Research Article: Methods/New Tools | Novel Tools and Methods

neuTube 1.0: a New Design for Efficient Neuron Reconstruction Software Based on the SWC Format123

neuTube: Efficient Neuron Reconstruction Software

Linqing Feng[1,*], Ting Zhao[2,*] and Jinhyun Kim[1,3]

[1]*Center for Functional Connectomics, Korea Institute of Science and Technology (KIST), Seoul, 136-791, Korea*
[2]*Janelia Research Campus, Howard Hughes Medical Institute, Ashburn, Virginia, 20147, United States*
[3]*Neuroscience Program, University of Science and Technology, Daejeon, 305-350, Korea*

DOI: 10.1523/ENEURO.0049-14.2014

Received: 30 October 2014

Revised: 23 December 2014

Accepted: 23 December 2014

Published: 2 January 2015

Author Contributions: L.F., T.Z., and J.K. designed research; L.F. and T.Z. performed research; L.F. and T.Z. analyzed data; L.F., T.Z., and J.K. wrote the paper.

Funding: KIST Institutional Program
Project No. 2E24210

Funding: WCI Program
NRF Grant Number: WCI 2009-003

Funding: Howard Hughes Medical Institute

Conflict of Interest: Authors report no conflict of interest.

This work was supported by the KIST Institutional Program (Project No. 2E24210) and WCI Program (NRF Grant Number: WCI 2009-003). Ting Zhao is supported by Howard Hughes Medical Institute.

Correspondence should be addressed to: Dr. Ting Zhao, Janelia Research Campus, Howard Hughes Medical Institute, 19700 Helix Drive, Ashburn, VA 20147, United States (zhaot@janelia.hhmi.org).) and Dr. Jinhyun Kim, Center for Functional Connectomics, Korea Institute of Science and Technology (KIST), 39-1 Hawolgokdong, Seoul, 136-791, Korea (kimj@kist.re.kr).)

Cite as: eNeuro 2015; 10.1523/ENEURO.0049-14.2014 eNeuro

http://eneuro.msubmit.net

eN-TMNT-0049-14R1

neuTube 1.0: A New Design for Efficient Neuron Reconstruction Software Based on the SWC Format

Manuscript Title Page

1. Manuscript Title (50 word maximum)

neuTube 1.0: A New Design for Efficient Neuron Reconstruction Software Based on the SWC Format

2. Abbreviated Title (50 character maximum)

neuTube: Efficient Neuron Reconstruction Software

3. List all Author Names and Affiliations in order as they would appear in the published article

Linqing Feng, Center for Functional Connectomics, Korea Institute of Science and Technology (KIST), Seoul, 136-791, Korea

Ting Zhao, Janelia Research Campus, Howard Hughes Medical Institute, Ashburn, Virginia, 20147, United States

Jinhyun Kim, Center for Functional Connectomics, Korea Institute of Science and Technology (KIST), Seoul, 136-791, Korea and Neuroscience program, University of Science and Technology, Daejeon, 305-350, Korea

4. Author Contributions: Each author must be identified with at least one of the following: Designed research, Performed research, Contributed unpublished reagents/ analytic tools, Analyzed data, Wrote the paper.

T.Z., L.F. and J.K. Designed Research; T.Z. and L.F. Performed Research; T.Z. and L.F. analyzed data; T.Z., L.F. and J.K. Wrote the paper.

L.F. and T.Z. contributed equally to this work.

5. Correspondence should be addressed to (include email address)

Dr. Ting Zhao, Janelia Research Campus, Howard Hughes Medical Institute, 19700 Helix Drive, Ashburn, VA 20147, United States (zhaot@janelia.hhmi.org) and Dr. Jinhyun Kim, Center for Functional Connectomics, Korea Institute of Science and Technology (KIST), 39-1 Hawolgokdong, Seoul, 136-791, Korea (kimj@kist.re.kr)

6. Number of Figures: 6

7. Number of Tables: 1

8. Number of Multimedia: 0

9. Number of words for Abstract: 170

10. Number of words for Significance Statement: 62

11. Number of words for Introduction: 412

12. Number of words for Discussion: 521

13. Acknowledgements

The authors would like to thank Camden Markham for proof reading the manuscript and thank Osung Kwon and Bokyoung Lee for testing the software.

14. Conflict of Interest A. No (State 'Authors report no conflict of interest') B. Yes (Please explain)

A

15. Funding sources

This work was supported by the KIST Institutional Program (Project No. 2E24210) and WCI Program (NRF Grant Number: WCI 2009-003). Ting Zhao is supported by Howard Hughes Medical Institute.

Abstract

Brain circuit mapping requires digital reconstruction of neuronal morphologies in complicated networks. Despite recent advances in automatic algorithms, reconstruction of neuronal structures is still a bottleneck in circuit mapping due to a lack of appropriate software for both efficient reconstruction and user-friendly editing. Here we present a new software design based on the SWC format, a standardized neuromorphometric format that has been widely used for analyzing neuronal morphologies or sharing neuron reconstructions via online archives such as NeuroMorpho.org. We have also implemented the design in our open-source software called neuTube 1.0. As specified by the design, the software is equipped with parallel 2D and 3D visualization and intuitive neuron tracing/editing functions, allowing the user to efficiently reconstruct neurons from fluorescence image data and edit standard neuron structure files produced by any other reconstruction software. We show the advantages of neuTube 1.0 by comparing it to two other software tools, namely Neuromantic and Neurostudio. The software is available for free at http://www.neutracing.com, which also hosts complete software documentation and video tutorials.

Significance Statement

Compared to other existing tools, the novel software we present has some unique features such as comprehensive editing functions and the combination of seed-based tracing and path searching algorithms, as well as their availability in parallel 2D and 3D visualization. These features allow the user to reconstruct neuronal morphology efficiently in a comfortable ‘What You See Is What You Get’ (WYSIWYG) way.

1. Introduction

Digital reconstruction, or tracing, of neuron morphologies from light microscope images is an important step in the mapping of brain circuits. In this task, the input is images and the output is usually a tree structure, which can be described by the SWC file format (Cannon et al., 1998). Although numerous neuron reconstruction software tools have been developed for producing SWC files (Meijering, 2010), none of them has taken full advantage of the SWC format to optimize the user interface for efficient and accurate reconstruction. An optimal user interface means that the user can interact with the software with minimal cognitive load, which requires data visualization to be clear and operations to be straightforward. In other words, with the visual information provided by the software, the user should be able to quickly figure out the underlying SWC model, how the model can be manipulated, and the results of manipulations. With these criteria in mind, we may identify disadvantages of many tracing software applications. For example, FARSIGHT (Luisi et al., 2011), which focuses on semi-automated reconstruction of neurons, does not provide intuitive low-level editing options to correct subtle errors. Simple Neurite Tracer, a popular plugin of Fiji (Longair et al., 2011), also lacks editing functions. Neurolucida, a mainstream commercial software tool, allows complete manual reconstruction of a neuron structure. However, it does not support neuron reconstruction in a 3D visualization window, despite the fact that 3D interaction has been demonstrated to improve both the speed and accuracy of the reconstruction procedure (Long et al., 2012). Some other popular software tools, such as Neuromantic (Myatt et al., 2012) and Neurostudio (Wearne et al., 2005), similarly lack advanced 3D editing functions. On the other hand, Vaa3D (Peng et al., 2010) provides innovative interactive neuron tracing functions in 3D, but these functions are not available in 2D to resolve dense or faint structures.

Here, we propose a new and comprehensive software design based on the SWC format as a solution to the diverse limitations of current tracing software. Based on this design, or, as we call it, the SWC framework, we have converted our previously reported software, neuTube (Kim et al., 2012), into a novel tool that enables efficient reconstruction by combining robust automatic tracing algorithms and versatile user-friendly editing functions in both 2D and 3D. This paper formally presents the redesigned software, neuTube 1.0, not only as a major upgrade of the previous release, but also as the first software to implement the SWC framework.

