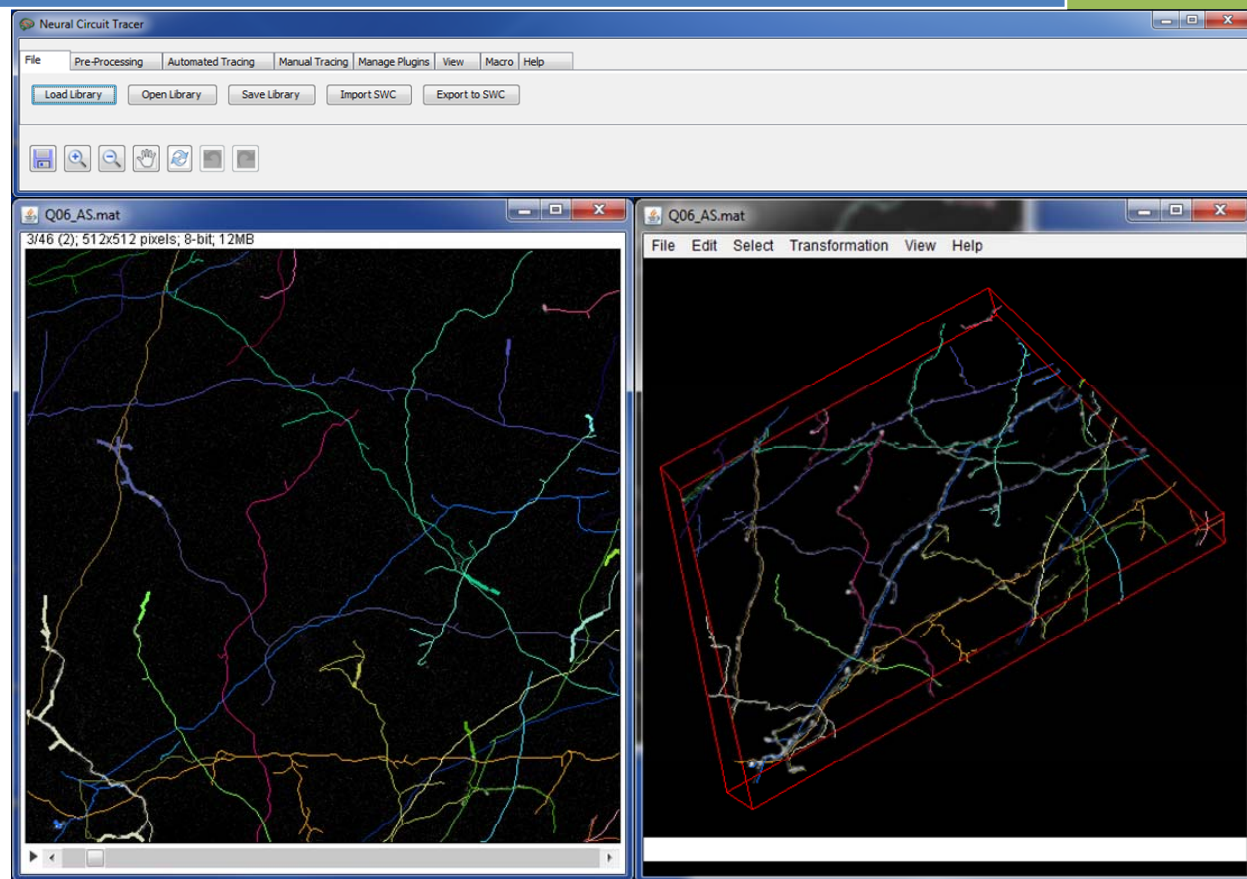


2012

Neural Circuit Tracer



Software for automated tracing of
neurites from light microscopy stacks
of images

User Guide
Version 1.0

Contents


1. Introduction.....	- 2 -
2. Installation and system requirements	- 2 -
3. File tab: loading, opening, and saving libraries	- 2 -
4. Pre-Processing tab: MatLab and ImageJ plugins	- 4 -
5. Automated Tracing tab: initial trace, automated branch merger, and online training	- 5 -
6. Manual Tracing tab: trace editing, optimization, and finding loops.....	- 8 -
7. Manage Plugins tab: adding and removing MatLab and ImageJ plugins.....	- 10 -
8. View tab: image stack, z-projection, 3D view, and console	- 11 -
9. Macro tab: creating and running macros	- 13 -
10. Help tab: user manual and website.....	- 14 -
11. Examples.....	- 14 -
12. About NCTracer	- 15 -

1. Introduction

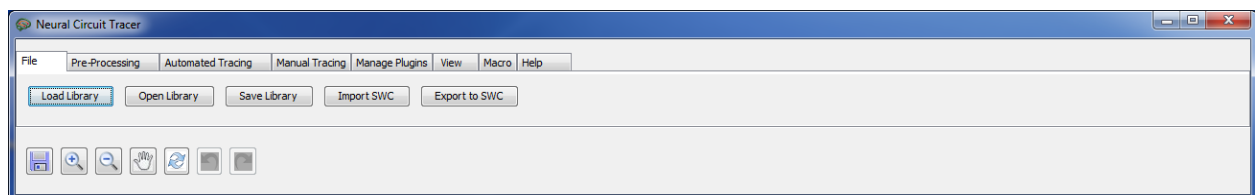
Neural Circuit Tracer (NCTracer) is open source software built using Java and MatLab. The software is based on the core of ImageJ (<http://rsbweb.nih.gov/ij>) and the graphic user interface was developed in Java Swings. NCTracer combines a number of ImageJ functions with several newly developed algorithms for automated and manual tracing of neurites. The software is designed in a way that allows the user to add Java and MatLab based plugins, build and run macros. These and other functionalities of NCTracer are described in detail below.

