Segmentation and Tracing of Neurons in 3D

Saurav Basu, Alla Aksel, Barry Condron and Scott T. Acton, *Member, IEEE*

Abstract— In order to understand the brain, we need to first understand the morphology of neurons. In the neurobiology community, there have been recent pushes to analyze both neuron connectivity and the influence of structure on function. Currently, a technical roadblock that stands in the way of these studies is the inability to automatically trace neuronal structure from microscopy. On the image processing side, proposed tracing algorithms face difficulties in low contrast, indistinct boundaries, clutter and complex branching structure. To tackle these difficulties, we develop *Tree2Tree*, a robust automatic neuron segmentation and morphology generation algorithm. Tree2Tree uses a local medial tree generation strategy in combination with a global tree linking to build a maximum likelihood global tree. Recasting the neuron tracing problem in a graph-theoretic context enables Tree2Tree to estimate bifurcations naturally, which is currently a challenge for current neuron tracing algorithms. Tests on cluttered confocal microscopy images of *Drosophila* neurons give results that correspond to ground truth within a margin of $\pm \mathbf{2.75}$ % normalized mean absolute error.

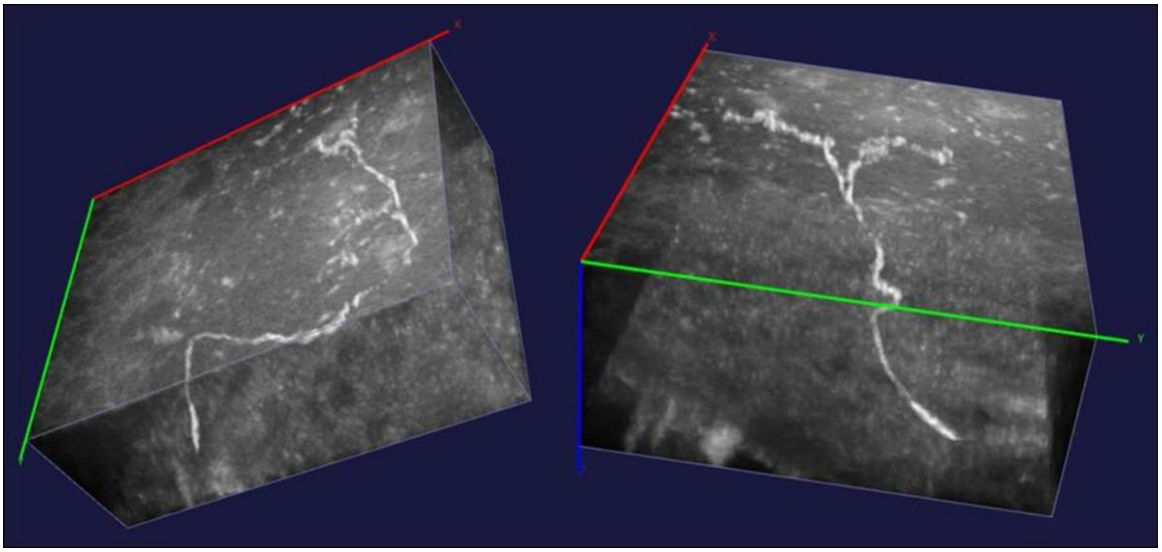
Index Terms— Neuron image analysis, neuron tracing, segmentation,

Manuscript received March 15, 2011. This work was supported in part by the NIH (EB0001826) and by CACI through a grant from the ARO (S11-113016).

S. Basu and A. Aksel are with the Department of Electrical and Computer Engineering, University of Virginia, Charlottesville, VA 22904 (e-mail: saurav, alla@virginia.edu).

B. Condron is with the Department of Biology, University of Virginia, Charlottesville, VA 22904 (e-mail: condron@virginia.edu).

S.T. Acton is with the Department of Electrical and Computer Engineering. Department of Biomedical Engineering, University of Virginia, Charlottesville, VA 22904 (e-mail: acton@virginia.edu).



(a) (b)

Fig. 1. (a) GFP stained *Drosophila* neuron 3D stack. (b) Same neuronal branch structure from a different viewing angle.

I. INTRODUCTION

AUTOMATED segmentation of neurons is one of the critical open problems in neurobiological image analysis.The knowledge of neural structure and morphology play a major role in understanding the brain [1]. In order to validate hypotheses regarding the neural connectivity and function of the brain, there exists an ongoing effort to build atlases and libraries of neural structures [2]. The major stages in this effort are to define the morphology of the neurons, the *neurome*, and the connectivity, the *connectome* [3]. Such atlases would yield both shape and structure that aids in understanding how cellular structure regulates brain function.

At present, the only fully disclosed neurome is that of the worm *C. elegans* [4]. As a result, *C. elegans* has become an established model for studying developmental and behavioral neurobiology. Limitations of this model organism include the structural simplicity and lack of morphological variation. A reasonable next step is to develop a similar set of neuronal atlases for more complex organisms, of which *Drosophila* (the fruit fly) is a salient candidate.

In this paper, we validate our segmentation and tracing method on 3D images of *Drosophila* neurons. Within the central nervous system of the fruit fly, we find a brain with about 20,000 neurons and a ventral nerve cord (VNC) of about 5000 cells [5]. The VNC is divided into subesophageal segments, which each contain about 130 neurons. The development of our image analysis methodology is motivated by the desire to segment and trace each of these VNC neurons such the structural information can be used to model conduction in large realistic networks.

Furthermore, automated segmentation (discriminating the neuron from the background, clutter and other cells) and

automated tracing (mapping the structure of the neuron) hold the possibility of opening new doors in biological research. Among these activities, in the near future, is the mapping of transcription factors in embryonic neurons and specific promoter elements that restrict gene expression to small groups of neurons ([5], [6]).

A. Limitations of Current Methods

Informatics, not bioimaging or biology itself, remains as the major roadblock in creating a neurome for complex organisms that span *Drosophila* to the mouse. Specifically, unsolved problems in automated image analysis currently preclude the development of these atlases. Automatic algorithms for segmenting and tracing neuronal structures available in literature face difficulties with neuron images which have low contrast, amorphous filament boundaries and heavy branching. Current solutions can be roughly divided into two strategies: 1) seed initialization followed by directional tracing and 2) foreground clustering followed by linking. A careful observation of these solution strategies with elaboration on the drawbacks with respect to the type of neuron images we experiment on will establish the motivation of our new method.

Broadly speaking, the first strategy is to use a seed voxel/position for initialization followed by tracing the loci of the brightness maxima of the filament like structures in the neuron images. In the method of Lu *et al*. [7], the user selects seed points to guide the reconstruction in a semiautomated fashion. Directional filters and search techniques attempt to reliably choose a tracing direction while a rule based strategy is employed to detect filament ends or bifurcation points. Al-Kofahi *et al*. ([8], [9]) estimate the best possible direction of the neuron filament by using the median value of a directional filter. This algorithm depends on sufficient contrast as well as continuous and parallel boundaries of neuronal structures for success. In another approach, Streekstra and van Pelt [10] use Gaussian derivative kernels to track neuronal filaments. In addition to it being semi-automatic with user defined seed points, these algorithms might underperform in automatic determination of branching in low contrast images and heavily fragmented neuronal filaments. In the method presented by Wolf *et al*. [11], a quadrilateral directional filter is manually initialized over a seed point and local relative grey value differences and local maxima of intensity are used to trace the axons. The seed initialization and filament tracking algorithms, in their various forms, rely on relatively clutter free images of consistent and contiguous intensity.

In the second overall strategy, the neuronal images are processed globally without a starting seed point. Various filtering techniques have been attempted to reduce clutter before passing on the image stacks to morphological

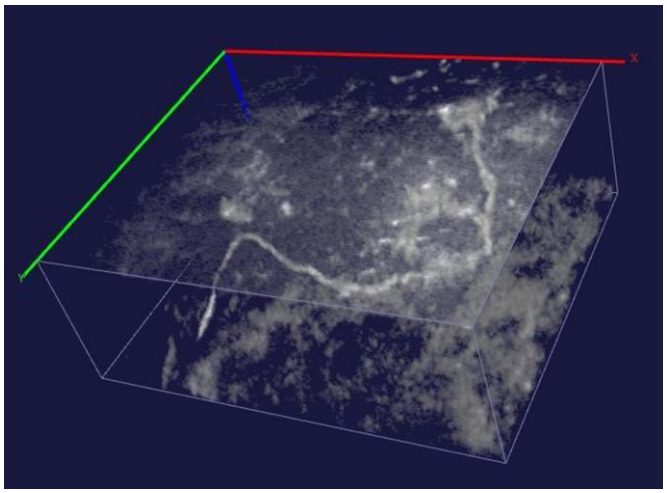


Fig. 2. Simple global binary voxel clustering performed on neuron from Fig. 1

operations. Morphological operations achieve independence from assumed geometrical regularity assumptions in contrast to the aforementioned first strategy.