2. Materials and Methods

2.1 The SWC framework

The overall layout of the SWC framework is shown in Figure 1. Software with this architecture takes a raw image or an SWC file as input and outputs a neuron structure satisfying the user. The design of the SWC framework follows the principle of 'What You See Is What You Get' (WYSIWYG) (Peng et al., 2011), *i.e.* what the user is editing is explicitly visualized and no third-party viewer is needed to check the results. Therefore, the SWC framework consists of the following features: clear visualization of SWC structures, clear visualization of source images as reference data, explicit definition of operation units and intuitive map from user inputs to editing operations. Except image visualization, these features are designed based on the SWC format, which describes a simple directed tree model, called the SWC model. Here, we described in detail how to construct operations on the SWC model by first defining the model in an abstract way.

2.1.1 Abstract Definition of The SWC Model

From a mathematical point of view, the SWC model can be defined as a set of nodes $\{\mathbf{n}_i = (x_i, y_i, z_i, r_i, \mathbf{n}_j) | i = 1, \dots, N, j = 0, \dots, N, i \neq j, x_i, y_i, z_i, r_i \in R\}$, where each node $\mathbf{n}_i$ is a sphere with the center (x_i, y_i, z_i) and the radius r_i. $\mathbf{n}_0$ is an empty node for defining the roots of a neuron structure, and $\mathbf{n}_j$ is called the parent of $\mathbf{n}_i$. An upstream path from $\mathbf{n}_i$ to $\mathbf{n}_j$ is an array of node $(\mathbf{n}_{k_1}, \dots \mathbf{n}_{k_n})$ where $\mathbf{n}_{k_{i+1}}$ is the parent of $\mathbf{n}_{k_i}$, $k_1 = i, k_n = j$. To form a valid tree structure of a neuron, no loop is allowed, *i.e.* there is at most one upstream path from one node to another. In this model, the basic structural unit is a node, which defines how we should design visualization and interactions.

2.1.2 SWC Operation

Assuming S_1 and S_2 are two sets of nodes, the operation of a neuron structure is defined as

$$f(S_1) = S_2$$

For example, $f(\{\mathbf{n}_1, \dots, \mathbf{n}_n\}) = \phi$, where ϕ denotes the empty set, defines a removal operation. However, some operations may result in a new node set that forms an invalid neuron structure. How to construct a valid operation depends on the data structure describing the model. In our framework we used a redundant tuple to store a node, which is $\mathbf{n} = \left(G(\mathbf{n}), P(\mathbf{n}), C(\mathbf{n}), S(\mathbf{n})\right)$, where $G(\mathbf{n}) = (x(\mathbf{n}), y(\mathbf{n}), z(\mathbf{n}), r(\mathbf{n}))$ defines that the node is located at $\left(x(\mathbf{n}), y(\mathbf{n}), z(\mathbf{n})\right)$ with radius $r(\mathbf{n})$, $P(\mathbf{n})$ is the *parent* node of $\mathbf{n}$, $C(\mathbf{n})$ is the first *child* of $\mathbf{n}$ and $S(\mathbf{n})$ is the next *sibling* of $\mathbf{n}$. A sibling of $\mathbf{n}$ shares the same parent with $\mathbf{n}$, i.e. $P(\mathbf{n}) = P(S(\mathbf{n}))$. The redundancy is designed to improve computational efficiency of visiting a node. For example, to query a child of a node, the program needs only to check its first child and traverse other children through the sibling link, while in a non-redundant representation where each node is only linked to its parent, the program may need to check every node in the tree.

Editing a node $\mathbf{n}$ is defined as changing the value of the corresponding tuple. We call any change on $G(\mathbf{n})$ a *geometrical operation* and any change on $P(\mathbf{n}), C(\mathbf{n})$or $S(\mathbf{n})$ a *structural operation*. While a geometrical operation is straightforward, a structural operation may cause invalid neuron structures. For example, changing $P(\mathbf{n})$ alone may break the rule that $P(C(\mathbf{n})) = \mathbf{n}$ and $P(\mathbf{n}) = P(S(\mathbf{n}))$. To avoid this problem, we construct SWC operations at three levels in terms of operation complexity. The first level consists of three elementary operations linking a node $\mathbf{n}$ to another node $\mathbf{n'}$, as defined as follows

$$f_p(\{\mathbf{n}\}|\mathbf{n}') = f_p(\{(G(\mathbf{n}), P(\mathbf{n}), C(\mathbf{n}), S(\mathbf{n}))\}|\mathbf{n}') = \{(G(\mathbf{n}), \mathbf{n}', C(\mathbf{n}), S(\mathbf{n}))\}$$

$$f_c(\{\mathbf{n}\}|\mathbf{n}') = f_c(\{(G(\mathbf{n}), P(\mathbf{n}), C(\mathbf{n}), S(\mathbf{n}))\}|\mathbf{n}') = \{(G(\mathbf{n}), P(\mathbf{n}), \mathbf{n}', S(\mathbf{n}))\}$$

$$f_s(\{\mathbf{n}\}|\mathbf{n}') = f_s(\{(G(\mathbf{n}), P(\mathbf{n}), C(\mathbf{n}), S(\mathbf{n}))\}|\mathbf{n}') = \{(G(\mathbf{n}), P(\mathbf{n}), C(\mathbf{n}), \mathbf{n}')\}$$

At this level, structure validity is not guaranteed.

The second level consists of simple valid operations. Assuming $F_{p_0}(\mathbf{n})$ is the operation of setting the parent of $\mathbf{n}$ to $\mathbf{n}_0$ (the empty node), if $C(P(\mathbf{n})) = \mathbf{n}$, i.e. $\mathbf{n}$ is the first child of its parent, then

$$F_{p_0}(\mathbf{n}) = \begin{cases} f_s(\{\mathbf{n}\}|\mathbf{n}_0) \circ f_p(\{\mathbf{n}\}|\mathbf{n}_0) \circ f_c(\{P(\mathbf{n})\}|S(\mathbf{n})), & C(P(\mathbf{n})) = \mathbf{n} \\ f_s(\{\mathbf{n}\}|\mathbf{n}_0) \circ f_p(\{\mathbf{n}\}|\mathbf{n}_0) \circ f_s(\{S^{-1}(\mathbf{n})\}|S(\mathbf{n})), & \text{Otherwise} \end{cases}$$

where $f \circ g$ denotes a composite operation and $S(S^{-1}(\mathbf{n})) = \mathbf{n}$. To define an operation on a single node more explicitly, $F_{p_0}(\mathbf{n})$ is defined as a function of a node instead of a node set without adding any ambiguity.

The operation of setting a parent is

$$F_p(\mathbf{n}|\mathbf{n}') = f_c(\{C(\mathbf{n}')\}|\mathbf{n}) \circ f_s(\{\mathbf{n}\}|C(\mathbf{n}')) \circ f_p(\{\mathbf{n}\}|\mathbf{n}') \circ F_{p_0}(\mathbf{n})$$

This operation also sets $\mathbf{n}$ as the first child of $\mathbf{n}'$. In principle, this operation is sufficient for building all other operations. But in practice, it is useful to define one more operation, for setting a sibling:

$$F_s(\mathbf{n}|\mathbf{n}') = f_s(\{\mathbf{n}\}|\mathbf{n}') \circ f_s(\{\mathbf{n}'\}|S(\mathbf{n})) \circ f_p(\{\mathbf{n}'\}|P(\mathbf{n})) \circ F_{p_0}(\mathbf{n}')$$

The third level is a set of composition operations, which include any operation composed of the operations from the second level. At this level, we categorize the operations into two types, morphology-dependent and morphology-independent. An operation is morphology-dependent if the result of the operation depends on the positions or sizes of the nodes; otherwise it is morphology-independent.

Decomposing an operation into elementary operations helps guarantee the validity of neuron structure manipulation, and more importantly, helps implement the undo/redo functionality on arbitrary operations. An undo operation requires inverting the corresponding operator, which can be complicated because of the consistency requirement. For example, the inverse operation of deleting multiple nodes would require recovery of all the neighbors of the nodes. Direct inference of such an inverse operation not only takes significant effort, but also leads to errors that can be difficult to track. After decomposing an operation into a sequence of elementary operations, we can construct the undo operation easily by reversing the sequence.