2. Installation and system requirements

NCTracer 1.0 is developed for the Windows 7, 64-bit operating system and requires a minimum of 4 GB of RAM. *This version does not run on 32-bit computers, Mac or Linux OS.*

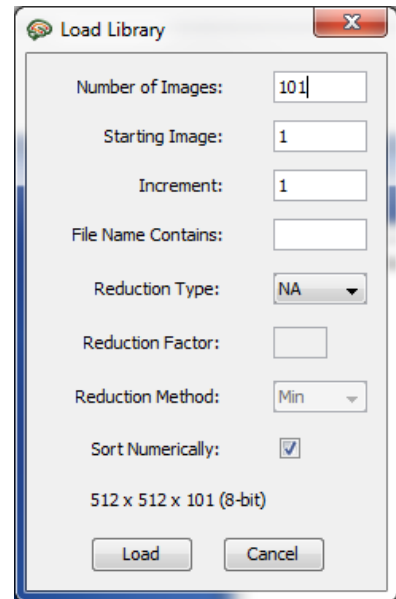
To install NCTracer download NCTracer_1_0.zip located at <http://www.neurogeometry.net/resources/tools> and run the Setup.exe file. This will install four components: Java (jre-7-windows-x64), Java3D (java3d-1_5_2-windows-amd64), MatLab Runtime Compiler (MCR_R2012a_win64), and NCTracer (NCT_1_0). It is recommended that you install all four components even if some of these components are already present in your computer. **To run NCTracer** double-click on the NCTracer icon  created on your desktop.


3. File tab: loading, opening, and saving libraries



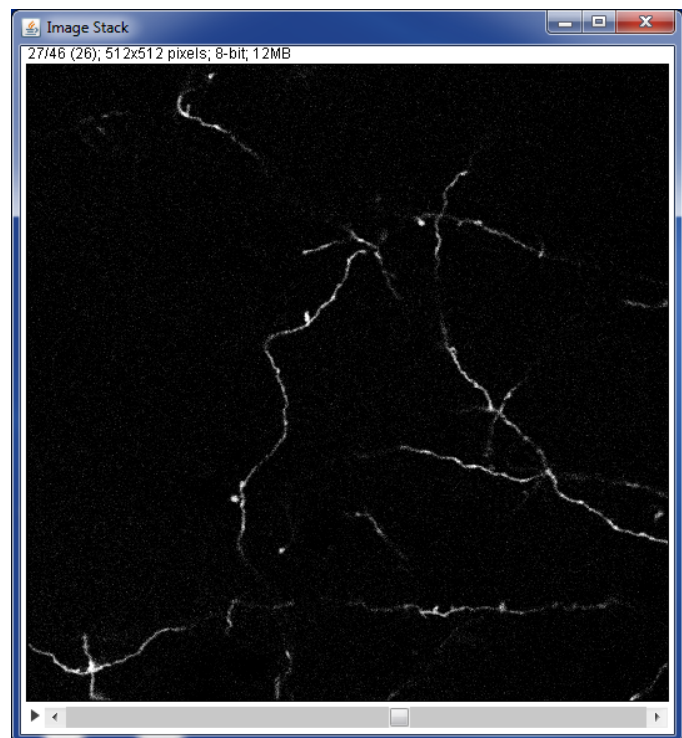
The current version of NCTracer supports image stacks saved in several standard image file formats, as well as MatLab data file format (.mat). Trace must be saved in .mat or SWC format of neuron morphology.

To load a new image stack click on the Load Library button located in the File tab of the main NCTracer window. After selecting an image from the stack of images a new Load Library dialog box (right) will appear. The user may select the images to be loaded by providing a starting image number, an increment, and a characteristic string contained in the file names. Images can be reduced during loading. Reduction type NA produces no reduction, while xy and xyz reduction types reduce the stack in the respective dimensions. The user must specify the reduction factor (positive double) and choose one of five reduction methods: Min, Max, Mean, Median, and SD. The loaded stack of images will be displayed in the Image Stack window (below). This window allows the user to scroll through the individual image planes within the stack.



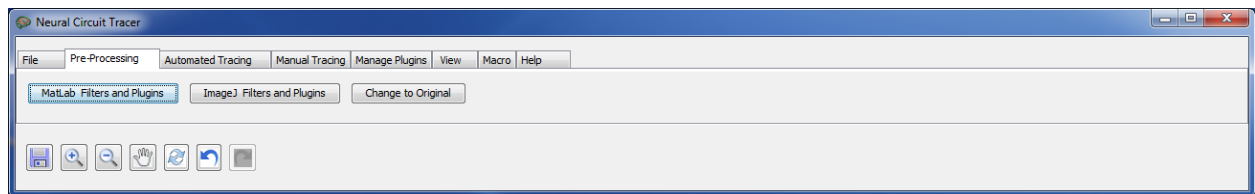
The original image stack, the pre-processed image stack, the trace, and the parameter file are collectively referred to as the Library. The **Save Library** button located in the File tab or  icon located in the main NCTracer window saves these components in the MatLab (.mat) data file format.

To open a library click on the Open Library button located in the File tab of the main NCTracer window. The user can **import / export the trace** of neurites by clicking on the Import SWC / Export to SWC buttons. The entire trace as well as



individual trees contained in the trace can be exported. In the latter case follow the instructions given in the Manual Tracing tab section. Import / export features only work with the SWC format of neuron morphology.