As an example, Cai *et al*. [12] segment neuronal cross sections in 2D image stacks with active contours without edges as in [13] and stitch corresponding segmented contours across the slices. The algorithm relies on the spatial correspondence of contours between stack slices; and thus, the method requires optimal alignment of neuronal branches along the z axis for the segmentation to be successful. Evers *et al*. [14] use a similar approach of using active contours and snakes to trace neuronal filaments.

Other examples of this strategy include the method of Cuntz *et al*. [15] use tiling of multiple high magnification fields of view to create single mosaic images, followed by 3D skeletonization and thinning. Dima *et al*. [16] perform edge detection and validation on multiple scales to distinguish true edges from noise. Gradient following and edge traversal lead to neuron boundaries that are used later for separating the foreground from the background. 3D skeletonization is also performed by placing local restrictions on omnidirectional rays emanating at each voxel to give rise to a neural graph.

Weaver *et al*. [17] divide the whole neuronal volumes into a mosaic of volume elements and iteratively adjust the brightness and overlap manually to increase visibility and continuity of the neurons. Fine adjustments are continuously applied until binary thresholding and morphological thinning generate a continuous neuron branches. Methods in [18], [19], and [20] use a combination of thresholding and edge detection with a semiautomated selection of branch points and gap filling between thresholded neuron fragments. Selinummi *et al*. [21] incorporate Sobel edge detectors in their edge detection scheme followed by edge linking.

A commonality between all approaches in the second strategy is the dependence on binary thresholding of filtered images followed by morphological thinning and gap filling. Although noise filtering removes some of the clutter in the images, the clutter is often more distinguishable in their geometry rather than their signal strength. Binary thresholding of images that are filtered on raw voxel intensities tends to either discard information or to retain clutter.

Here, we attempt to cast the neuron segmentation/tracing problem in a more efficient paradigm that simultaneously exploits both local properties of brightness maxima and global properties of shape and structure.

B. Contribution of Our New Algorithm

A large collection of 3-D image stacks, generated in the Condron lab at the University of Virginia, represents most or all neuronal types in the *Drosophila* ventral nerve cord CNS. In this paper, we have used confocal microscopy images of individual sets of GFP-labeled *Drosophila* neurons from the Condron lab [22]. Image sets have been acquired in focal planes from dorsal to ventral, an example is shown in Fig. 1(a). A careful observation of Fig. 1(b) will show that the neuron images are characterized by low contrast, filament discontinuity, and poorly defined boundaries.

Our algorithm, called *Tree2Tree*, is an automatic neuron segmentation and morphology generation algorithm that does not require manual seed points, is not dependent on edge consistency and filament continuity, and handles neuron branching naturally. Tree2Tree gets its name from the fact that it fits a graph-theoretic tree to the neuronal tree [23]. Instead of following brightness maxima along filaments, Tree2Tree builds a maximum likelihood tree that describes the local orientation and intra-neuronal fragment connectivity. This neuron segmentation and morphology generation algorithm is the first step in generating a morphology comparison technique that might be used as a second step to query and retrieve similar neurons from a neuron database.

Brightness maxima, in combination with orientation information, provide powerful cues for segmentation of the neurons. We use geometry based maxima detectors to discard clutter with non-elongated shapes. Instead of following a maxima ridge through tracking, we gather evidence of all suitable maxima points and discard clutter for our subsequent linking phase. We therefore avoid the pitfalls of the first strategy discussed in the background such as termination of tracking due to discontinuous filaments or insignificant minima.

In Tree2Tree, we gather local information in terms of isolated segments of likely neuronal segments and perform 3D skeletonization with appropriate smoothing and pruning at a prescribed scale. Instead of using local morphological operations such as seed growing and gap filling, we now transform the neuron building process to a different domain. Using the component skeletons as independent nodes in a graph theoretic context, we impose global geometric connectivity constraints (torsion, bending and stretching) to infer likely connectivity between these nodes. The neuron building process now transforms to a building a minimum cost spanning tree of nodes themselves

made of smaller trees. Tree2Tree uses graph theoretic techniques to construct an overall tree out of smaller component trees, instead of local morphological operations which may fail to connect isolated segments due to noise or missing stretches of data. Residual clutter is also removed by a novel graph theoretic inference technique based on selecting an optimal subtree that minimizes connectivity weight while maximizing brightness information.

Overall, our process uses deterministic functionality from both existing strategies but casts the problem of neuron tracing in a new graph theoretic context such that the tree inference is performed in a transformed graphical domain rather than in the image domain. Shape, brightness and geometry are used as cues in a minimum weight spanning tree determination process with edge weights representing links which minimize bending, stretching and torsion of the overall neuronal tree. The rest of the paper is arranged as follows. Section II introduces Tree2Tree and details each step of the algorithm. Section III shows results of applying Tree2Tree to a dataset of neurons. Section IV discusses strengths and limitations of Tree2Tree and its future extension to a neuronal atlas project.

II. TREE2TREE

As mentioned, segmentation and tracing of 3D neurons from the confocal microscopy image stacks face significant obstacles. The neuron images are characterized by low contrast, inconsistent or missing neuronal branches, and filaments having amorphous shapes with poorly defined boundaries that deviate considerably from an idealized cylinder. In addition, practical difficulties in removing fragments and parts of other structures in the nerve cord during specimen preparation contribute to distracting clutter in the 3D image stacks. For example, in Fig. 1, a neuronal branch structure of the dbd sensory neuron in the Ventral Nerve Cord (VNC) of the Drosophila is shown. Note the lack of clear and consistent filamentous shape, missing information in form of broken branches and clutter generated from other structures.

Our goal is to reliably and consistently distinguish the neuronal branches from clutter having similar intensity (segmentation), and to consequently piece together the incomplete information from the inconsistent and broken branches into a coherent whole neuron (tracing). To this end, we employ a sequential methodology that performs local enhancement, partial binary clustering, local neuronal tree generation and a graph theoretic combination of local trees into a global tree structure in order. A brief sketch of our algorithm is presented in the next subsection before we proceed to give a detailed description of our algorithm.

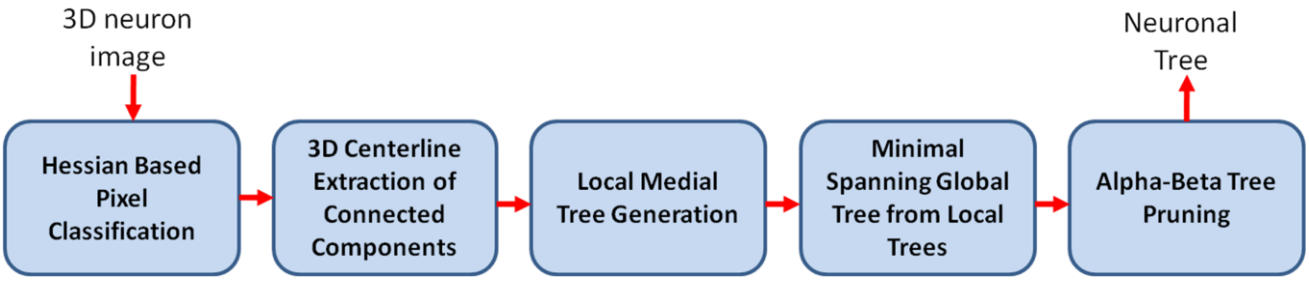


Fig. 3. A brief schematic of our algorithm showing the main processing and conceptual blocks. The 3D neuron confocal microscope image is supplied to the Hessian based preprocessing module which then passes through various intermediate forms of processing and representation to finally come out as a traced neuronal tree as a final product from the tree pruning processing block.