2.1.3 User Interaction

The fundamental function of tracing software is changing neuron morphology with user inputs, which are usually composed of mouse clicks and key inputs. Since we defined an operation as the mapping of one set of nodes to another set, user interaction starts with node selection, which requires two components, SWC visualization and user input response. High-quality visualization of a neuron might be the most important feature of successful neuron editing. The ability to view the structures clearly greatly reduces examination time needed to identify errors. It is also necessary to provide both 2D and 3D views because each provides unique advantages. For example, a 3D view is well suited for displaying a global structure; and a 2D view provides precise inference of dense local structures.

The most intuitive way to select a node is to move the mouse cursor to the node and then click. This requires mapping the screen cursor coordinates into the 3D SWC space. Multiple selections should also be supported to specify a set of nodes as the input of an operation. After selection, the user can trigger an operation with some input. So the operation becomes

$$f(S_1|\Theta) = S_2$$

where Θ is the set of parameters supplied from user input. For example, $f(\{\mathbf{n}\}|(x,y,z)) = \{\mathbf{n}, F_p(((x,y,z,r(\mathbf{n})),\mathbf{n}_0,\mathbf{n}_0,\mathbf{n}_0)|\mathbf{n})\}$ defines an operation of extending a branch from $\mathbf{n}$ to a node at (x,y,z).

2.1.4 Create SWC Nodes from Image Signal

For any standalone neuron-tracing software, it is essential to allow reconstructing neuron structure from raw image signals. In the SWC framework, this function can be formulated as

$$g(\mathrm{S}_1|\Theta,\mathrm{I}) = \mathrm{S}_2$$

where I is the image signal. Note that this actually defines a superfamily of SWC operations. The function is the same as an SWC operation if it is independent of I. An example of an image-dependent operation is shortest path creation, such as the one used by Simple Neurite Tracer (Longair et al., 2011), where $S_1 = \{\mathbf{n}_i, \mathbf{n}_j\}$ defines the source and target node and $S_2 = \{\mathbf{n}_i, \mathbf{n'}_1, \dots, \mathbf{n'}_k, \mathbf{n}_j\}$ forms the resampled shortest geodesic path from $\mathbf{n}_i$ to $\mathbf{n}_j$. The radii of $\mathbf{n'}_1, \dots, \mathbf{n'}_k$, which are denoted as $r(\mathbf{n'}_1), \dots, r(\mathbf{n'}_k)$ in the node definition, can be estimated automatically or linearly interpolated, depending on how the operation is defined.

2.2. Software Implementation

2.2.1 Architecture

Based on the SWC framework, we have built neuTube 1.0 as a GUI application upon four core modules: 2D visualization, 3D visualization, image analysis and neuron structure operation (Figure 2).

2.2.2 2D Visualization

The 2D visualization module provides functions of displaying a 3D image and neuron structures slice by slice, as well as functions allowing the user to interact with the 2D display. This module facilitates close examination and precise editing. For example, driven by this module, the user can zoom into a region of interest to view details, locate tracing point precisely, or apply fine-tuning on a neuron structure. As the purpose of 2D visualization is to show the matching quality between the reconstruction and the data rather than a realistic neuron structure, we only used two geometrical primitives, lines and circles, to represent the morphology of a neuron (Figure 3B). The 2D visualization is useful for showing the exact planar position of a node, yet not suitable for showing the position perpendicular to the plane. We used two strategies to address the issue. First, each node of the neuron is displayed as a circle when the plane cuts through the node. The circle is as large as the corresponding cross section of the node, informing the user by its size how far the node is from the plane. Second, we used colors to distinguish whether a node is centered on the current plane (on-plane) or not (off-plane): the node is shown with a fully saturated and opaque color when it is on-plane; otherwise the node color is semi-transparent and less saturated (Figure 3B). The coloring options were tuned manually according to the user feedback and then used as immutable parameters of the software. To allow the user view the global structure of a neuron under reconstruction, we also project the whole skeleton onto the slice view, but with a thin and semi-transparent mode to minimize its interference with in-focus structures.

2.2.3 3D Visualization

The 3D visualization module is designed to provide real-time rendering of 3D images and neuron structures. The user can perform tracing (Figure 3D) and editing (Figure 3F) in the 3D visualization window directly, in which any change in the neuron structure will be reflected in the 2D visualization window simultaneously, and vice versa.

This module supports both realistic neuron rendering and structural rendering by decomposing a neuron structure into three geometric primitives, including sphere, line and conical frustum. The user can choose to view a neuron as connected spheres (Figure 3E), tubes (Figure 3G) or lines (Figure 3H) for checking different morphological properties of the neuron. Besides the different view styles, the module also provides multiple color modes for inspecting topological properties of a neuron or dissecting multiple neurons.

2.2.4 Image Analysis

This module offers automatic tracing of a neuron or a neuron branch to allow the user to obtain neuron structures with minimal interaction. For example, to select a branch, the user only needs to specify a point on the branch with one click. The algorithm and design were described in Zhao et al., (2011) and Kim et al., (2012). In this paper one major improvement over the previously reported version (Kim et al.,

2012) is the replacement of the cylindrical model by the tree model defined in the SWC framework. In addition, we have implemented a point-to-point tracing function based on the shortest path method used previously in automated reconstruction (Zhao et al., 2011). This function is similar to semi-automated tracing in the Simple Neurite Tracer and Vaa3d, but we have made it available in both 2D and 3D views by following the SWC framework.

2.2.5 Neuron Structure Manipulation

The module of neuron structure manipulation provides functions for the arbitrary editing of neuron nodes (Figure 3C and Figure 3F). The user can change the geometry and topology of a neuron structure with intuitive mouse clicks or keyboard shortcuts. This module supports operations described in the SWC framework, and separates them into different levels.

We have also built high-level operations from elementary ones to reduce the labor required for structural operations. These operations are following listed.

2.2.5.1 Interpolate

In many cases, a neuron branch or a segment thereof is smooth enough to be represented by piecewise linear structures. Interpolation takes advantage of this property and allows the user to quickly correct geometrical attributes of multiple nodes (Figure 4B) by specifying the nodes that need interpolation (Figure 4A).

2.2.5.2 Set branch point

It often happens that a branch point is missed when the end of a segment is close to the interior of another segment. A completely manual editing operation would consist of selecting two nodes and joining them together. The operation of setting branch point simplifies this work by connecting the selected node (Figure 4C) to the latest node in isolated branches when the connection creates a branch point (Figure 4D).

2.2.5.3 Reset branch point

This operation provides another way to correct a branch point. In this operation, the user selects a node (Figure 4E) and the program will try to move the neighboring branching structure to the selected node (Figure 4F). The program automatically determines which branch to move based on their angles.

2.2.5.4 Connect multiple nodes

Connecting two nodes is one of the most basic operations, yet one that requires multiple steps, including selecting the nodes and triggering the connection command. When there are more and more nodes to connect, the number of human interactions increases proportionally. Therefore, neuTube 1.0 provides an operation for automatically connecting multiple nodes (Figure 4G) by their edges in the minimal spanning tree of their pairwise distance graph (Figure 4H).

2.2.5.5 Remove turn

A turn is defined as three sequentially connected nodes that form an acute angle. The node in the middle is the turning point and the other two nodes are the flank nodes. The operation of removing a turn is to set the turning point (Figure 4I) as the interpolation of the flank nodes (Figure 4J). When the turning point is a branch point, the flank nodes are its two neighbors that form the sharpest turn.

2.2.5.6 Resolve crossover

Crossover is a common tracing error in tracing when two branches are close at a certain point (Figure 4K). Correcting a crossover requires several operations of connecting and breaking nodes. Therefore, we added an operation of automatic inference of crossover (Figure 4L) to make the editing easier.