4. Pre-Processing tab: MatLab and ImageJ plugins



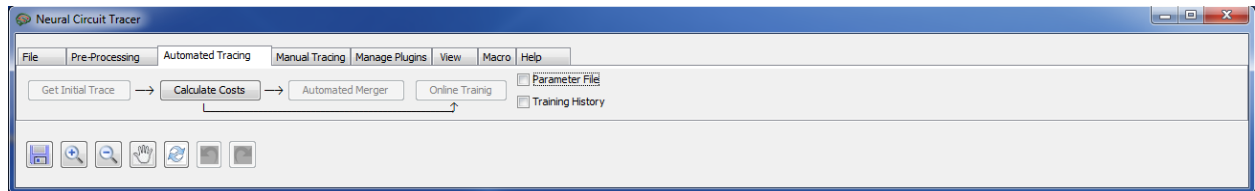
The Pre-Processing tab contains a number of custom filters written in MatLab and plugins adopted from ImageJ. These functions can be used to enhance the intensity of neurites and eliminate the background prior to performing automated tracing. *Automated tracing will not work if the background intensity in the preprocessed stack is non-zero; automated tracing may be slow if the number of voxels in the foreground is much higher than 10^5 .*

Current version of NCTracer includes the following preprocessing functions:

MatLab plugins	ImageJ plugins		
Eliminate Small Regions	Auto Gamma	Gaussian Blur	RATS
Multi-scale Gaussian	CLAHE	Hessian CP Measures	Rolling Ball Background
Multi-scale LoG	Contrast & Brightness Adjuster	Hybrid 3D Median Filter	Sigma Filter Plus
Reduce Background	Convolver	Image Stabilizer	Threshold Adjuster
Simple Filters	FeatureJ Filters	Mean Shift	Unsharp Mask
Threshold Background	FFT Bandpass Filter	Rank Filters	Variance Filter

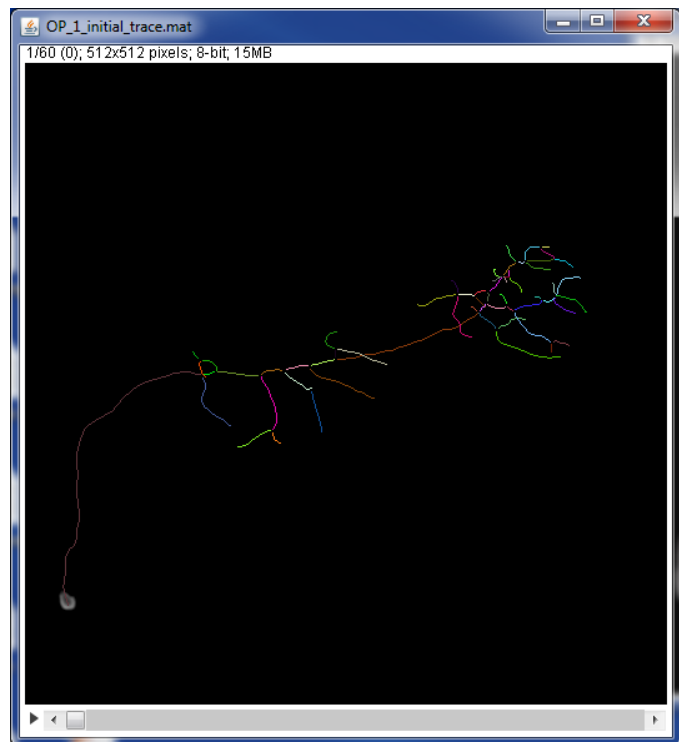
All preprocessing operations can be undone and redone using  and  buttons. New plugins can be created and the existing plugins can be removed in the Manage Plugins tab.

5. Automated Tracing tab: initial trace, automated branch merger, and online training



The main steps of the automated tracing algorithm are described in detail in [2, 4]. Briefly, they include the following components:

- **Preprocessing:** Typically, the image stack is filtered to smooth the intensity along the neurites and sharpen the transition between the neurites and the background. Next, image is binarized by setting an adoptive threshold. Both the binary and the original intensity images are used in subsequent calculations (also see the Pre-Processing tab section).
- **Initial trace:** The centerline of all neurites contained in the image is detected with the voxel-coding algorithm [1]. Erroneous short terminal branches and small loop clusters are eliminated. The resulting trace is optimized by using methods described in [2, 4]. Finally, the branches are disconnected from one another. The result is referred to as the initial trace (right).
- **Calculation of cost components:** Various cost components, reflecting the likelihoods of different branch merging scenarios, are calculated at this stage (see [2, 4] for details).
- **Automated branch merger:** Branches in the initial trace are merged into tree structures. All branch merging scenarios are considered at this step of the algorithm, and the scenario with the lowest overall cost of merger is implemented. Weights of individual cost components are determined with online training. *Complex mergers, involving more than 12 branch end-points, are not carried out in this version of the software.*



To perform automated tracing, the user must create a preprocessed image stack and load Parameter and Training History files. Click on the Get Initial Trace button located in the Automated Tracing tab of the main NCTracer window. The initial trace will be generated and displayed in the Image Stack window. If necessary, the user may edit the initial trace by switching to the Manual Tracing tab. Click on the Calculate Costs button to generate cost components for all branch merging scenarios. Finally, click on the Automated Merger button. The initial trace will be merged into tree structures and redisplayed. The above steps can be combined into a macro and run with no user involvement (see the Macros tab section). Progression of the automated tracing algorithm can be monitored in the Console window. Automated tracing time depends on the complexity of neural trees and the number of foreground voxels.