A. Brief Sketch of our Algorithm:

To distinguish from clutter, we determine neuronal pixels based on a Hessian based shape descriptor that indicates the probability of the neighborhood from which the pixel has been sampled to belong to a neuronal branch. In order to piece together the incomplete and fragmented information of the individually determined neuronal branches, we deviate from the established practice of following brightness maxima along filaments after starting from a seed point. Instead, *Tree2Tree* builds a maximum likelihood tree that optimally explains the orientation and connectivity of visible fragmented neuronal branches in the neuron images. Residual clutter that resembles neuronal branches and could not be removed through the Hessian based decluttering phase is removed in the last step through a novel graph based pruning method.

Using a 3D intensity image of a neuron as input, our algorithm outputs the centerline of the neuron as a tree, with adjacent nodes placed at user-defined resolution. A brief schematic of the overall algorithm with the major processing/conceptual blocks are presented in Fig. 3. In the following sections, the details of the sequential steps, the

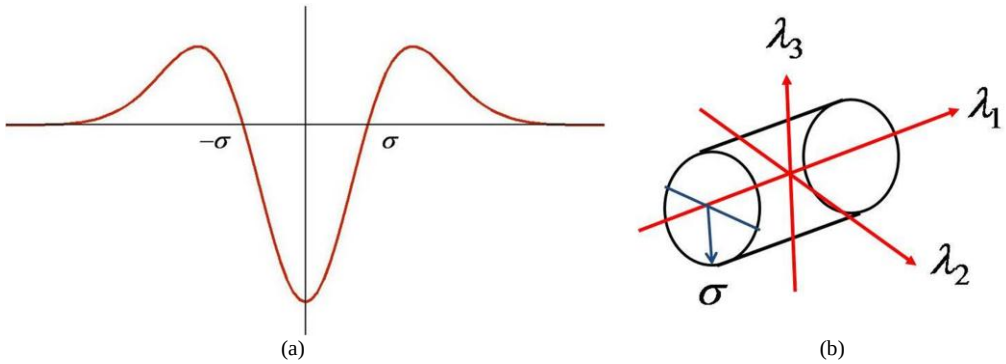
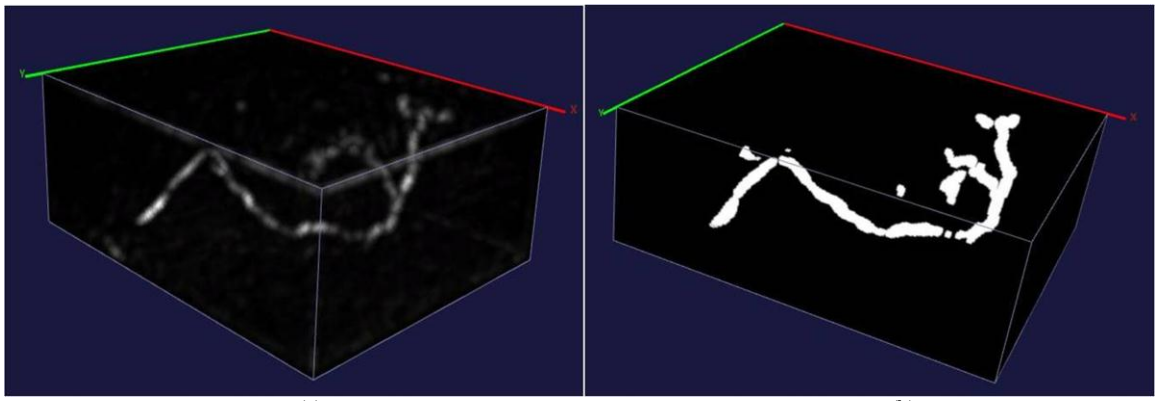


Fig. 4. (a) Kernel used to measure neuron response. (b) Ideal eigenvector directions for a tubular structure

rationale behind them and the mathematical formulation are presented.



(a) (b)

Fig. 5. (a) Hessian based suppression of clutter. (b) Filamentary measure image from 5(a) after Gaussian smoothing and binary voxel clustering

B. Hessian Based Pixel Classification (Step 1):

The preliminary neuronal pixel segmentation can be treated as a binary classification problem into the foreground neuron pixels (brighter) and the background pixels (darker). A global 2-class classification over the entire image does not give satisfactory results because the relative brightness of the neuron compared to the background varies widely over the image. Fig. 2 shows poor neuron segmentation based on a binary intensity clustering algorithm (in this case, the method of Otsu [24]). Instead of simply thresholding, we exploit the shape characteristics of neuron branch to differentiate a neuron from a non-neuron area.

We use a Hessian based scheme to detect pixels belonging to elongated tubular structures in the image volume. Let us briefly describe the strength of Hessian based approaches in detecting elongated structures in an intensity image, after which we will describe our particular scheme in this research. Methods such as the one presented in [25] have used vessel enhancement algorithms using the Hessian of the images to enhance vessels in angiograms for improved viewing quality. In its simplest form, Hessian based vessel enhancement might be viewed as a filter that enhances tubular structures but deemphasizes non-tubular structures, many of them having higher intensities in the intensity space than the actual vessels. A common formulation to analyze the local filamentous nature of an image I is to test the second derivative $\mathcal{L}_{u,v}$ of the image I at point x_0 in directions (u, v) and scale σ. More formally,

$$\mathcal{L}_{u,v}(I, \sigma, x_0) = \lim_{t \to 0+} \frac{\mathcal{L}_u(I, \sigma, x_0 + tv) - \mathcal{L}_u(I, \sigma, x_0)}{t} \tag{1}$$

where,

$$\mathcal{L}_u(I, \sigma, x_0) = \lim_{t \to 0+} \frac{G(\sigma) * (I(x_0 + tu) - I(x_0))}{t} \tag{2}$$

is the first derivative of the Gaussian smoothed image $G(\sigma) * I$ at scale σ at point x_0 and direction v. It has been shown in [25] that computing the second derivative of the image in a direction v is equivalent to convolving the image with a kernel that is the second derivative of the Gaussian $G(\sigma)$ in the direction v. For example, if the image is two-dimensional, taking the probe directions (u, v) to be aligned to one of the coordinate axes x (*i.e.*, $u = v = x$),

$$\mathcal{L}_{u,v}(I, \sigma, x_0) = \frac{\partial^2 G(\sigma)}{\partial x^2} * I(x)_{x=x_0} \tag{3}$$

For a 2D image, the kernel $\frac{\partial^2 G(\sigma)}{\partial x^2}$ in equation (3) along a direction x is shown in Fig. 4(a). The second derivative operator of the smoothed image at the scale σ is a vessel detector at vessel radius σ and has a high negative response at the center of the vessel along a radial direction of the vessel but near zero response along the axis of the vessel. Since we have assumed that the image formation is smooth, points nearby to the approximate center of an elongated tubular structure will have relatively low second derivatives along the approximate axis of the elongated structure. Orthogonal to this axis, the second derivative will be highly negative within the neuron. Here, we have orthogonal directions $v_1, v_2,$ and v_3 such that the second derivative of the intensity is smallest along v_1 and largest along v_3, and if λ_i denotes the value of the second derivative along direction v_i, we want to have $\lambda_1 \leq \lambda_2 \leq \lambda_3$. If we express the Hessian matrix $\mathcal{H}$ of the 3D image I at scale σ as

$$\mathcal{H}(I, \sigma) = \begin{bmatrix} \mathcal{L}_{x,x} & \mathcal{L}_{x,y} & \mathcal{L}_{x,z} \\ \mathcal{L}_{y,x} & \mathcal{L}_{y,y} & \mathcal{L}_{y,z} \\ \mathcal{L}_{z,x} & \mathcal{L}_{z,y} & \mathcal{L}_{z,z} \end{bmatrix}$$

$$= \begin{bmatrix} \frac{\partial^2 G(\sigma)}{\partial x^2} * I & \frac{\partial^2 G(\sigma)}{\partial x \partial y} * I & \frac{\partial^2 G(\sigma)}{\partial x \partial z} * I \\ \frac{\partial^2 G(\sigma)}{\partial y \partial x} * I & \frac{\partial^2 G(\sigma)}{\partial y^2} * I & \frac{\partial^2 G(\sigma)}{\partial y \partial z} * I \\ \frac{\partial^2 G(\sigma)}{\partial z \partial x} * I & \frac{\partial^2 G(\sigma)}{\partial z \partial y} * I & \frac{\partial^2 G(\sigma)}{\partial z^2} * I \end{bmatrix} \tag{4}$$

then the directions of the largest and smallest second derivatives at any point in the image volume are given by the eigenvectors $\mathrm{v}_1, \mathrm{v}_2,$ and v_3 of the matrix $\mathcal{H}(\mathrm{I}, \sigma)$ evaluated at that point, and the corresponding second derivative values are given by the eigenvalues $\lambda_1 \leq \lambda_2 \leq \lambda_3$.

