Neuron segmentation/identification pipeline

This is a description of our neuron segmentation and identification pipeline. In brief, it is based on Yu Toyoshima's elegant algorithm (Toyoshima et al. 2016) for neuronal segmentation which we augmented with a MatLab GUI that displays neurons in 3D together with their activation to modify neuronal read-out zones. Aberrant read-out geometry can be corrected. Intensities can then be re-read from the original data. This serves only as documentation. This version of the pipeline is not maintained anymore. We recommend the use of more contemporary means of segmentation/identification.

Requirements:

- Matlab
- MatLab MCR 8.5 (specifically)
- Yu toyoshima's code
  FIJI
- MIJI (ImageJ matlab interface)

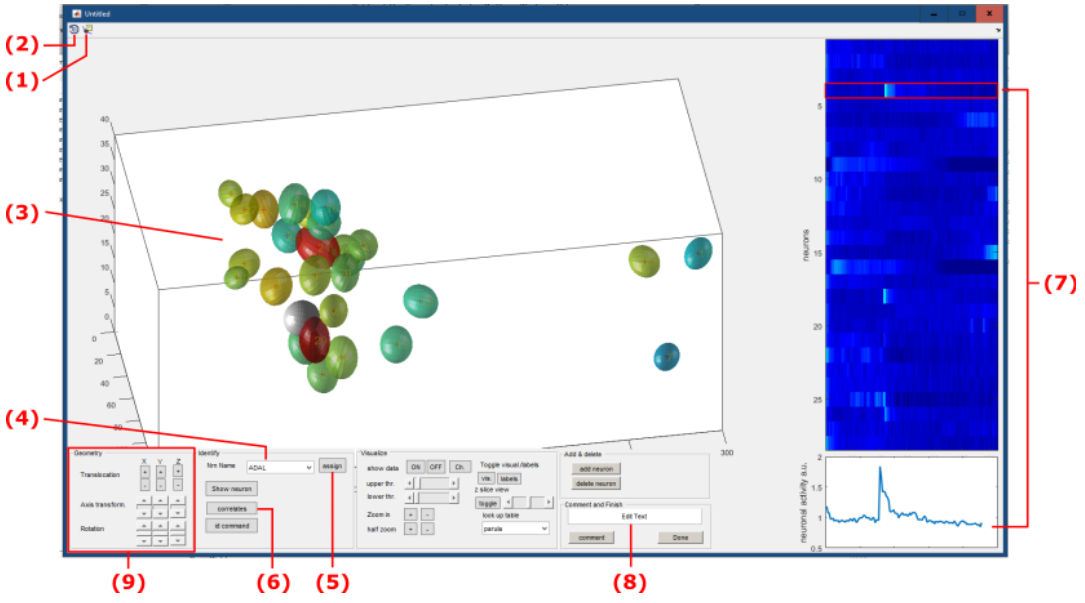
Installation:

- Install Yu Toyoshima's gaussian fitting package including MatLab MCR 8.5 as outlined in the instructions.
- Download/Install FIJI (move to an operating folder e.g. '`C:\Users\[your name]\ImageJ.app`')
- Install MIJI as outlined here.
- Download the code in the 'NSI code' and copy it to your preferred MatLab working folder.

How to:

- Once all components are installed, xyzt-series can be processed with Yu toyoshima's algorithm. After segmentation and tracking, the output is a mat file that includes a variable named 'data_edited'
- Change to Matlab.
- Add your local 'NSI code' folder including subfolders to the MatLab path.
- Browse to the folder containing the image data and the output

- Miji interface is activated by initPaths(‘YOUR-FIJI-PATH-AS-STRING’), e.g.: initPaths('H:\Fiji.app')
- Start the neuron visualisation/identification by calling ‘identifyNrns’ e.g.: identifyNrns(‘exampleData’,exampleData.tif);
  Name the neurons, adapt the read-out zones using UI (see figures below for explanations). Once finished, quit the procedure with the ‘Done’ button.
- Re-read the intensities from modified reat-out zones using batchExtractIntensities(‘PATH’), e.g.: batchExtractIntensities(cd)
- Instantiate neuronal activities, labels and the time vector. Choose the appropriate read-out by using ‘exportData’, e.g.: [data,labels,time] = exportData(‘interpolate’,2,90); Select the ‘processingPkg.mat’ file from directory when prompted by the UI. Follow the instructions by the prompt. Note that this script provides the option to correct the activities by the signal of another channel (e.g. a mCherry tag). Here, we refrained from this since the mCherry tag exhibited sometimes high bleaching rates or bleed-thru from other channels. This script provides the following output: neuronal activation data (data), neuronal labels (labels), and a time vector (time). Note that the intensities are only preliminary normalized by dividing by the 3% quantile of the each neuron’s activity. Moreover, the time vector needs to be synchronized with the applied stimulus profile, if a stimulation has been applied..