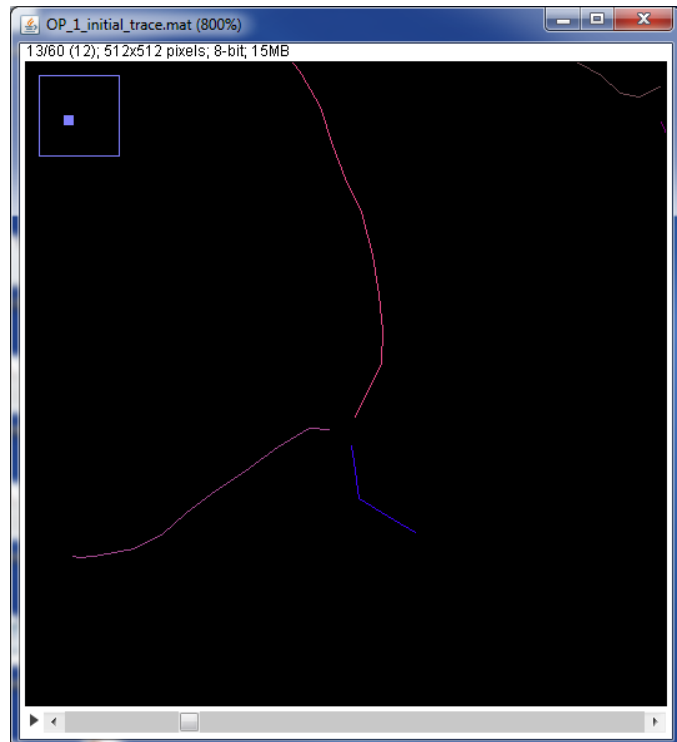
Training history files can be created in the online training module of the software. These files can be generalized on image stacks obtained under similar experimental conditions (same class of neurites and similar labeling / imaging techniques). **To create a training history file** load the image stack, obtain its initial trace, and calculate the cost components (as described above). Clicking on the Online Training button will open the Online Training table:

Cluster Number	No. of End Points	Scenario	Weight	Confidence
1	2	- 1 +	0.0	0.5
2	3	- 1 +	0.0	0.2
3	3	- 1 +	0.0	0.2
4	3	- 1 +	0.0	0.2
5	3	- 1 +	0.0	0.2
6	3	- 1 +	0.0	0.2
7	3	- 1 +	0.0	0.2
8	3	- 1 +	0.0	0.2
9	3	- 3 +	1.0	0.2
10	3	- 1 +	0.0	0.2
11	3	- 3 +	0.0	0.2
12	3	- 4 +	0.0	0.2
13	3	- 5 +	0.0	0.2
14	3	- 1 +	0.0	0.2
15	4	- 1 +	0.0	0.07
16	4	- 1 +	0.0	0.1
17	4	- 1 +	0.0	0.14
18	4	- 1 +	0.0	0.1

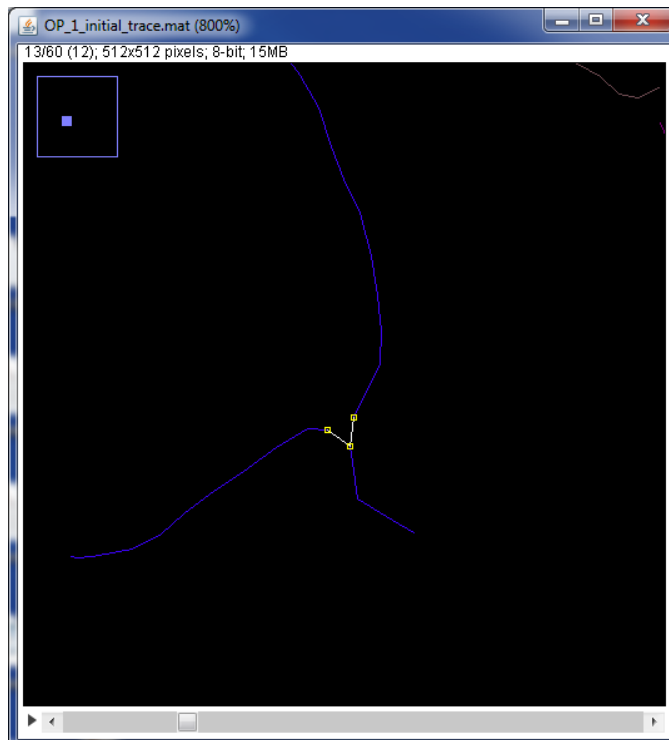
☒ WPerceptron
☐ WSVM

Load Training History Apply Save Training History Cancel

The above table shows branch merging scenarios for different clusters of branch points (rows of the table). One such cluster is shown on the right. Clicking on different rows of the table or right-clicking on different parts of the initial trace in the Image Stack window will move the focus (squares) from one cluster to another. To view different branch merging scenarios within the selected cluster (bottom image) click on + (next scenario) or - (previous scenario) signs, or select a scenario number from the dropdown window in the table.



Once the correct scenario is selected, assign a positive weight (typically 1) to it. Weighted scenarios will be highlighted in yellow (see the table above). After such training on several clusters of branch points select WPerceptron or WSVM classifier and click on the Apply button to generate the Training History. Correctly classified branch mergers will be highlighted with green in the table, while erroneous mergers will be highlighted in red.

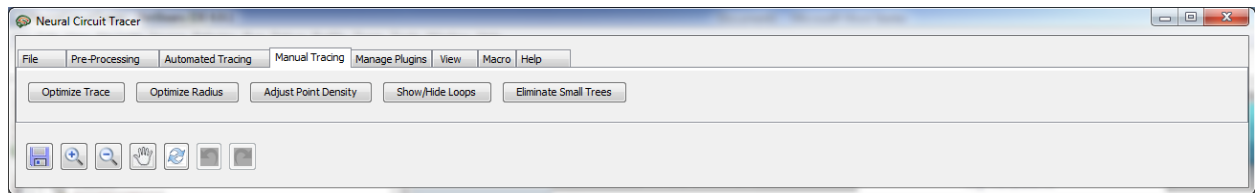


Erroneous mergers need to be corrected or their weights must be reduced to zero to ensure that the classifiers are not trained on errors.