Calculation of the directions of maximum and minimum second derivatives is not enough to indicate whether a voxel belongs to a neuronal branch or not. If a voxel belongs to a neighborhood exhibiting elongated tubular

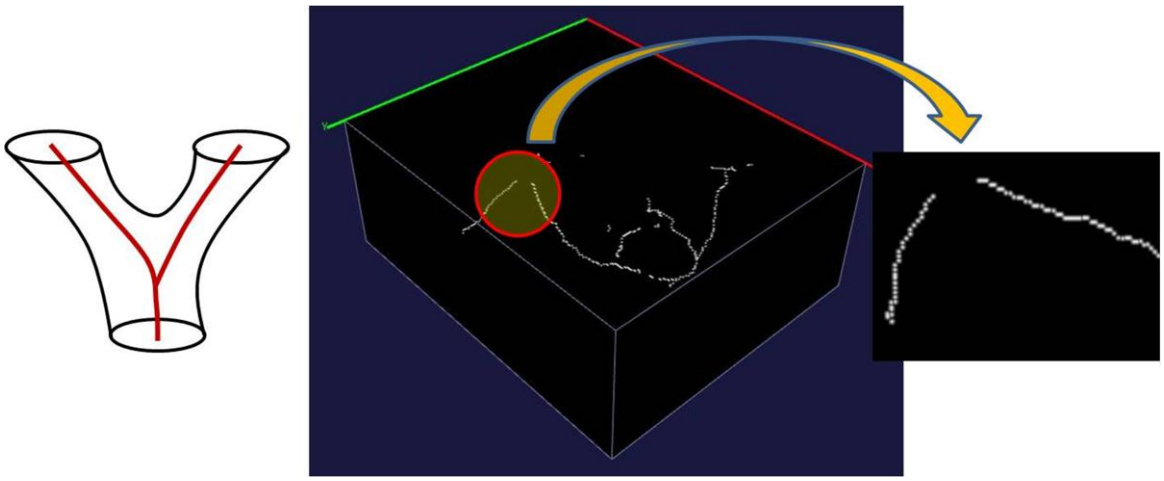
structure, then we expect certain relationships between the concerned eigenvalues. The minimum second derivative aligned along the approximate centerline of the structure should be close to zero. The other two eigenvalues should be aligned along approximate radial directions of the vessels, should be negative and high magnitude, and since a tubular structure should not have a flattening along any particular direction in its cross section, should be close to each other in their value. Also, since we have implicitly assumed that the neuronal structures are brighter than the background, a positive response for the second derivative kernel would indicate non-neuronal structure. Symbolically, the constraints are written as

$$\begin{aligned} &|\lambda_1| \approx 0 \\ &|\lambda_1| \ll |\lambda_2| \\ &\lambda_2 \approx \lambda_3 \\ &\lambda_2, \lambda_3 \leq 0 \end{aligned} \tag{5}$$

In order to incorporate the constraints in equation (5) into a neuron like distinguishing metric, we build a *filamentary* measure $\mathcal{N}_\sigma(\mathbb{p})$ at voxel position $\mathbb{p}$ and scale σ below:

$$\mathcal{N}_\sigma(\mathbb{p}) = \begin{cases} \dfrac{|\lambda_1(\mathbb{p}) - \lambda_2(\mathbb{p})|^2}{|\lambda_1(\mathbb{p})||\lambda_2(\mathbb{p}) - \lambda_3(\mathbb{p})|} & \text{when } \lambda_2(\mathbb{p}), \lambda_3(\mathbb{p}) \leq 0 \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

As noted in the previous paragraphs, when the approximate radius of the vessel equals the scale of smoothing σ, we expect a high negative second derivative values along radial directions, and low values along the axial direction.



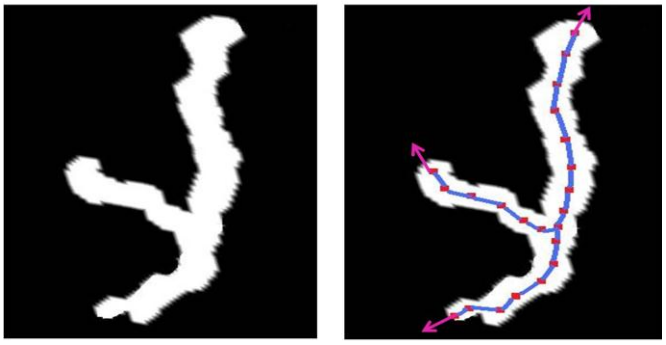
(a) (b)

Fig. 6. (a) 3D skeleton of a bifurcated tube shown in red. (b) 3D skeletons of the individual connected components in Fig. 4(b). Close-up demonstrates the gap between 3D skeletons

Thus, we choose that particular scale at which the filamentary measure gives a maximum to get a final filamentary measure $\mathcal{N}(\mathbb{p})$:

$$\mathcal{N}(\mathbb{p}) = \max_{\sigma_{min}<\sigma<\sigma_{max}} \mathcal{N}_{\sigma}(\mathbb{p}) \tag{7}$$

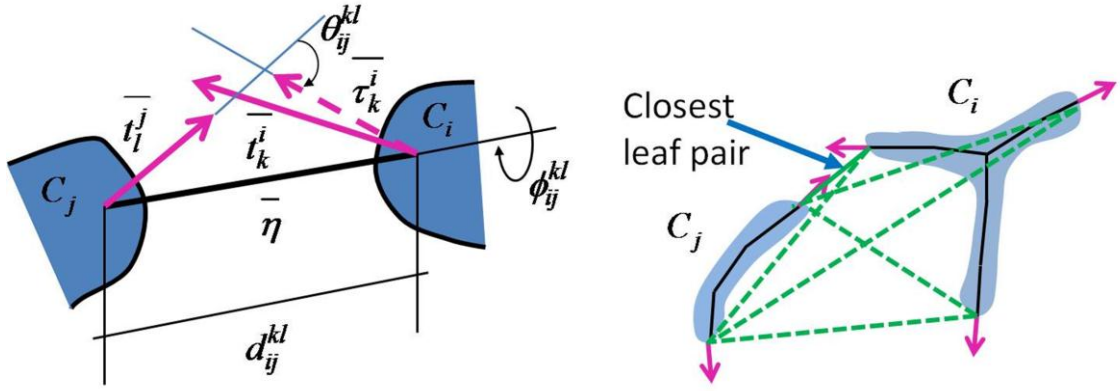
Fig. 4(b) shows the expected eigenvector directions for the best scale for a tubular element. We now have filamentary measure defined at each voxel that we can use as a transformed image. Fig. 5(a) shows the normalized filamentary image of the neuron from Fig. 1. Since the filamentary measure image is independent of the raw intensity of the neuron image stack, we can employ a global binary clustering of the filamentary measure into a foreground and background region. We have used a two-class clustering approach with maximization of inter-class to intra-class variance as in [24] for our binary clustering step. The binary clustered image is shown in Fig. 5(b) after suitably Gaussian convolving the filamentary image in Fig. 5(a).



(a) (b)

Fig. 7. (a) A single connected component (b) The medial tree of (a) with nodes shown in red, edges in green and leaf tangents in magenta.

We are at a position now where we have extracted possible neuron regions from the originally cluttered neuron image. The information at hand is a group of disjoint sets of 3D binary connected components which suggest



(a) (b)

Fig. 8. (a) The calculation of orientation and distance between leaf pairs. (b) 2 connected components C_i and C_j shown in blue, leaf tangents shown in red and medial trees in black. 6 pairs of leaf node connectivity (shown in dotted green) are tested for the closest distance between the components. The closest leaf pair is indicated.

possible neuron segments. The next challenge is to extract meaningful topological information out of this disjoint set of components.

C. 3D centerline extraction of the disjoint binary volumes (Step 2):

In order to determine the orientation of the isolated components obtained from step 1 and the possible connectivity between the different volumes, we need to investigate morphological properties of the binary volumes and use the properties to reliably interconnect the volumes. Since the binary volumes from step 1 consist mostly of pixels that belong to elongated tubular regions, it is reasonable to expect that the binary volumes themselves will have long skeletons or centerlines.