(1) Use this to select neurons, selected neurons are displayed in gray.

(2) Use this tool to turn the neurons

(3) Display of 3D neuronal geometry - Select a neuron name and

(4) Assign the neuron name using this menu

(5) When finished press the ‘assign’ button

(6) Correlated neurons can be displayed by selecting a neuron and clicking this button

(7) Activity of selected neurons is displayed here

(8) Comments concerning the activities or the worm can be entered here

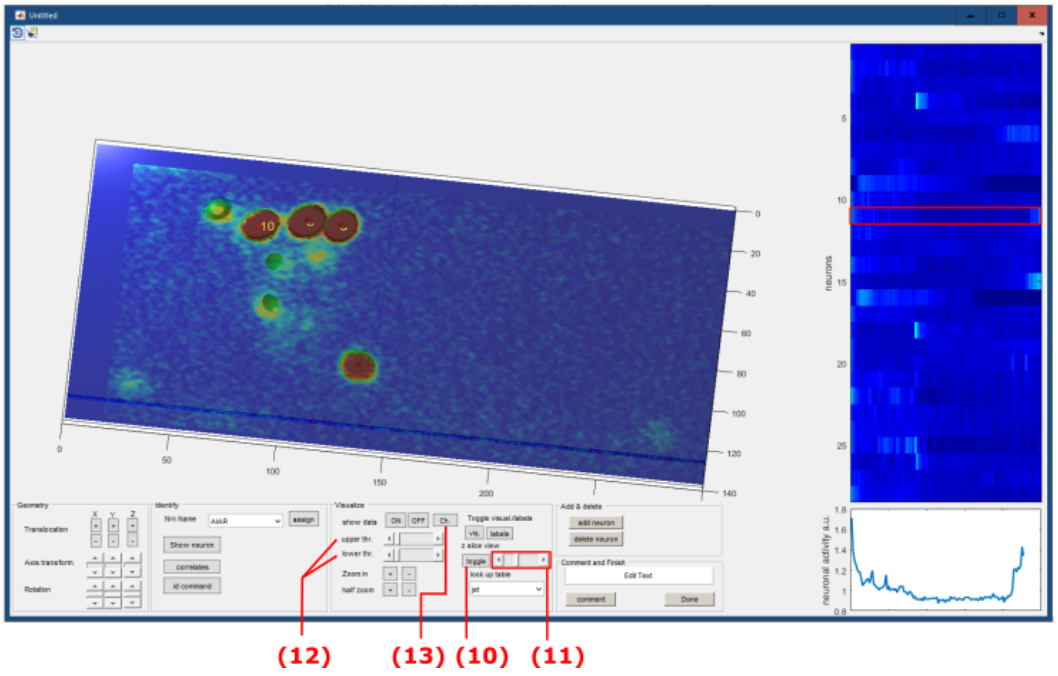
(9) This panel can be used to modify the geometry of the read-out zone when the gaussian fit is aberrant and the read-out zone is irregularly shaped.

(10) Geometry can be adjusted to fit the actual expression by overlaying the read-out geometry with the image data

(11) Use this slider to browse through the imaging data.

(12) Dynamic range of the data can be adjusted using these sliders

(13) Channels can be switched using this button.



Reference:

Toyoshima, Yu, Terumasa Tokunaga, Osamu Hirose, Manami Kanamori, Takayuki Teramoto, Moon Sun Jang, Sayuri Kuge, Takeshi Ishihara, Ryo Yoshida, and Yuichi Iino. 2016. “Accurate Automatic Detection of Densely Distributed Cell Nuclei in 3D Space.” *PLoS Computational Biology* 12 (6): e1004970.