The user may continue training or save training history and move to another training stack. New training examples will be appended to the old training history as long as the latter is loaded (check the loaded Training History path in the Automated Tracing tab). Training on about 100 clusters of branch point is typically sufficient to achieve error rates of less than 5% [4]. *It is recommended not to use ambiguous*

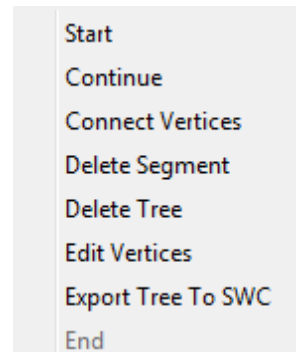
examples during training. Branch mergers that are open to doubt should not be included in training.

6. Manual Tracing tab: trace editing, optimization, and finding loops






The Manual Tracing tab makes it possible to trace neurites manually and/or edit automated traces. Right-click over the Image Stack window to open the manual tracing menu (right) and enable the following functionalities:

- Select **Start** and click at a desired location in the Image Stack window to initiate tracing of a new tree structure. Continue clicking within the Image Stack window to add vertices. Each new vertex will be connected to the previous vertex with a straight-line segment.
- Tracing can be continued from any vertex of the tree. Select **Continue**, place the cursor in the vicinity of the vertex, and click on the vertex to activate it.
- **Connect Vertices** makes it possible to connect any two vertices of the trace with a segment. The user will be prompted if this operation results in the formation of a loop.
- Individual segments of the trace can be eliminated by selecting the **Delete Segment** function, and moving the cursor to the vicinity of the segment's vertices. Clicking on one of these vertices will activate its neighbors. A subsequent click on one of these neighboring vertices will eliminate the connecting segment.
- **Delete Tree**, followed by mouse over the tree vertices will highlight the entire tree. A subsequent click on one of the vertices will result in the deletion of the tree.
- Individual vertices of the trace can be moved in *x*, *y*, and *z* directions. Select **Edit Vertices** function and click and drag the vertex in the selected plane. To move the vertex to a different plane, scroll through the stack to the desired *z*-location and click on the vertex.

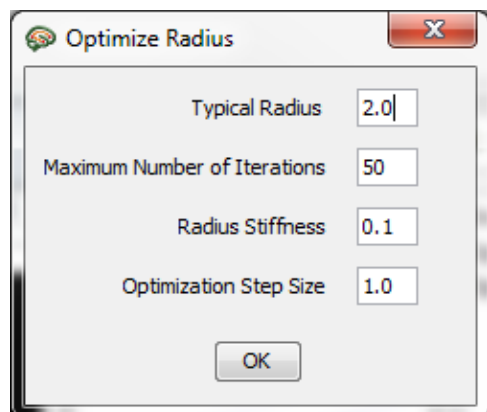
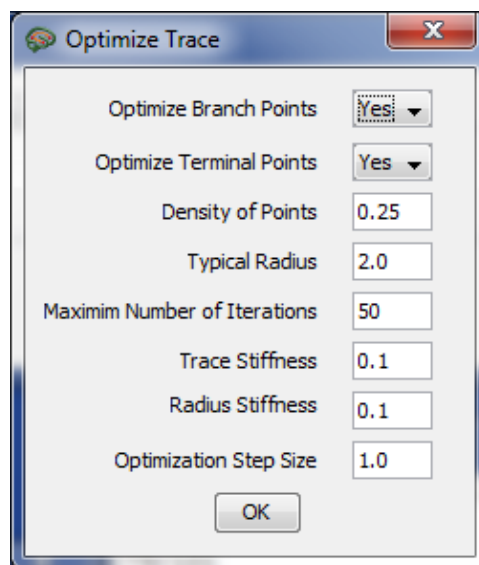


- To export a single tree in the SWC format the user may select **Export Tree to SWC** and move the mouse over the tree of interest to highlight it. A subsequent click on any of the vertices of the tree will initiate the export process. To export the entire trace the user must click on the **Export to SWC** button located in the File tab of the main NCTracer window.
- **End** terminates the previously selected action and disables active vertices.

During manual tracing, the user can scroll through the image planes of the stack (by using the slide bar at the bottom of the Image Stack window or the mouse wheel), zoom-in, zoom-out (with Page Up / Down), and pan in the plane of the stack (with arrow keys). Manual tracing operations can be undone / redone by pressing Ctrl+Z / Ctrl+Y or by clicking on / icons located at the bottom of the main NCTracer window. The user must be in the Manual Tracing tab to perform undo / redo operations on the trace.

The intensity and xyz coordinates of an image voxel at the position of the cursor, as well as the caliber (radius) and xyz coordinates of the trace at that location are displayed in the main NCTracer window. Trace segments whose vertices are located in the current plane of view are indicated with thick lines. Trace can be switched on and off, viewed on z-projection and in 3D (see the View tab section). These views can be refreshed by clicking on  icon.