Previous research in 3-D skeletonization probes a number of potential approaches ([26], [27]). Following [28], we define the skeleton or centerline of a binary volume as a single-pixel thick graph that has the same topology as the original volume and intersects the cross section of the volume orthogonal to the skeleton at the center of area of the cross section. Fig. 6 demonstrates the concept of the 3D skeleton as used in Tree2Tree.

We employ a modified version of the algorithm given in [26] to find the 3D skeleton of the individual binary volumes. We assume 26-connectivity of the foreground (white) pixels and 6-connectivity of the background (black) pixels.

The 3D skeleton extraction algorithm employed here follows the sequential thinning algorithm presented in [26] that iteratively deletes border points (border points, following [26], can be loosely defined as peripheral voxels of a binary volume, deletion of which does not change the topology of the volume) of the binary volume in ordered passes from the Up, Down, North, South, East, West direction, always maintaining the order. The skeleton is accomplished when none of the passes successfully finds a border point in the binary volume. Fig. 6 (b) shows the skeletons of the binary volumes in Fig. 5(b), all presented in a composite image. Note that the individual binary volumes have isolated skeletons, and the close-up in Fig. 6(b) shows the gap between two individual skeletons.

D. Local Medial Trees (Step 3):

The output from step 2 is a set of single-pixel thick skeletons of the individual binary neuronal components, which we use as the initial evidence in constructing the individual neuron segments. Once a meaningful mathematical description of the connected components is generated according to their filamentous properties, orientation and

volume, Tree2Tree connects the individual components in a physically meaningful way to obtain the complete neuron.

Let $\mathcal{C} \equiv \{C_1, C_2, .., C_n\}$ be n connected components obtained from step 1. For each connected component, the raw skeleton obtained from step 2 is re-sampled with a user defined resolution r to generate individual medial trees. Each connected component can then be described by the following pair

$$C_j \equiv \{\mathcal{M}_j, \mathfrak{U}_j\} \tag{8}$$

where

$$\begin{gathered}\mathcal{M}_j \equiv \{V_j, E_j\}, V_j \equiv \left\{\bar{p}_j^1, .., \bar{p}_j^{m_j}\right\}, \\ E_j \equiv \left\{\left(\bar{p}_j^k, \bar{p}_j^l\right) | \text{ with some } k, l \in \{1,2, \ldots, m_j\}, k \neq l\right\}, \\ \text{and } \mathfrak{U}_j = volume\left(C_j\right)\end{gathered} \tag{9}$$

In (8), $\mathcal{M}_j$ is the medial tree of C_j, with the set V_j of m_j vertex nodes $\bar{p}_j^{m_j}$ (with adjacent nodes placed at resolution r) and the set E_j of $(m_j - 1)$ edges (since $\mathcal{M}_j$ is a tree). In (9), $volume\left(C_j\right)$ simply means the total number of voxels in the connected component C_j and therefore is a measure of the total space this connected component occupies in the image. Thus at the end of step 3 we have characterized the n neuronal segments (connected components) each of which is represented by its medial tree and volume. It is helpful to note here that a higher number of nodes in a medial tree points to a possibly elongated structure of the original connected component. Also, a relatively higher volume of a binary component gives evidence of the component belonging to a neuron. Tree2Tree utilizes both these properties to check the suitability of a component of being a neuronal branch.

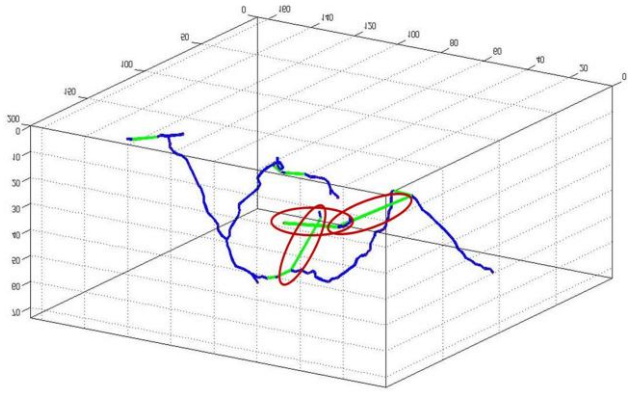


Fig. 9. Minimum spanning tree optimally connecting the individual medial trees (in blue) from Fig. 5(b) with green edges. The circled (with red) green edges signify erroneous branches contributed by clutter.

E. Global k-NN Graph (Step 4):

As previously discussed with regard to step 3, the local medial trees that have been obtained at the end of step 3 are pieces in a complete neuronal tree. The individual medial trees need to be connected in a compatible manner that

minimizes bending and torsional stress while aligning connected components, as well as prefers nearby trees to distant medial trees for interconnection. We utilize a graph theoretic approach in estimating the connectivity between the connected components in order to approximate a complete neuron. In order to evaluate the weighted distance between two connected components, we need to characterize the alignment and proximity between components.

For a connected component C_j with medial tree $\mathcal{M}_j$ that has f_j number of leaf nodes (a leaf node is a terminating node in a tree and is connected to only one other node), we can define f_j tangent vectors $\{\bar{t}_1^j, .., \bar{t}_{f_j}^j\}$ with $\bar{t}_l^j = \bar{p}_j^l - a(\bar{p}_j^l)$ ($a(\bar{p}_j^l)$ is the ancestor node for node $\bar{p}_j^l$ in $\mathcal{M}_j$). Figs. 7(a) and 7(b) shows the medial tree and the leaf tangents for a connected component.

If two connected components are in fact part of the same branch of a neuron, then we can expect at least one leaf pair (with each connected component contributing one leaf) to be close to each other and optimally aligned.

For a leaf pair $(\bar{p}_l^k, \bar{p}_j^l)$ between two connected components C_i and C_j, the potential alignment can be defined in the following way. The two tangent leaf tangents $\bar{t}_k^i$ and $\bar{t}_l^j$ the axis $\bar{\eta} = \bar{p}_j^l - \bar{p}_l^k$ will generally not lie on a common plane. Let $(\phi_{ij}^{kl}, \theta_{ij}^{kl})$ be a pair of angles such that

(*i*) ϕ_{ij}^{kl} is the minimum rotation required along the axis $\bar{\eta}$ to bring the leaf tangent $\bar{t}_k^i$ to a configuration $\overline{\tau_k^i}$ such that $\bar{\tau}_k^i$, $\bar{\eta}$ and $\bar{t}_l^j$ are coplanar and

(*ii*) θ_{ij}^{kl} is the minimum angle of rotation of the leaf tangent $\bar{t}_l^j$ on the plane determined by $\bar{\eta}$ and $\bar{t}_l^j$ to bring $\bar{t}_l^j$ in direct opposition to $\bar{\tau}_k^i$.

Out of several discrete possible pairs of such $(\phi_{ij}^{kl}, \theta_{ij}^{kl})$ choose a pair that minimizes $|\phi_{ij}^{kl}| + |\theta_{ij}^{kl}|$. It can be noticed that the chosen ϕ_{ij}^{kl} can be regarded as the twist or torsional angle and θ_{ij}^{kl} can be regarded as the bending angle to align the two leaf nodes $\bar{p}_l^k$ and $\bar{p}_j^l$. Fig. 8 illustrates the calculation of the angle pair $(\phi_{ij}^{kl}, \theta_{ij}^{kl})$. Here, let $d_{ij}^{kl} = \sqrt{|\bar{p}_l^k - \bar{p}_j^l|^2}$ denote the Euclidean distance between the leaf pair.

Then, for connected components C_i and C_j, we define the distance $\mathcal{D}_{ij}$ between them as

$$\mathcal{D}_{ij} \equiv \min_{k \in \{1,..,f_i\}, l \in \{1,..,f_j\}} \{\lambda d_{ij}^{kl} + (1-\lambda)(|\phi_{ij}^{kl}| + |\theta_{ij}^{kl}|)\} \tag{10}$$

In (10), λ is a weight parameter between 0 and 1 that blends contributions from the Euclidean distance and the angle components between a particular leaf pair. We have observed that the leaf tangents are more susceptible to measurement noise and have therefore assigned a value of $\lambda = {}^{2}/_{3}$ throughout our experiments. For a leaf pair (k, l) that gives the minimum distance in (10), the distance d_{ij}^{kl} between leaf nodes is small and the tangent vectors are aligned in such a way that minimum amount of torsion ϕ_{ij}^{kl} and bending θ_{ij}^{kl} is required to align them. See Fig. 8(b) for understanding the distance calculation in (10).

