and transform the image (with 't' or "Transform" from the contextual menu) until it fits.

[1] Built by Stephan Saalfeld

8 Floating text labels

Clicking on a display with the text tool will popup a text window to type in text. On closing the window, the text is set as a floating object inside the display (see Fig. 4.5). Like any other object, you may drag, scale, rotate, adjust its transparency, etc. To change the text color, be sure to select the text object and then double-click the color dropper tool to open the color palette. Click on the palette to set the color of any selected text object.

Alternatively, select "Properties" from the contextual menu. Beyond the color, the font type, style and size may be adjusted as well. By default, a new text object takes the font as specified in ImageJ's font dialog (double-click the font tool to open it).

9 3D modeling

9.1 Segmentation types: AreaList, Pipe, Profile and Ball

All segmentation types are created and destroyed from the Project Tree, not from a Display (as opposed to images and text labels, where the opposite applies). This rule ensures consistency, since all segmentations are but interpretation, not hard data.

All segmentation types are edited with the pen tool.

The **AreaList** is akin to an ImageJ ShapeRoi, and consists of any number of disconnected areas painted over any number of sections. A 3D mesh is created by marching cubes. Click+drag paints; alt+click+drag deletes; shift+click will fill a hole; and shift+alt+click will delete a separated island. All operations affect the currently 2D visible part only. Adjust the size of the brush with shift+scroll wheel.

The **Pipe** is a tube in 3D space, represented by any number of points over a Bézier curve. Each point has a radius, adjustable by shift+drag. Adjust the handles by dragging or by alt+drag over the backbone point. Delete one point by shift+control+click. New points may be added at the end, at the beginning, or anywhere over the existing tube.

The **Ball** is a collection of points with a radius each. Each point may sit on any layer throughout the layer set. Adjust its radius with shift+drag; delete one point with shift+control+click.

The **Profile** is a 2D-only outline made of an open or closed Bézier curve. Close it by dragging on the last point, or shift+click to toggle open/close. Shift+control+click to delete a backbone point.

The 3D visualization of segmentation types is tree-base, recursive, and skips hidden nodes. I.e. on the Project Tree, select a node and right-click, then choose "Show in 3D". All children nodes, recursively, will be shown in the 3D display unless marked as hidden.

10 Measuring

Measurements depend heavily on the calibration. Be sure to call, on an open display, the menu "Image - Properties..." or "Analyze - Set Scale" beforehand.