2.2.6 Implementation

The software is written in the C and C++ programming languages with several third-party libraries. The main third-party library is the Qt library (http://qt-project.org), which provides a cross-platform framework for GUI development. The 3D visualization module is built upon OpenGL 2.0 (http://www.opengl.org) and its shading language, GLSL (http://www.opengl.org/documentation/glsl). We developed a fast engine for rendering neuron structures by writing highly efficient shaders for two geometric primitives, sphere and conical frustum. The vertex shader finds bounding boxes of the geometric primitives on the screen, and then the fragment shader calculates ray-quadric intersections for each pixel inside the rasterized bounding box. All of our geometric primitives have adjustable opacity options and can be visualized in the order needed to generate a reasonable semi-transparent scene. For realistic rendering of complicated semi-transparent scenes, we have also implemented Dual Depth Peeling and Weighted Average Blending (Bavoil and Myers, 2008), which are two commonly used order-independent transparency methods. Since the two methods do not require special hardware features of high-end graphical cards, they provide neuTube 1.0 the ability of rendering complicated scenes realistically without comprising the software portability. The user can switch from one method to the other in runtime to determine which one is better for the current scene.

To show an image signal in 3D, a volume, which contains the original image of the neurons to reconstruct, is uploaded to GPU as 3D texture and is rendered by a volume shader. The volume shader provides several volume composite methods, including Direct Volume Rendering (DVR), Maximum Intensity Projection (MIP) and its opaque variant, Local Maximum Intensity Projection (LMIP) (Sato et al., 1998). Each method has its own advantages. For example, MIP opaque allows the user to see weak signals that are typical of thin neural branches, LMIP is an extended version of MIP that can clearly depict spatial interrelations of neural branches, and DVR illustrates bright structures with low noise (Fishman et al., 2006). Users can also trace interactively in a 3D view by providing a seed point for tracing with a single mouse click, which represents a ray passing through the 3D volume. The seed point used for tracing is determined as the first location with maximum intensity along the ray.

3. Results

We compared our neuTube 1.0 to other neuron reconstruction softwares, namely, Neuromantic and Neurostudio. These two softwares were chosen because their designs are close to the SWC framework, although they lack several important features available in the framework (Table 1). Four 3D images from the DIADEM datasets (Brown et al., 2011) were traced using all three softwares by four users given the same time constraint. Similar to the situation of real applications, the user can decide to stop tracing whenever he/she could not identify or fix an error. This reflects how well the software visualizes the reconstruction along with the data and the flexibility of the editing functions. The accuracy of tracing was measured by how well the critical points, including branching points and termini, were reconstructed compared to ground truth reconstructions. We extracted branching and terminal points as two point sets from each tracing result and matched them to the ground truth by solving the Linear Assignment Problem (LAP) using the Jonker-Volgenant Algorithm (Jonker and Volgenant, 1987). Assuming there are a total of N points with M of them matched to the ground truth, the reconstruction error is calculated as:

$$\text{Error} = \frac{T_d\left(F_p + F_n\right) \ + \sum_{m=1}^{M} d_m}{N}$$

where F_p and F_n are the number of false positives and the number of false negatives respectively, T_d is the maximal distance allowed between two matched points (Figure 5A), and d_m is distance between the mth matched pair of points. In this calculation, the term $T_d\left(F_p + F_n\right)$ is the cost of missing critical points and $\sum_{m=1}^{M} d_m$ is the cost of position offset.

Our error metric is designed on the basis of the DIADEM metric (Gillette et al., 2011), but with two major modifications for better evaluation of interactive neuron reconstruction. One modification is that our metric matches critical points globally, while the DIADEM metric matches critical points in a certain order, which starts from the root position and may give an upstream node more importance. For user editing, missing an upstream node and missing a downstream one usually mean the same type of error. Our matching method is order-independent and treats these nodes equally. The other different feature of our metric is the combination of topological errors and position errors, with the introduction of the matching threshold (T_d), as the weight of mismatches. The threshold T_d is similar to the threshold region of the DIADEM metric, but we do not assign it a fixed value, which is often subjective or application dependent. Instead, we define the error metric as a function of T_d.

By comparing scores across a wide range of threshold values, we showed that neuTube 1.0 achieved consistently better reconstruction accuracy than Neuromantic and Neurostudio (Figure 5B). The advantage of neuTube 1.0 is more significant when the threshold is larger, indicating that neuTube 1.0 helps the user obtain more accurate neuron structures by identifying more critical points than the other two software tools.

Feature comparison table

Software	Undo/Redo	2D Editing	3D Editing	3D Image Interaction	2D Neuron Visualization	3D Visualization
neuTube 1.0	Unlimited	Yes	Yes	Yes	Slice-by-Slice	Volume & Neuron Structure
Neuromantic	1 step	Yes	Limited[2]	No	Slice-by-Slice	Neuron Structure
Neurostudio	1 step	Limited[1]	Limited[2]	No	Projection	Volume & Neuron Structure

[1] Cannot change node size
[2] No topological operation

4. Application Example

We have used neuTube 1.0 to map the fine-scale synaptic connectivity between hippocampal regions (CA3-CA1) of the mouse brain (Druckmann et al., 2014). To analyze the spatial synaptic connectivity pattern, mammalian GFP reconstitution across synaptic partners (mGRASP) (Kim et al., 2012) was used to label the synapses, and red fluorescence protein (i.e. dTomato) was used to label the post-synaptic dendrites (Figure 6A). neuTube 1.0 was used to reconstruct 3D structures of post-synaptic neurons (Figure 6B). In our application, we detected the mGRASP-labeled synapses using our mGRASP detection package (Feng et al., 2012) (Figure 6C), and then assigned each synapse to a reconstructed neuron by calculating its intensity-weighted distances to all nearby neurons (Figure 6D). To make the mapping more accurate in the step of synapse assignment, we need to reconstruct not only the selected neurons but also the remaining dendrite branches or background neurons (Figure 6B) because the distance to the nearest selected neuronal branch alone can mis-assign synapse puncta (Feng et al., 2014). A practical solution to this is to reconstruct all dendrite branches from the 3D image first and then edit the target neurons, which must be reconstructed correctly. neuTube turned out to be the right tool for this problem because the SWC framework specifies that the software can start the reconstruction from any SWC file. With the help of neuTube 1.0, we have built a fine-scale mapping of the hippocampal CA3-CA1 circuit and, with further statistical analysis, revealed spatially structure and clustered synaptic connectivity patterns between CA3 and CA1 (Druckmann et al., 2014).

5. Discussion

We designed the SWC framework and implemented it in neuTube 1.0 (www.neutracing.com) to improve the efficiency of reconstructing neuron structures accurately. Guided by the framework, the software combines 2D/3D visualization, semi-automated tracing algorithms and flexible editing options to simplify the task of neuron reconstruction. The SWC framework is not designed to solve the problem of high-throughput neuron tracing, which is different and more challenging. As revealed by the recent DIADEM competition (Liu, 2011), a reliable and generally applicable high-throughput neuron-tracing tool may not be available in the near future. While waiting for the ideal solution, neuroscientists will benefit from better neuron reconstruction tools. Therefore, the goal of the SWC framework is to provide a general architecture, which can adopt state-of-the-art image analysis methods and modern software techniques, for building better interactive neuron reconstruction tools.

Our framework has one limitation, which is that it can only produce neuron structures defined in the SWC format. However, this is usually not a significant concern because the SWC model suffices for most purposes, such as comparing neuron shapes, performing Sholl analysis, uploading neuron structures to NeuroMorpho.org (Ascoli et al., 2007), and simulating neuron activities. Many researchers prefer the SWC format rather than more complicated models because it helps to avoid overfitting to imaging artifacts: the resolution of optical microscopy is usually not high enough to reveal fine details. Even when a structure more complex than the SWC model is needed, reconstructing the neurons in the SWC model is still useful as an initial input for later shape refinement (Evers et al., 2005).

Our experiment showed that the results from neuTube 1.0 were generally better than those from Neurostudio (Myatt et al., 2012) and Neuromantic (Wearne et al., 2005), but it is still worth noting the strengths of these softwares. Neuromanic allows multi-tile tracing to reconstruct neurons from more than one field of view.

This is particularly useful for reconstructing a large neuron that requires horizontal stage movement to cover all branches. Neurostudio offers only limited free editing functions, but its ability to trace multiple branches from one seed point is a very useful feature to reduce labor, and its intrinsic radius estimation based on rayburst sampling (Rodriguez et al., 2006) can be implemented in any other software to refine the neuron structure.