At this point, we have a way to estimate compatibility for interconnection between two connected components. Considering each connected component as a node in a graph of connected components, we can pick a set of interconnections between the connected components in such a way that the cost of connecting the components is minimal.

A global *k*-nearest neighbor graph $\mathcal{G}$ with the connected components C_j themselves considered as nodes is used to complete the neuron. $\mathcal{G}$ is defined as

$$\begin{aligned} \mathcal{G} &\equiv \{\mathcal{V}, \mathcal{E}\}, \mathcal{V} \equiv \{C_1, .., C_n\}, \\ \mathcal{E} &\equiv \left\{\left(C_i, C_j, \mathcal{D}_{ij}\right) | \text{ with some } i, j, \in \{1,2, \dots, n\}, i \neq j\right\} \end{aligned} \tag{11}$$

Edge $\left(C_i, C_j, \mathcal{D}_{ij}\right)$ with edge weight $\mathcal{D}_{ij}$ only exists if C_j is one of the *k* nearest nodes to C_i based on the distance $\mathcal{D}_{ij}$ calculated from (10) (Note that $\mathcal{G}$ is directed).

In our experiments, we have taken *k* as the lowest integer such that the resultant global *k* -NN graph is connected. An alternate strategy is to specify *k* as the total number of connected components after step 1. This might affect subsequent computational speed but is conceptually simpler and precise.

F. Global Connectivity Tree (*Step 5*):

The global connectivity graph from step 4 gives different possibilities of interconnection between the connected components. Since we assume that our neuronal structure does not contain loops, we can now form an optimal connectivity tree by constructing a minimum spanning tree $\mathcal{M}$ [29] of the *k*-NN global graph $\mathcal{G}$ such that the sum of the edge weights $\mathcal{D}_{ij}$ over the entire tree is a minimum. This result is a connected tree composed of the smaller

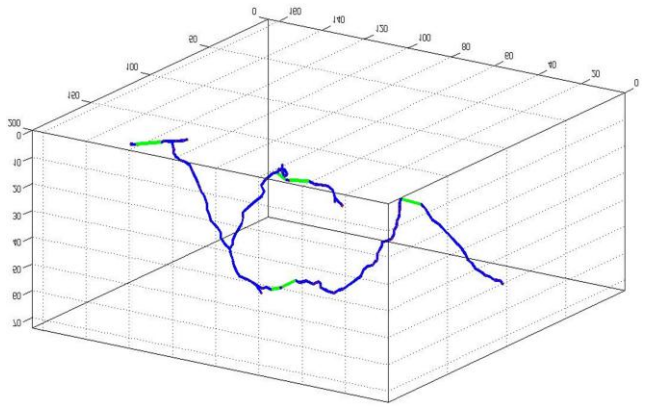


Fig. 10. Neuronal tree from Fig. 8 passed through the $\alpha - \beta$ graph pruning algorithm.

medial trees arising from each connected component. Together, in Tree2Tree, a global tree connectivity of the neuronal structure is yielded.

G. $\alpha - \beta$ *Graph Pruning* (*Step 6*):

The Hessian based neuron enhancement technique used in step 1 will still allow some clutter to pass through to step 2, possibly because small non-neuronal elongated fragments might be present as an artifact of the specimen preparation process. In other words, the vertex set $\mathcal{V} \equiv \{C_1, .., C_n\}$ from step 5 might have some connected components contributed by image clutter. Fig. 9 shows the minimum spanning tree (after step 5) of the connected components of the neuron from Fig. 5(b). Note that the individual medial trees of the connected components are colored in blue, and the green edges represent the best possible connections between the medial trees according to (10). We have also circled the connections in red that connect small clutter fragments to the actual neuron.

At this point, Tree2Tree removes the final pieces of clutter components by utilizing a simple assumption: connected components that are too costly to connect to the neuronal tree compared to the relative contribution they have in the total neuronal volume are considered as clutter. We formalize the concept below mathematically.

If we define the node-likelihood $\mathcal{W}(C_i)$ of each connected component C_i as

$$\mathcal{W}(C_i) = \left|E_j\right| + \sqrt[3]{\mathfrak{U}_j} \tag{12}$$

such that the node likelihood $\mathcal{W}(C_i)$ is a sum of the length $\left|E_j\right|$ of the medial tree E_j (E_j is defined in (9)) and the third root of the node volume $\mathfrak{U}_j$ ($\mathfrak{U}_j$ is also defined in (9)). As a result, components with longer medial trees and higher volumes have higher likelihood of belonging to the neuron and contributing to the total neuronal mass. We can expect connected components arising from clutter to have relatively low node-likelihood, at the same time having poorer proximity and alignment with their neighboring connected components, thus contributing to a high edge weight. Therefore, the global tree from step 5 is used as an input to the $\alpha - \beta$ graph pruning function $\mathcal{F}$, which is defined as

$$\mathcal{F}(\mathcal{M}) = \mathcal{M}_f \text{ where } \mathcal{M}_f = \{\mathcal{V}_f, \mathcal{E}_f\}, \mathcal{M}_f \text{ is connected,} \tag{13}$$

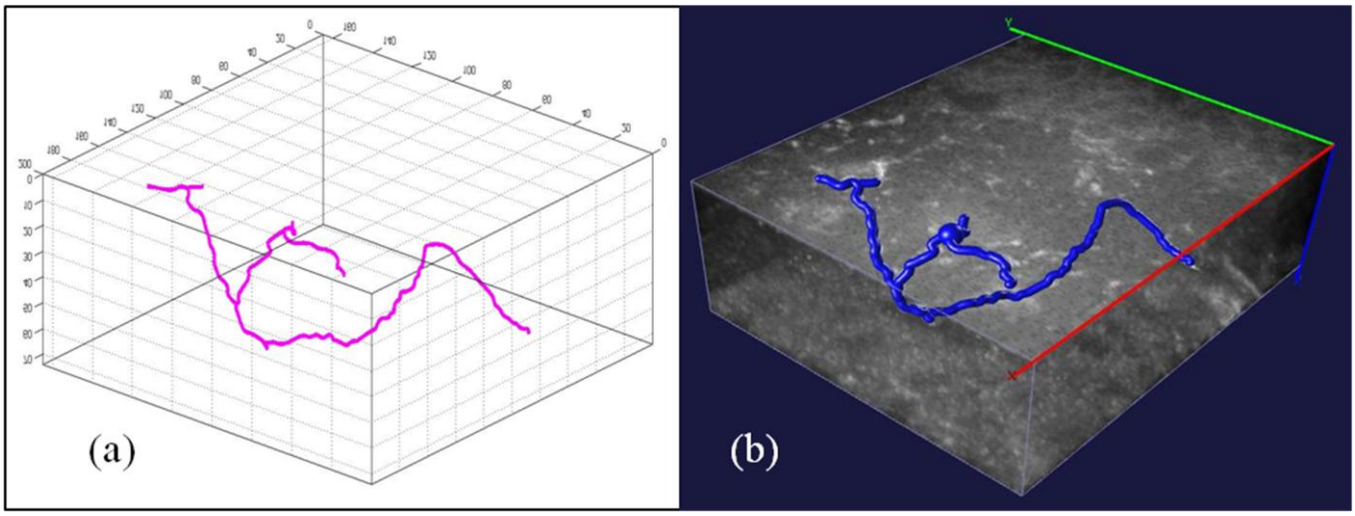


Fig. 11. (a) Pruned tree from Fig. 9 after global node numbering and spline fitting. (b) Spline fit tree from Fig. 10(a) overlaid on the original neuron image in Fig. 1.

$$\mathcal{M}_f \subseteq \mathcal{M}, \mathcal{V}_f = \mathcal{V} \setminus \mathcal{V}_N, \mathcal{V}_N = \{C_{i_1}, .., C_{i_k}\} \subseteq \mathcal{V} \text{ such that}$$

$$\mathcal{V}_N = arg \min_{\mathcal{V}_N \subseteq \mathcal{V}} |\mathcal{V}_N|$$

$$\text{satisfying } \frac{\sum_k \mathcal{W}(C_{i_k})}{\sum_j \mathcal{W}(C_j)} \geq \alpha$$

$$\text{and } \frac{d(\mathcal{M}_\mathrm{f})}{d(\mathcal{M})} \leq \beta$$

Here $d(\mathcal{M})$ denotes the sum of edge weights of the tree $\mathcal{M}$.