The trace can be optimized with two optimization functions located in the Manual Tracing tab of the main NCTracer window (details of these optimization



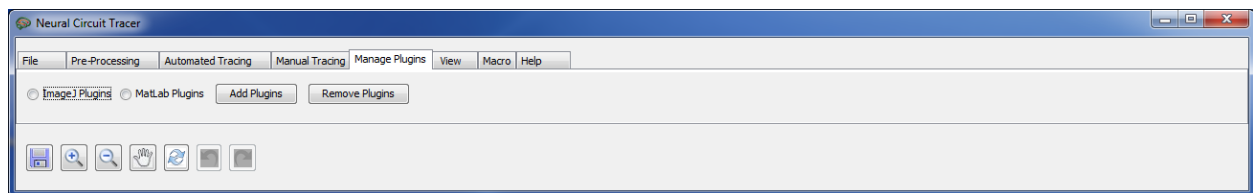
procedures are described in [2, 4]). The **Optimize Trace** function (right) optimizes both the positions and calibers of vertices in the trace. Branch and terminal points can be kept fixed during the optimization. Default values of optimization parameters are displayed in the Optimize Trace dialog box. The **Optimize Radius** function (left) calculates the optimal calibers of trace vertices, while the vertex positions remain unchanged.

The average density (number / length) of trace vertices can be adjusted with the **Adjust Point Density** function located in the Manual Tracing tab of the main NCTracer window.

The **Eliminate Small Trees** button allows the user to automatically delete trees that are shorter than a user defined length threshold.

Loops (if any) in the trace can be displayed by pressing the **Show / Hide Loops** button.

7. Manage Plugins tab: adding and removing MatLab and ImageJ plugins



NCTracer allows the user to import plugins designed for ImageJ. **To import an ImageJ plugin** check the ImageJ Plugins radio button and click on Add Plugins. The user will need to find and open the main file (.jar, .java, or .class) associated with the plugin. The plugin will be copied into the NCTracer\plugins folder, and will automatically appear in the list of plugins in the Pre-Processing tab.

Functions written in MatLab can also be converted into NCTracer plugins. In this case the user must first employ MatLab Builder JA (<http://www.mathworks.com/products/javabuilder/>) to create a .jar file associated with the MatLab function. Once the .jar file is created it can be imported into NCTracer by checking the MatLab Plugins radio button and clicking on Add Plugins. This will open the Input / Output Selection window (below), in which the user must specify the name, type, order, and default value of every input and output variable of the MatLab function.

There are four variables which must be formatted in a specific way to work with NCTracer. These variables include: Image Stack (3D MatLab array), Adjacency Matrix ($N \times N$ array of doubles), Position Vector ($N \times 3$ array of doubles), and Branch Caliber ($N \times 1$ array of doubles). N here denotes the number of vertices in the trace. If some of these four variables are not in use, remove them from the Input / Output Selection window by un-checking the corresponding check-boxes. Additional input / output parameters may be included by specifying the number of these parameters in the Input / Output Selection window.

Multi_Thresholder

Input Parameters

Number of Parameters: 2 Generate Parameters

☒ Image Stack ☒ Adjacency Matrix ☒ Position Vector ☐ Branch Caliber

Parameter Name	Parameter Type	Default Value	Order
Image Stack	Array		1
Adjacency Matrix	Array		2
Position Vector	Array		3
			4
			5

Output Parameters

Number of Parameters: 1 Generate Parameters

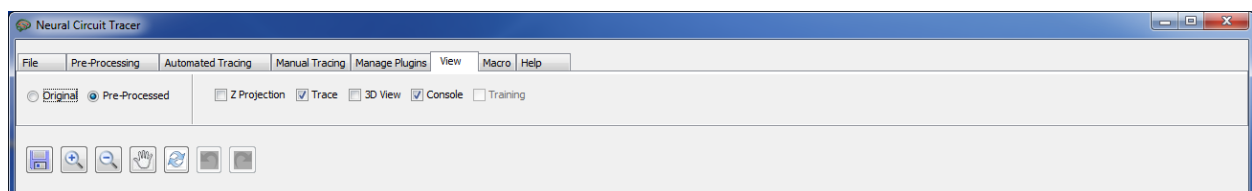
☒ Image Stack ☒ Adjacency Matrix ☒ Position Vector ☐ Branch Caliber


Parameter Name	Parameter Type	Order
Image Stack	Array	1
Adjacency Matrix	Array	2
Position Vector	Array	3
		4

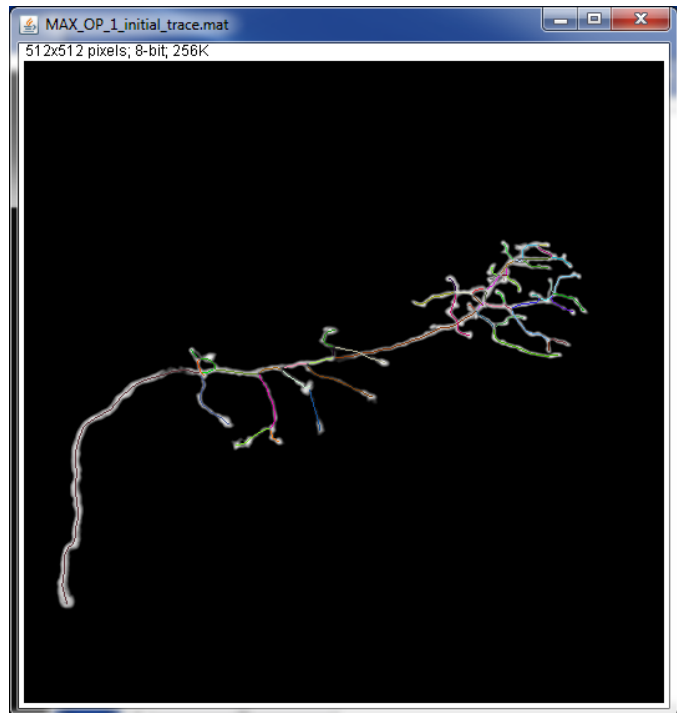
OK




Any plugin can be removed from NCTracer by selecting the plugin type (Image) or MatLab) and clicking on the Remove Plugins button.