As the functions of multi-branch tracing and rayburst radius estimation naturally fit in the SWC framework, we plan to include them in the future upgrade of neuTube 1.0. Additionally, there are ongoing efforts to extend the software to broader applications, including tracing neurons in bright-field images and analyzing neuron morphologies, such as identifying neuron types from electron microscope reconstructions (Zhao and Plaza, 2014).

Because a user can import results from other software into neuTube 1.0 to do further editing, neuTube 1.0 is also a complementary tool to other automated or interactive neuron tracing tools. For instance, the Vaa3d software has added neuTube 1.0 as a plug-in in recent releases (vaa3d.org). On the other hand, other developers can improve their own software by adopting the SWC framework. To facilitate any such adoption, we have made the source code of neuTube 1.0 available at https://github.com/janelia-flyem/NeuTu.

References

Ascoli GA, Donohue DE, Halavi M (2007) NeuroMorpho.Org: a central resource for neuronal morphologies. J Neurosci 27:9247–9251.

Bavoil L, Myers K (2008) Order independent transparency with dual depth peeling. NVIDIA OpenGL SDK:1–12.

Brown KM, Barrionuevo G, Canty AJ, De Paola V, Hirsch JA, Jefferis GSXE, Lu J, Snippe M, Sugihara I, Ascoli GA (2011) The DIADEM Data Sets: Representative Light Microscopy Images of Neuronal Morphology to Advance Automation of Digital Reconstructions. Neuroinformatics 9:143–157.

Cannon RC, Turner DA, Pyapali GK, Wheal HV (1998) An on-line archive of reconstructed hippocampal neurons. Journal of Neuroscience Methods 84:49–54.

Druckmann S, Feng L, Lee B, Yook C, Zhao T, Magee JC, Kim J (2014) Structured Synaptic Connectivity between Hippocampal Regions. Neuron 81:629–640.

Evers JF, Schmitt S, Sibila M, Duch C (2005) Progress in functional neuroanatomy: precise automatic geometric reconstruction of neuronal morphology from confocal image stacks. J Neurophysiol 93:2331–2342.

Feng L, Kwon O, Lee B, Oh WC, Kim J (2014) Using mammalian GFP reconstitution across synaptic partners (mGRASP) to map synaptic connectivity in the mouse brain. Nature Protocols 9:2425–2437.

Feng L, Zhao T, Kim J (2012) Improved synapse detection for mGRASP-assisted brain connectivity mapping. Bioinformatics 28:i25–i31.

Fishman EK, Ney DR, Heath DG, Corl FM, Horton KM, Johnson PT (2006) Volume Rendering versus Maximum Intensity Projection in CT Angiography: What Works Best, When, and Why1. RadioGraphics 26:905–922.

Gillette TA, Brown KM, Ascoli GA (2011) The DIADEM Metric: Comparing Multiple Reconstructions of the Same Neuron. Neuroinformatics 9:233–245.

Jonker R, Volgenant A (1987) A shortest augmenting path algorithm for dense and sparse linear assignment problems. Computing 38:325–340.

Kim J, Zhao T, Petralia RS, Yu Y, Peng H, Myers E, Magee JC (2012) mGRASP enables mapping mammalian synaptic connectivity with light microscopy. Nat Methods 9:96–102.

Liu Y (2011) The DIADEM and Beyond. Neuroinformatics 9:99–102.

Long F, Zhou J, Peng H (2012) Visualization and Analysis of 3D Microscopic Images Lewitter F, ed. PLoS Comput Biol 8:e1002519.

Longair MH, Baker DA, Armstrong JD (2011) Simple Neurite Tracer: open source software for reconstruction, visualization and analysis of neuronal processes. Bioinformatics 27:2453–2454.

Luisi J, Narayanaswamy A, Galbreath Z, Roysam B (2011) The FARSIGHT trace editor: an open source tool for 3-D inspection and efficient pattern analysis aided editing of automated neuronal reconstructions. Neuroinformatics 9:305–315.

Meijering E (2010) Neuron tracing in perspective. Cytometry 77A:693–704.

Myatt DR, Hadlington T, Ascoli GA, Nasuto SJ (2012) Neuromantic - from semi-manual to semi-automatic reconstruction of neuron morphology. Front Neuroinform 6:4.

Peng H, Long F, Zhao T, Myers E (2011) Proof-editing is the bottleneck of 3D neuron reconstruction: the problem and solutions. Neuroinformatics 9:103–105.

Peng H, Ruan Z, Long F, Simpson JH, Myers EW (2010) V3D enables real-time 3D visualization and quantitative analysis of large-scale biological image data sets. Nature Biotechnology 28:348–353.

Rodriguez A, Ehlenberger DB, Hof PR, Wearne SL (2006) Rayburst sampling, an algorithm for automated three-dimensional shape analysis from laser scanning microscopy images. Nature Protocols 1:2152–2161.

Sato Y, Shiraga N, Nakajima S, Tamura S, Kikinis R (1998) Local Maximum Intensity Projection (LMIP): A New Rendering Method for Vascular Visualization. Journal of Computer Assisted Tomography 22:912–917.

Wearne SL, Rodriguez A, Ehlenberger DB, Rocher AB, Henderson SC, Hof PR (2005) New techniques for imaging, digitization and analysis of three-dimensional neural morphology on multiple scales. Neuroscience 136:661–680.

Zhao T, Plaza SM (2014) Automatic Neuron Type Identification by Neurite Localization in the Drosophila Medulla. arXiv q-bio.NC:1409.1892.

Zhao T, Xie J, Amat F, Clack N, Ahammad P, Peng H, Long F, Myers E (2011) Automated reconstruction of neuronal morphology based on local geometrical and global structural models. Neuroinformatics 9:247–261.

Legends

Tables

Table 1: Feature comparison of neuTube 1.0 with Neuromantic and Neurostudio.

Figures

Figure 1 Workflow of reconstructing or editing a neuron structure in the SWC framework, which defines GUI software that takes either a raw image or an SWC file as input and

generates an acceptable neuron structure through user interactions. The user can save the neuron structure into standard SWC files during or after reconstruction.