(13) is explained as follows: $\mathcal{M}_f$ is the largest connected subtree of the original minimum spanning tree $\mathcal{M}$ such that $\mathcal{M}_f$ is constructed by deleting the maximal vertex set $\mathcal{V}_N$ from the original vertex set $\mathcal{V}$ with the requirement that the subtree $\mathcal{M}_f$ has a proportional node weight $\frac{\sum_k \mathcal{W}(C_{i_k})}{\sum_j \mathcal{W}(C_j)} \geq \alpha$ and a proportional edge-weight $\frac{d(\mathcal{M}_\mathrm{f})}{d(\mathcal{M})} \leq \beta$. In other words, the $\alpha - \beta$ graph pruning removes a minimal set of connected components from the tree $\mathcal{M}$ such that the proportional net node likelihood does not fall below α but the proportional net edge weight is reduced below β. The pruning thus achieves a tradeoff between highly likely nodes and highly likely connections. Fig. 10 shows the pruning of the circled (in red) edges and corresponding connected components of the tree in Fig. 9 by passing through the $\alpha - \beta$ graph pruning algorithm in (13), using $\alpha = 0.7$ and $\beta = 0.2$.

The $\alpha - \beta$ pruning algorithm is the only place in our algorithm where the user manually inputs his preference. It gives the user considerable control over setting the limits of acceptable clutter in the neuron segmentation and simple tests over a few neurons in a database is enough to set the parameters of the $\alpha - \beta$ graph pruning algorithm

efficiently. Although an automatic optimization of the $\alpha - \beta$ parameters on a database might make our algorithm totally automated, we believe that giving the neurobiologist a few quick and easily tunable parameters might make the segmentation more realistic and useful. In our tests, $\alpha = 0.7$ and $\beta = 0.2$ has worked reliably in our dataset of images.

H. Global Node Numbering and Spline Fitting (Step 7):

Although we have obtained the final neuron connectivity from step 6, we need to output a global node-numbered tree in order to generate the overall neuronal morphology. Our final step consists of merging the individual medial trees $\{\mathcal{M}_j\}_{j=1}^{|\mathcal{V}_f|}$ from step 6 using the connectivity information from the global tree $\mathcal{M}_f$ to form the final neuronal tree $\mathcal{N} = \cup_{j=1}^{|\mathcal{V}_f|} \mathcal{M}_j$. A global node number is assigned to the nodes of the merged graph and cubic splines are fit to individual branches of $\mathcal{N}$ to generate the neuronal morphology. Fig. 11(a) shows the final version of the spline fitted tree from Fig. 10. In Fig. 11(b), we have overlaid the spline fitted final neuron on the original neuron image of Fig. 1. The radial width of the neuron in Fig. 11(b) has been arbitrarily set to one pixel for visualization.

Finally, we have all the segments of our method in place to automatically trace and generate the branch structure of neurons from 3D confocal microscopy datasets. Tree2Tree involves a few manually tuned parameters such as α and β for the graph pruning step that can be kept constant for a vast set of neurons with similar image acquisition parameters. Consequently, our algorithm is an almost fully automatic neuron tracing method that reliably distinguishes between neuron and background clutter and uses geometric and physical constraints to connect partial information into an entire neuron.

I. Discussion of the Tree2Tree Algorithm:

The novelty of Tree2Tree lies in the following four features:

(1) Current neuron tracing algorithms use some kind of preprocessing step to remove noise and clutter from the 3D neuron images before actual application of the algorithm. Our application of the Hessian based geometric clutter removal (step 1) and pixel classification follows a similar line. Yet, it is novel in the way we have devised our filamentary measure $\mathcal{N}(\mathbb{p})$. It is particularly suited to our problem of brighter neurons on a darker background that have clutter in the form of pieces of round stray tissues.

(2) In Tree2Tree, the neuron segmentation/tracing problem is posed in a graph theoretic context with an efficacious definition of graph nodes and their edge connectivity. The graph nodes and edges do not reside on the image space but on a completely transformed space where nodes represent features of neuron segments (for example, the segment trees) and edges represent connectivity constraints of the features. Therefore, the graph theoretic neuron completion takes place in a transformed feature space ($\mathcal{C} \equiv \{C_1, C_2, .., C_n\}$) which denotes the space of neuronal segments or connected components.

Branch detection and bifurcation have proven to be a formidable challenge with edge-connection or seed following algorithms. In contrast, the minimum spanning tree solution in the transformed feature space naturally admits the branching of nodes and does not require heuristics.

(3) We have also developed a graph theory based graph pruning algorithm ($\alpha - \beta$ graph pruning algorithm) that poses the image clutter removal problem in a new paradigm. Instead of morphological or intensity-based filtering, we use the neuronal geometry to distinguish unlikely segments contributed by clutter.

In the transformed feature space, unlikely neuronal segments behave like outliers based on their feature space inter-segment distances. This feature space inter-segment distance includes geometric features like Euclidean distance, bending and torsion of neuronal branches. The clutter removal is implemented as a graph pruning algorithm that achieves a tradeoff between likely neuronal segments (nodes in the feature space) and their sparsity in the feature space.

We believe that this is a new approach to image clutter removal and is more suitable to the problem of neuron segmentation than direct application of directional and smoothing filters to the raw neuron image since they do not take into account the global structure of the neuron.

In the next section, we demonstrate both qualitatively and quantitatively, the results of applying Tree2Tree to *Drosophila* neurons.

III. EXPERIMENTS

Our neuronal dataset is obtained from the ventral nerve cord (VNC) of the central nervous system (CNS) of the Drosophila. The VNC contains about 5000 cells [5]. Structurally, the VNC contains a set of fused sub-esophageal segments, three large thoracic segments and seven central reiterated and relatively simple abdominal segments. About 130 neurons constitute each abdominal hemi-segment of the VNC. We have chosen 16 cells from this set of

130 neurons for the purposes of demonstration of our neuron tracing algorithm. These 16 cells contain single GFP-labeled axon terminals from *Drosophila* larval abdominal sensory neurons.

Each of the 3D images contains a single Green Fluorescence Protein (GFP)-labeled neuron from one dorsal view. Each 3D image is collected into three channels - serotonin, fasII and GFP within approximately 250 focal planes from dorsal to ventral, each having a size of 672x512 pixels. The GFP is expressed in single cells using a heat-shock activated GFP flip-out scheme [22]. Only the GFP channel has been used as input in these results.