8. View tab: image stack, z-projection, 3D view, and console



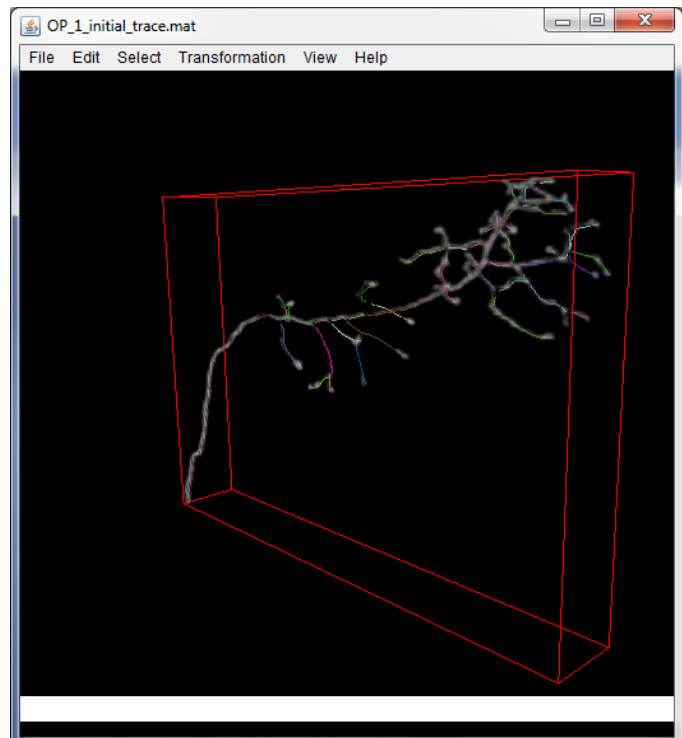
The View tab of the main NCTracer window allows the user to display the stack on a z-projection and in 3D. The user can switch between the **Original** and **Pre-processed** image stacks (radio buttons), as well as **display / hide the trace** (Trace check box). The user can scroll through the individual image planes with the slide bar located at the bottom of the Image Stack window or by using the mouse wheel. All views can be refreshed by clicking on  button in the main NCTracer window.



All three views are equipped with **zoom-in, zoom-out, and pan features**. In the Image Stack and Z Projection windows these features can be accessed by clicking on , , or  buttons at the bottom of the main NCTracer window and then clicking on the image or dragging the image with the mouse. Alternatively, these functions can be controlled with Page Up / Down and the arrow keys.

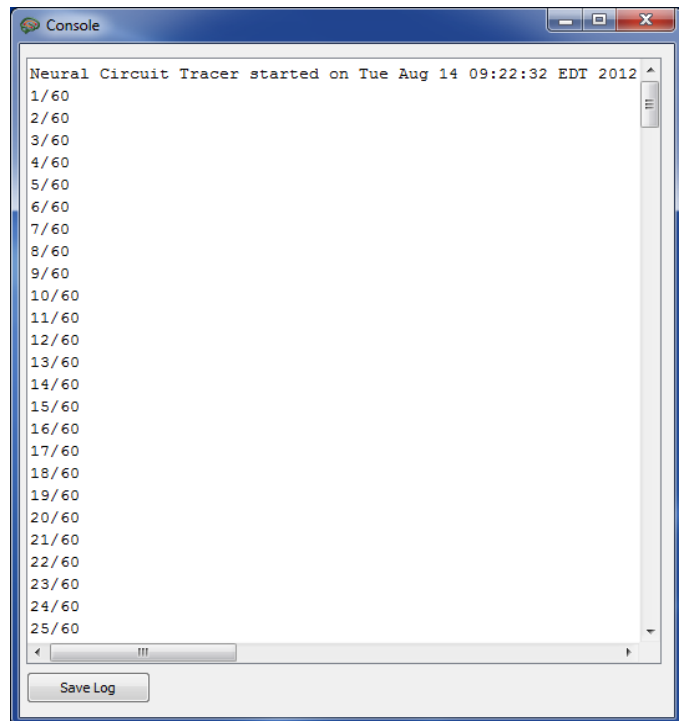
The user can determine the xyz coordinates and the intensity of any voxel in the Image Stack window or any pixel in the Z Projection window by pointing the cursor at that location. The information will be displayed in the main NCTracer window. Positions and calibers of the trace vertices can be displayed in a similar manner.

In the 3D View window (<http://rsb.info.nih.gov/ij/plugins/3d-viewer/>) zoom-in / zoom-out features are controlled with Page Up / Page Down or the mouse wheel. Panning can

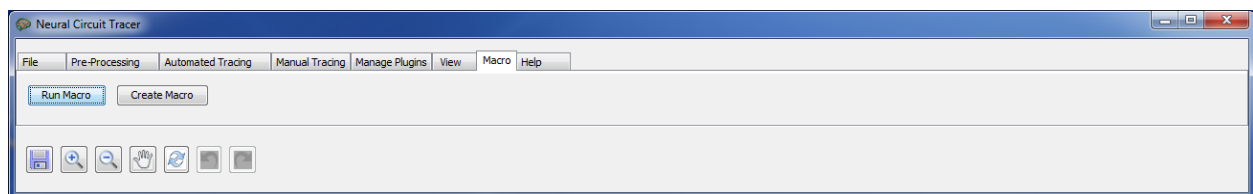


be done by pressing the arrow keys while holding down Shift. The image stack can be rotated in 3D by using the mouse or the arrow keys.

The View tab also displays the Console window (right) and the Online Training table. The former provides a log of all operations performed by the user and the software. The latter is used to create training history (see the Automated Tracing tab section).