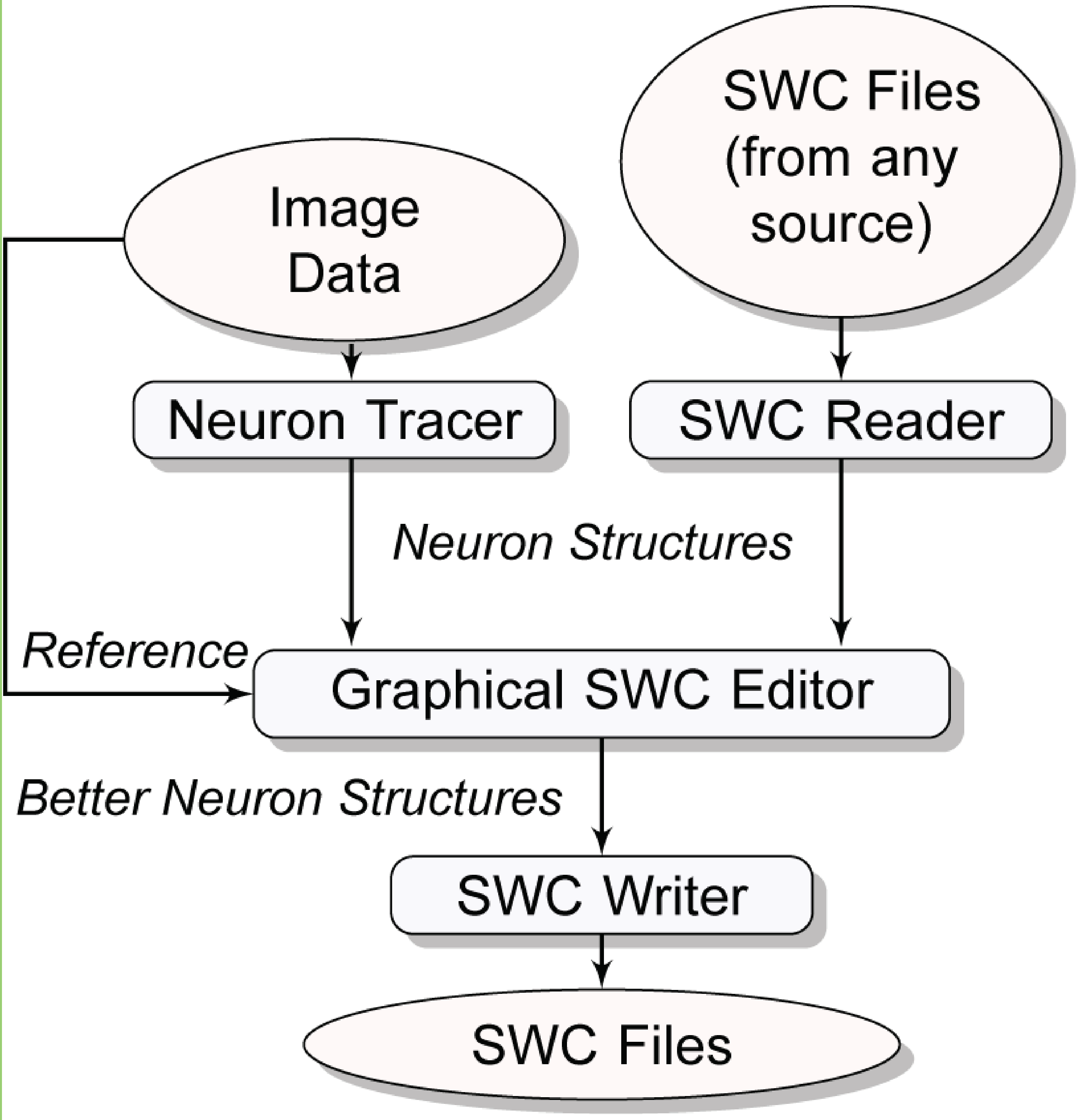
Figure 2 neuTube 1.0 is a GUI application built upon four major modules, including 2D visualization, 3D visualization, image analysis and neuron structure operation.

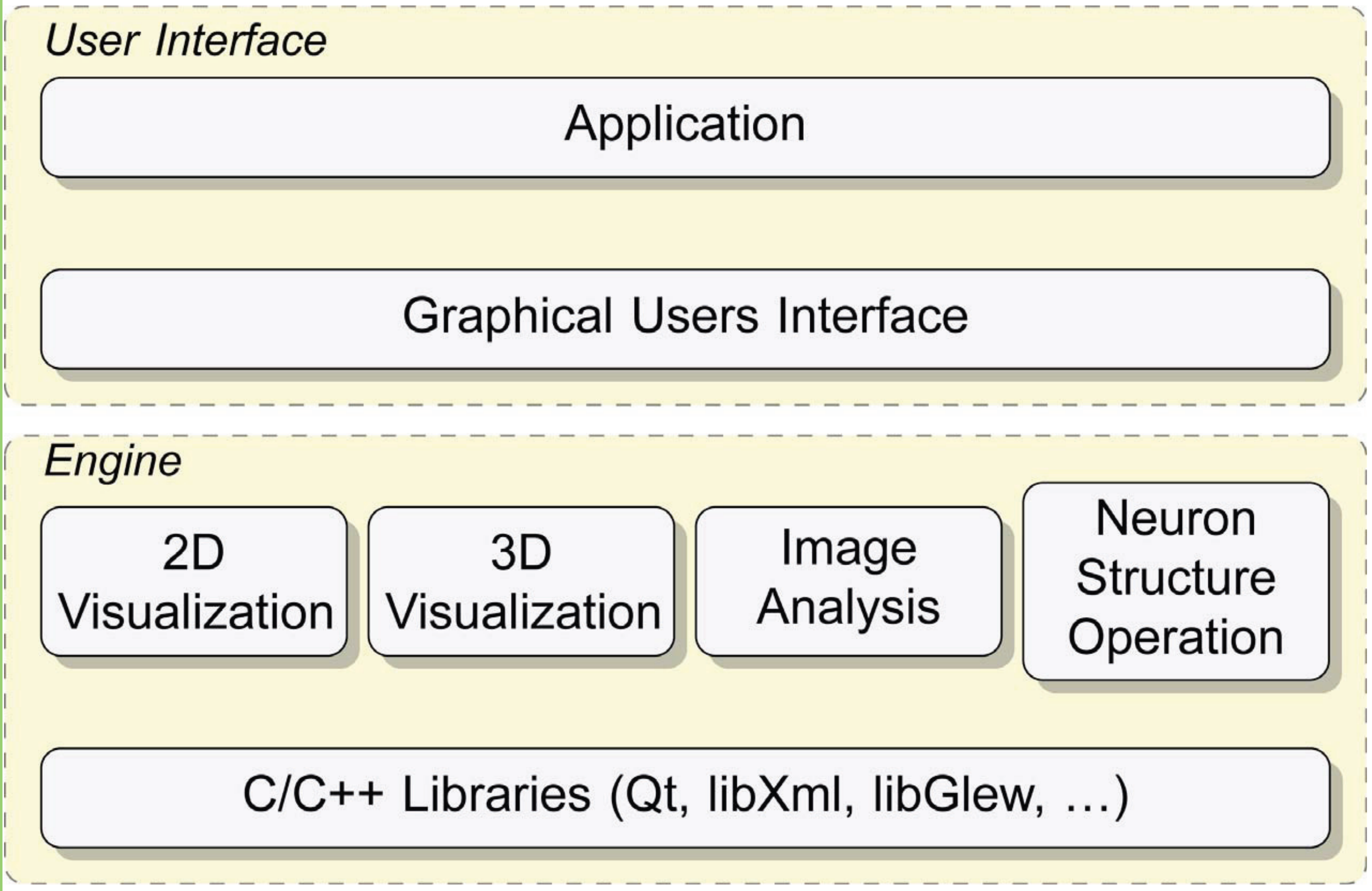
Figure 3 Tracing and editing interface of neuTube 1.0. (A) Interactive tracing in 2D view. (B) 2D view of SWC nodes. On-plane and off-plane nodes are distinguished by color saturation and transparency. A node with a yellow bounding box indicates that it is selected. (C) The context menu for editing in the 2D view, which can be triggered by a right mouse click. (D) Interactive tracing in 3D view. (E) 3D visualization of the tracing results. Branch nodes and terminal nodes are shown in green and yellow colors respectively. Selected nodes are shown with their bounding boxes. (F) The context menu for editing in the 3D view. (G) A neuron shown as connected tubes. (H) A neuron shown as lines.