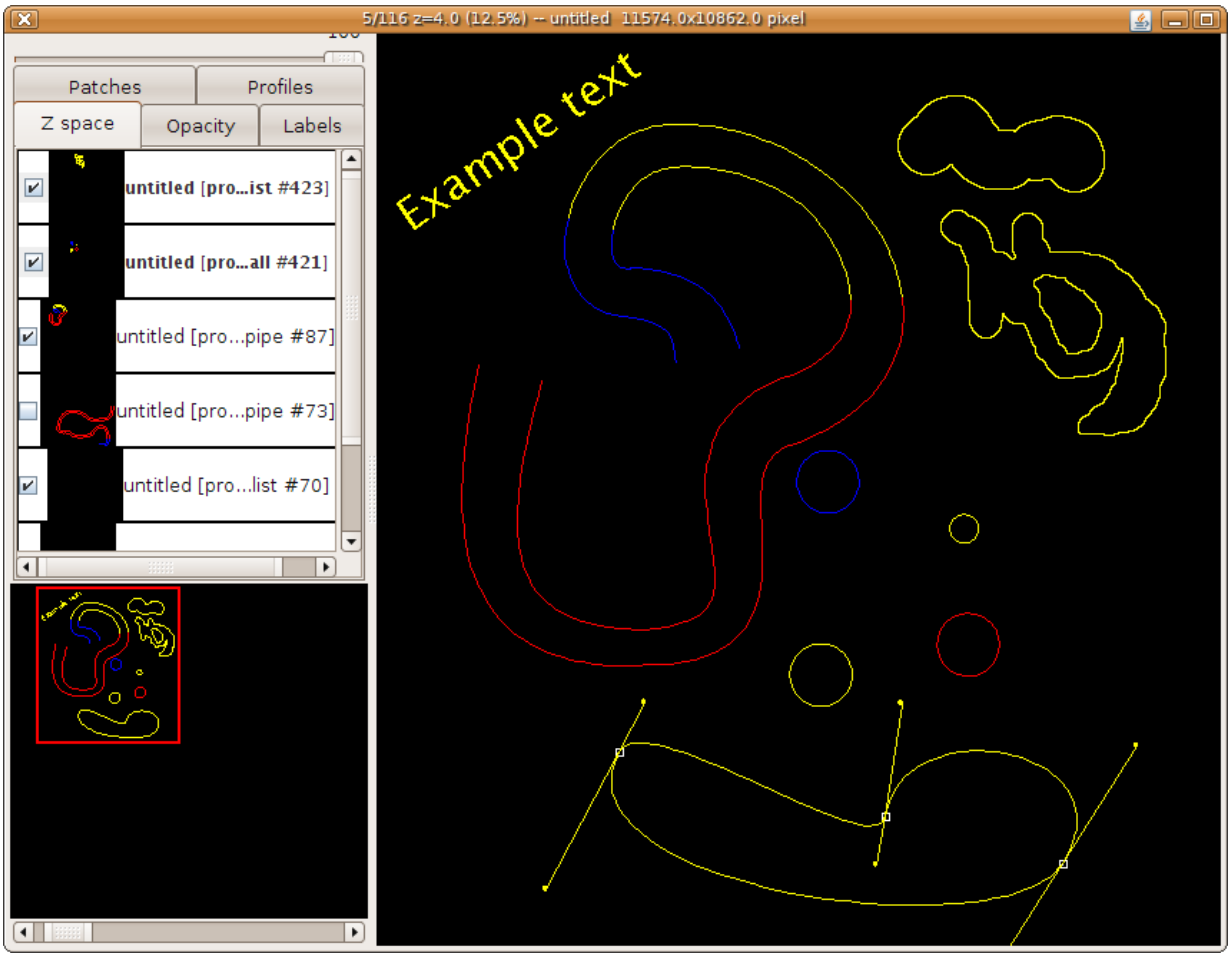

Figure 4: **A TrakEM2 Display**, showing an example of each type of segmentation: a Pipe (*left*), a Ball (*center*; 4 points), an AreaList (*top-right*), and a Profile (*bottom*). The segmentations sit over an empty canvas –no images– for clarity. The color cues indicate that part of the object is above or in the previous layer (*red*) and below or in the next layer (*blue*). Color cues may be toggled with 'p'. *Yellow* is the default color for a segmentation, and here it indicates the part of the object sitting on the current layer.

10.1 With ImageJ's built-in ROIs

As simple as selecting the appropriate ROI tool from ImageJ tool bar and drawing it on a display, then pushing 'm' or selecting the menu "Analyze - Measure".

10.2 Measuring segmentations

Like 3D visualization, measuring is tree-based, recursive, and skips hidden nodes. Each segmentation type will end up measured in its own results table. As many results table as involved segmentation types will be opened.

11 Scripting in Javascript

The following examples will work with ImageJ 1.41 and later and java 1.6.0 and later. The latter includes the Rhino javascript engine within its libraries.

An easy way to run the examples below is by calling "Plugins - New - Javascript", pasting the code below into the text window, and calling "Macros - Evaluate javascript". Alternatively, use the "Plugins - Scripting - Javascript Interpreter" from Fiji.

For permanent plugin creation, just create a file with a *.js* extension and an underscore in its file name, and place it in ImageJ's plugins folder. On restart, ImageJ will automatically recognize it as a valid script.

11.1 Importing an image

```
importPackage(Packages.ini.trakem2.display);
layer = Display.getFront().getLayer();
filepath = "/path/to/image.png";
imp = IJ.openImage(filepath);
x = 300;
y = 500;
patch = new Patch(layer.getProject(), x, y, imp);
patch.getProject().getLoader().addedPatchFrom(filepath, patch);
layer.add(patch);
// optional: enlarge/shrink the canvas dimensions to fit the new image
layer.getParent().setMinimumDimensions();
```

11.2 Querying properties of a selected image

```
importClass(Packages.ini.trakem2.display.Display);
importClass(Packages.ini.trakem2.display.Patch);

front = Display.getFront();
selection = front.getSelection();
IJ.log("Number of selected items: " + selection.getNSelected());

active = front.getActive();
IJ.log("Active is " + active);

// The min and max pixel values of a Patch, which wraps an ImagePlus:
if (active instanceof Patch) {
  IJ.log("min: " + active.min + ", max: " + active.max);
}

layer = active.getLayer();
IJ.log("Number of images:" + layer.getDisplayables(Patch).size());

layerset = layer.getParent();
IJ.log("Number of layers:" + layerset.size());

all = layerset.getDisplayables(Patch);
IJ.log("Total number of images:" + all.size());
```

11.3 Transforming the active image or object

```
importClass(Packages.ini.trakem2.display.Display);
front = Display.getFront();
active = front.getActive();

// Change transparency to 50%
active.setAlpha(0.5);

// Move 100 pixels to the right
active.translate(100, 0);
front.repaint(active);

// Rotate 45 degrees clockwise relative to center of image
box = active.getBoundingBox();
pivot_x = box.x + box.width/2;
pivot_y = box.y + box.height/2;
importClass(Packages.java.lang.Math);
radians = (45.0 / 180) * 3.14159265;
active.rotate(radians, pivot_x, pivot_y);
front.repaint(active);

// Change associated text
active.setTitle("A new name");
```

12 Acknowledgments

Without Wayne Rasband TrakEM2 would not have been possible – my deepest appreciation. TrakEM2 was born out of the needs of a biologist (A.C.) who was trying to image a fruit fly brain at 5,000 magnification (it's really huge), the vision of a neuroscientist, Rodney Douglas, and the infinite patience of Volker Hartenstein at UCLA. Over the last 3 years numerous people have left their mark in the form of great algorithms, most notably Stephan Preibisch and Stephan Saalfeld from Tomančák's group at MPI-CBG, Dresden; Bene Schmid and Johannes Schindelin, from Heisenberg's lab in Würsburg, and many others. My deepest thanks to you all.

References

[1] A. Cardona and V. Hartenstein, "Three-dimensional skin reconstruction by vector sequence alignment and morphing," in *Proceedings of the Blender Conference 2006*, (Amsterdam, The Netherlands), Blender Foundation, 2006.

[2] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.