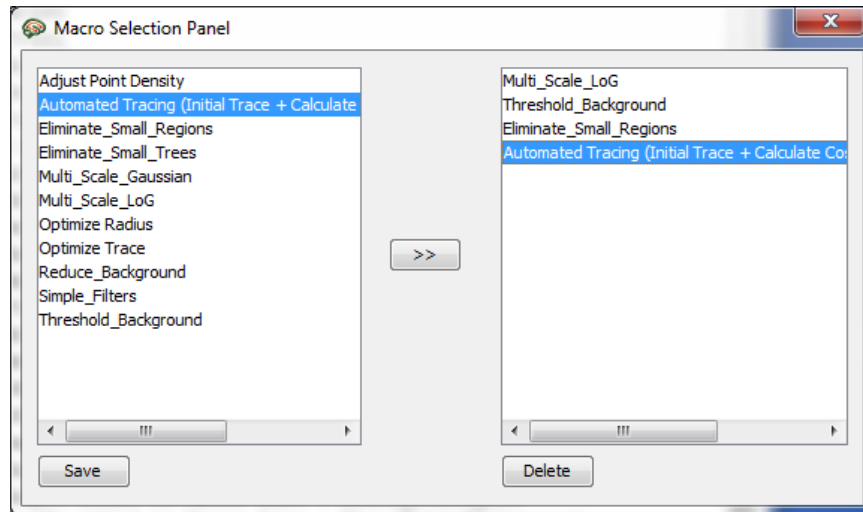


9. Macro tab: creating and running macros

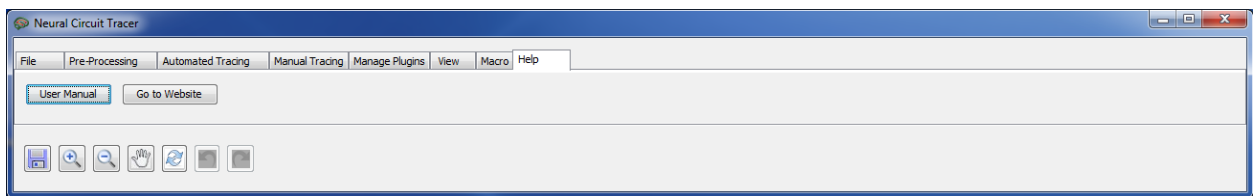


The user can create, save, and run sequences of operations (macros) to be performed on image stacks. **To create a macro**, click on the Macro tab in the main NCTracer window and assemble the desired macro components in the Macro Selection panel (below).

You will be prompted to provide the parameters for every function included in the macro. Click on Save to create a text file containing the macro sequence. **To run a macro** click on the Run Macro button and select the macro file. *ImageJ plugins cannot be included in a macro in this version of the software.*



10. Help tab: user manual and website



This tab contains the links to this User Manual and the Neurogeometry group website, <http://www.neurogeometry.net>.

11. Examples

This section provides examples of automated tracing with NCTracer 1.0. The two datasets used in this section (and in illustrations throughout this manual) were obtained from <http://www.diademchallenge.org/>.

Neocortical layer 6 axons

File tab

- Load an image stack. No reduction is necessary.

Pre-Processing tab

- Apply Multi-scale LoG plugin: Minimum Radius = 2, Step Size = 1, Minimum Radius = 2.
- Use Threshold Background plugin: Threshold = 10 to 25
- Use Eliminate Small Regions plugin: Minimum Region Size = 100.

Automated Tracing tab

- Load Parameter File: NCTracer/parameter-files/Parameters_L6.txt
- Load Training History: NCTracer/training-history/TrainingHistory_L6.mat
- Click on Get Initial Trace.
- Click on Calculate Costs.
- Click on Automated Merger.

Olfactory projection fiber

File tab

- Load an image stack. No reduction is necessary.

Pre-Processing tab

- Apply Multi-scale LoG plugin: Minimum Radius = 2, Step Size = 1, Minimum Radius = 4.
- Use Threshold Background plugin: Threshold = 5 to 10
- Use Eliminate Small Regions plugin: Minimum Region Size = 100.

Automated Tracing tab

- Load Parameter File: NCTracer/parameter-files/Parameters_OP.txt
- Load Training History: NCTracer/training-history/TrainingHistory_OP.mat
- Click on Get Initial Trace.
- Click on Calculate Costs.
- Click on Automated Merger.

The above tracing sequences may be written into macros (see the Macros tab section) and ran without any input from the user. Automated tracing of individual stacks from the above two datasets may take several minutes. The results can be edited in the Manual Tracing tab, saved in .mat format, and exported as .swc files.

12. About NCTracer

Contact information

NCTracer is being developed by the [Neurogeometry](#) group at the Department of Physics and the Center for Interdisciplinary Research on Complex Systems at Northeastern University in Boston. All inquiries should be addressed to Prof. Armen Stepanyants, a.stepanyants@neu.edu.

Support

NCTracer is supported by the NIH/NINDS grant NS063494.

Related publications

1. Vasilkoski Z. and Stepanyants A., Detection of the optimal neuron traces in confocal microscopy images, J Neurosci Methods, 178: 197-204 (2009)
2. Chothani, P., Mehta, V., and Stepanyants A., Automated tracing of neurites from light microscopy stacks of images, Neuroinformatics, 9(2-3): 263–278 (2011)
3. Mukherjee, A. and Stepanyants, A., Automated reconstruction of neural trees using front re-initialization, SPIE Medical Imaging, 8314 (2012)

Significant contributions

Graduate students Chintan Bhavsar, Paarth Chothani, and Vivek Mehta developed the Graphic User Interface of NCTracer 1.0. Graduate students Julio Chapeton and Rohan Gala tested and debugged the software. Prof. Armen Stepanyants developed and implemented the algorithms in MatLab.