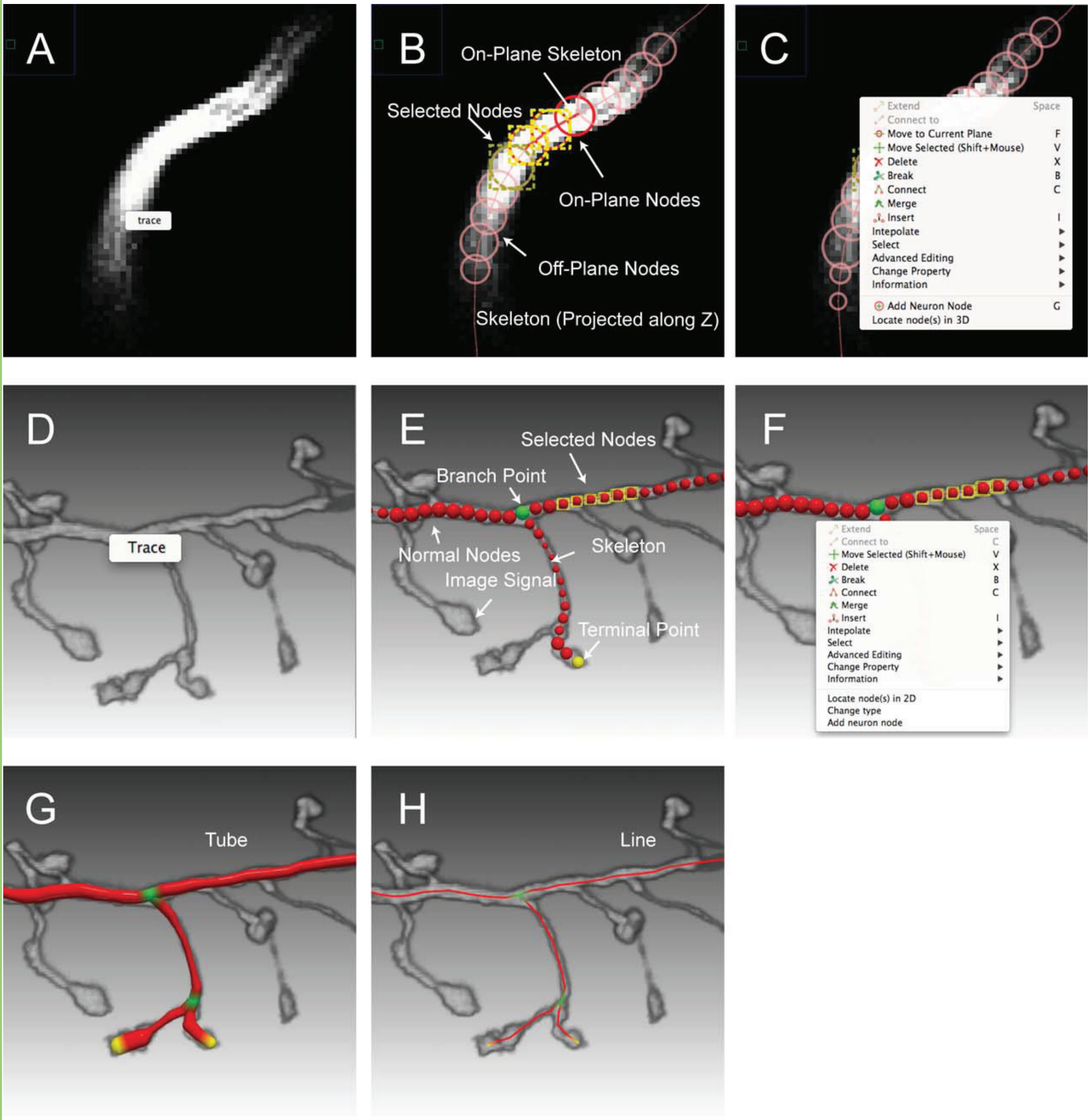
Figure 4 Examples of high-level operations of neuTube 1.0. To illustrate the operation, we visualize the nodes of a neuron in different colors according to the topology: blue for root nodes, green for branch nodes, yellow for leaf nodes and red for other nodes. Selected nodes are highlighted by a yellow bounding box. For the corresponding operation as named in each row, the figure on the left (A), (C), (E), (G), (I) or (K) shows the selected nodes to operate and the one on the right (B), (D), (F), (H), (J) or (L) shows the result of operation.

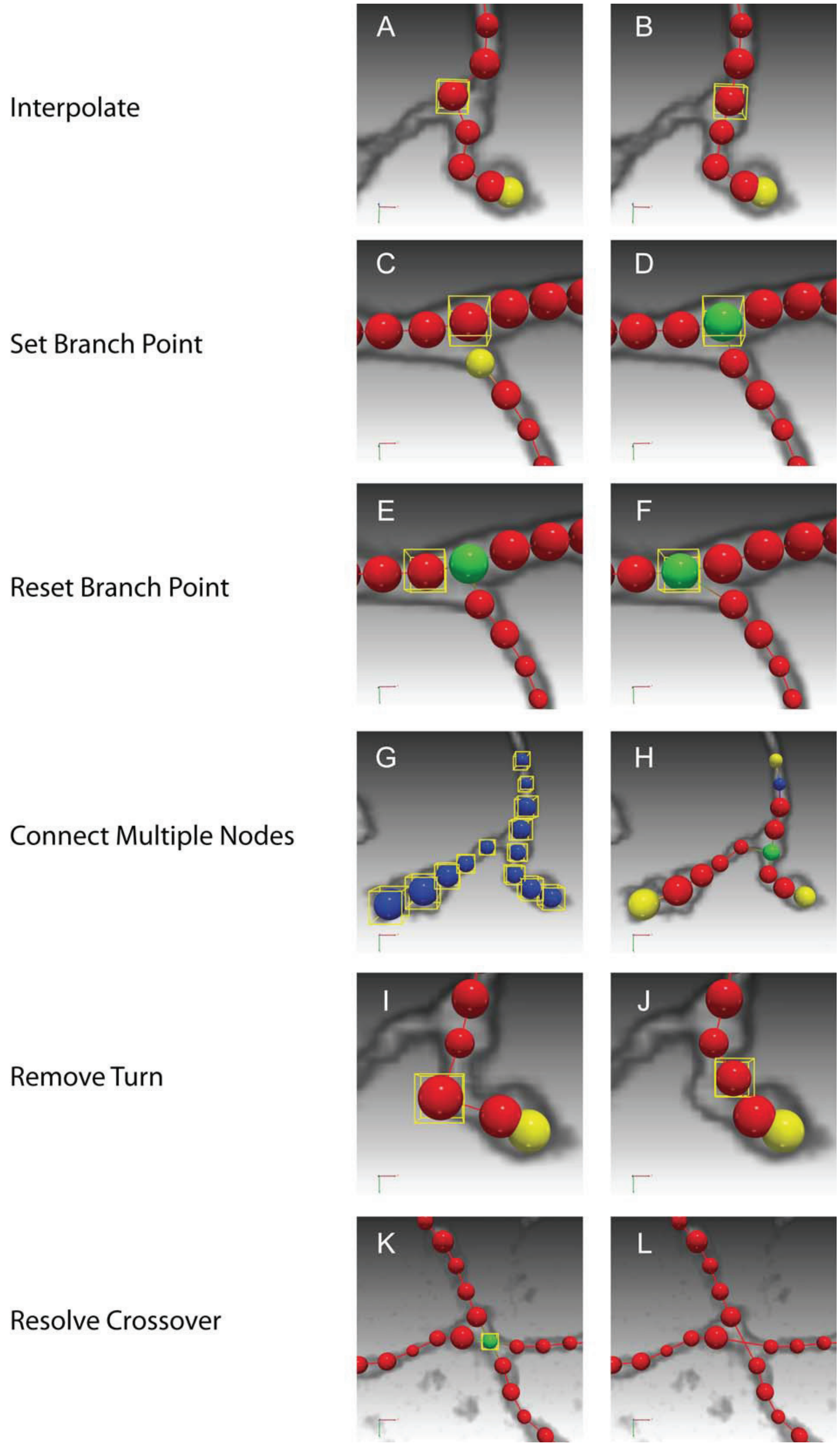
Figure 5 neuTube 1.0 helps produce significantly more accurate neuron structures than Neuromantic and Neurostudio do. (A) Node_g is a critical point from ground truth neuron and Node_t is a critical point from tracing result. These two points can be matched when $T_d = T_{d2}$ because the distance between them is less than T_{d2}. They cannot be matched when $T_d = T_{d1}$. (B) The solid curves show average errors measuring the discrepancy between the critical point sets from user reconstruction and ground truth under different distance thresholds. The surrounding envelopes are the 95% confidence intervals. The error curve of neuTube 1.0 (green) is consistently lower than the other two, with p value < 0.01 (t-test) when $T_d \geq 6$ (compared to Neurostudio) or $T_d \geq 5$ (compare to Neuromantic).

Figure 6 Mapping brain connectivity with neuTube 1.0. (A) The original 3D confocal image contains post-synaptic neurons. (B) The target neuron (green) was traced semi-automatically. The red branches belong to background neurons. (C) mGRASP-labeled synapses (yellow) were detected automatically, with sizes enlarged for better visualization. (D) The synapses were mapped to the target neuron (green) and background neurons (red).

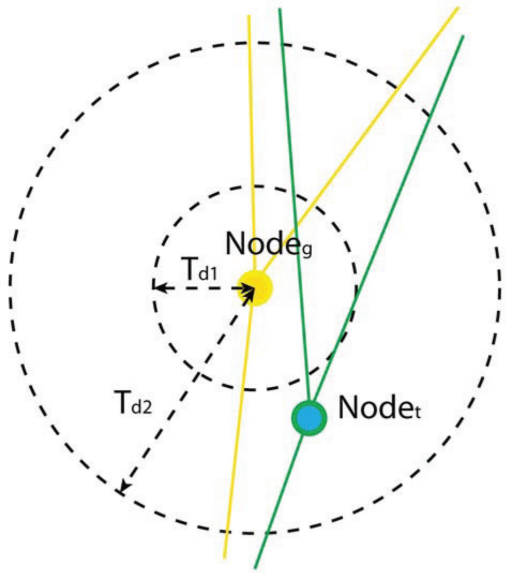

SWC Files (from any source)
Image Data
Neuron Tracer
SWC Reader
Neuron Structures
Reference
Graphical SWC Editor
Better Neuron Structures
SWC Writer
SWC Files


User Interface
Application
Graphical Users Interface
Engine
2D Visualization
3D Visualization
Image Analysis
Neuron Structure Operation
C/C++ Libraries (Qt, libXml, libGlew, …)


A
trace
B
On-Plane Skeleton
Selected Nodes
On-Plane Nodes
Off-Plane Nodes
Skeleton (Projected along Z)
C
Extend Space
Connect to
Move to Current Plane F
Move Selected (Shift+Mouse) V
Delete X
Break B
Connect C
Merge
Insert I
Intepolate
Select
Advanced Editing
Change Property
Information
Add Neuron Node G
Locate node(s) in 3D
D
Trace
E
Selected Nodes
Branch Point
Normal Nodes
Skeleton
Image Signal
Terminal Point
F
Extend Space
Connect to C
Move Selected (Shift+Mouse) V
Delete X
Break B
Connect C
Merge
Insert I
Intepolate
Select
Advanced Editing
Change Property
Information
Locate node(s) in 2D
Change type
Add neuron node
G
Tube
H
Line

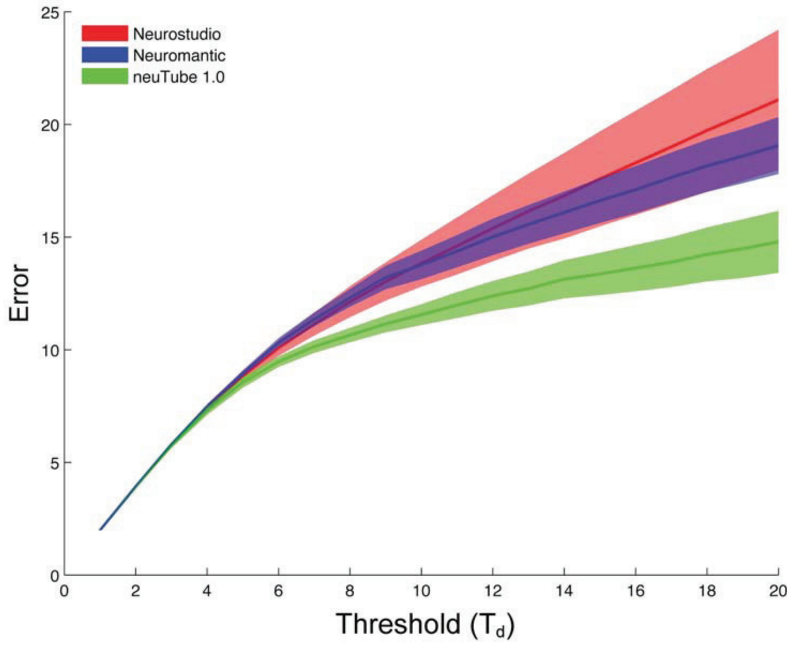

Interpolate
A
B
Set Branch Point
C
D
Reset Branch Point
E
F
Connect Multiple Nodes
G
H
Remove Turn
I
J
Resolve Crossover
K
L

A

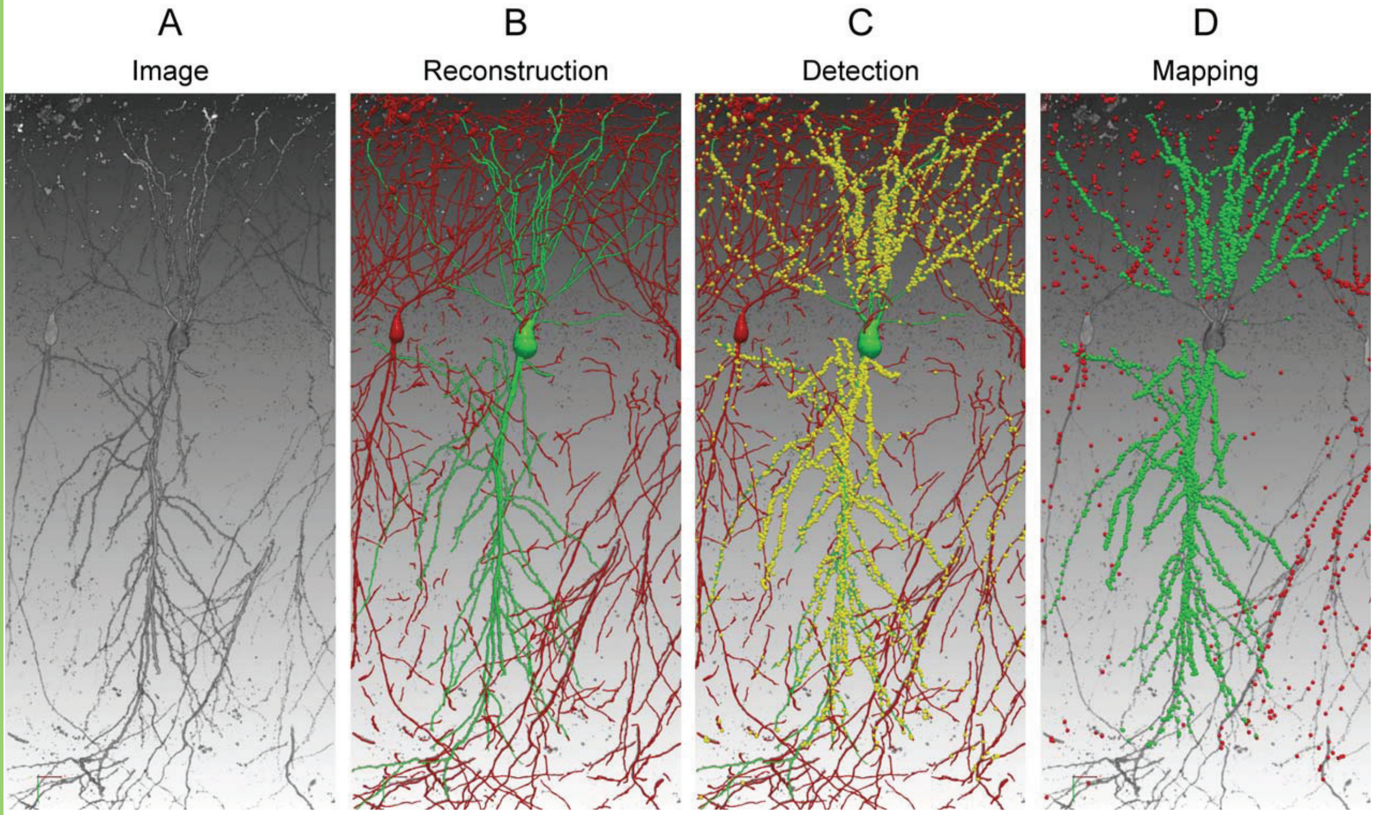


Node$_g$
T$_{d1}$
T$_{d2}$
Node$_t$

B



Neurostudio
Neuromantic
neuTube 1.0
Error
Threshold (T$_d$)
0
2
4
6
8
10
12
14
16
18
20
5
10
15
20
25


A
Image
B
Reconstruction
C
Detection
D
Mapping