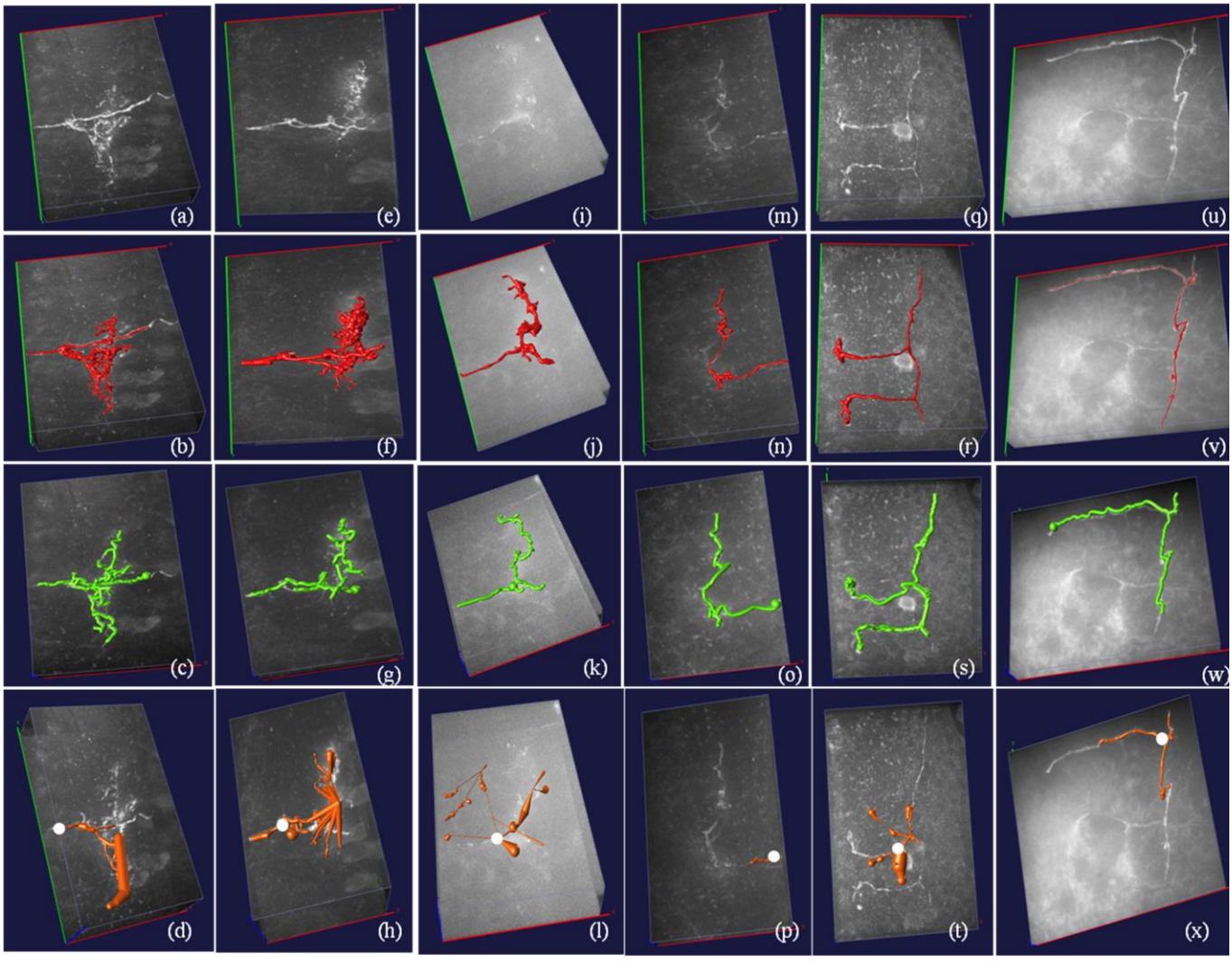


Fig. 12. Results of Tree2Tree applied to 3D confocal microscopy images of *Drosophila* neurons/neurites from the VNC. Each column shows Tree2Tree applied to one particular neuron. The first row shows the original 3D image of the neuron/neurite. The second row shows the corresponding (to the first row) manually segmented (ground-truth) neuron/neurite in red with the help of the software V3D [33]. The third row shows the result of applying Tree2Tree to the corresponding neurons/neurites in green. Note that the manual segmentation in row two has a manually estimated radial width of the neuron whereas in case of Tree2Tree (third row) we have arbitrarily set the radial width to 1. The fourth row shows the result of applying NeuronStudio [NeuronStudio] to the same six neurons from row 1 (in orange). The starting manual seed point is shown with a white circle in each of the images in the fourth row. For each NeuronStudio trace, the starting seed point is placed at ten different positions along the neuron and the longest trace with the best visual compatibility with the ground-truth (in row two) has been retained.

A. Setting Up Tree2Tree and Comparison Algorithms for Experimental Validation

In order to validate Tree2Tree as an automatic tracer of 3D neurons, we have applied Tree2Tree and two other freely available software to automatically trace the neurons in our dataset. We have investigated many of the free software listed in [30] available for the purposes of tracing neurons from microscope images. To have a fair comparison, the scope of the comparison algorithm has to closely match the scope of Tree2Tree.

The scope of Tree2Tree can be defined as follows – automatic tracing of *three dimensional neuronal tree* with *no manual placement of seed/tracing points* and *automatic determination of bifurcation/junctions* in the 3D neurons. Excluding software which work only for 2D neuron images/image stacks, we have determined that the software *NeuronStudio* [31] is the closest match in scope to Tree2Tree. NeuronStudio can automatically trace 3D neuronal trees as well as automatically determine bifurcations/junctions in a 3D neuron, *although* one seed point has to be placed manually for the tracing to begin (a manual intervention that Tree2Tree avoids). Since the tracing in NeuronStudio is heavily dependent on choice of a good starting point, we have kept the default settings for the tracing but have manually initialized the seed point at ten different positions in each 3D neuron. Out of the ten different tracings for each neuron, the longest trace (in terms of tree nodes) with the best visual compatibility as compared with the ground-truth has been retained for comparison with Tree2Tree. We note that NeuronStudio has functionalities for manual editing and capability of manually joining separate portions of the same neuron traced with separate starting seed points into a longer neuron. We have not used any manual editing and joining of subtrees in NeuronStudio as we specifically try to avoid placing multiple seed points on several positions manually and perform visual compatibility based editing. That would severely limit the scope of Tree2Tree and would be an unfair comparison.

We have also tested the freely available software *Simple Neurite Tracer* (SNT) [32] on our dataset. SNT works on 3D neurons but is semi-manual in nature. More specifically, SNT cannot automatically determine bifurcation points/junctions in the 3D neuron. It is essentially a path tracer where the end points of a path have to be manually specified by the user after which SNT connects an optimal path between the two end-points. Despite the heavy manual intervention needed in SNT, we have considered this algorithm because it is an easily usable software that works on 3D neurons and performs relatively well in tracing a path between two points. We have anecdotally

compared the performance of SNT with Tree2Tree in cases where we are not interested in bifurcation points of a neuron (section III-*E*).

Fig. 12 shows the results of applying Tree2Tree to six neurons from our dataset. Each column shows Tree2Tree applied to one particular neuron. The first row shows the original 3D image of the neuron/neurite. The second row shows the corresponding (to the first row) manually segmented (ground-truth) neuron/neurite in red with the help of the software V3D [33]. The third row shows the result of applying Tree2Tree to the corresponding neurons/neurites in green. The fourth row shows the result of applying NeuronStudio (in orange) to the same neurons with the starting seed points shown as white discs. Since a good comparison metric for comparing two trees with regards to geometry and connectivity pattern is still a topic of active research (see an introductory work in [34]), we believe visual comparisons can serve as an useful cue to understanding the success or failure of an algorithm.

Note that the manual segmentation in row two has a manually estimated radial width of the neuron whereas in case of Tree2Tree (third row) we have arbitrarily set the radial width to one only for visualization purposes. We remind the reader once again that currently Tree2Tree traces the centerline of the neuron but does not estimate the radial width. These results show different unique characteristics of our algorithm which are described below.

B. Natural Framework for Bifurcations:

Figs. 12(a)-(d) and 12(e)-(h) show the application of Tree2Tree and NeuronStudio to the dendritic termination of two sensory neurons from the *Drosophila* VNC. It takes a human observer upwards of 20 hours to approximately populate a 3D image stack with neuronal nodes and specify their inter-connectivity for these kinds of terminations. And yet, due to the unavoidably cluttered condition of the specimen and the complicated branch structure, the manually estimated interconnectivity is often biased and not highly dependable. The highly complicated branch structure can be observed from the manually segmented neurons in figs. 12(b) and 12(f) (in red).

Given the especially cluttered specimens, the lack of contrast and clear branch edges, regenerating the level of detail and interconnecting branch patterns for these neurons by an automated neuron tracing method is highly challenging. Yet, Tree2Tree performs reasonably well in figs 12(c) and 12(f) (in green). Compared to Tree2Tree, NeuronStudio underperforms in finding the proper branching substructure (Figs. 12(d) and 12(h)) as the seed based tracer in NeuronStudio gets thrown off by the close proximity and irregular branch thickness of the branch

substructures. Moreover, there is no heuristic estimation for determining neuron branching, which makes Tree2Tree tractable and free from tedious parameter adjustment.

Not only the major branching patterns were correctly predicted by Tree2Tree, the computation time for each tracing was less than 1 minute on an Intel core 2 duo 3.2 GHz desktop. Compared to current state of the art neuron tracing algorithms which work on relatively uncluttered data having simpler branching structures, we believe this is a significant improvement.

C. Tracing Neuronal Images with Highly Fragmented Stems:

The reader can observe in Figs. 12(i) and 12(m) that the neuronal branches and sub-branches are highly fragmented with lack of any clear orientation and crisp edges. Moreover, Fig. 12(i) contains significant background scatter that makes the contrast poor and Fig. 12(m) is characterized by low brightness that makes the neuronal stems almost indistinguishable from the background.

Figs. 12(j) and 12(n) show the manually estimated ground-truth for the neurons in Figs. 12(i) and 12(m) respectively. Because of the highly fragmented stems and lack of contrast or orientation, it takes a high amount of time for the manual segmentation with subsequent low confidence in several of the populated nodes. The result is often a costly overestimation of the neuronal volume.

The results of applying Tree2Tree to Figs. 12(i) and 12(m) are shown in Figs. 12(k) and 12(o) (in green). The reader can notice that the results are neuronal nodes are not overestimated and it closely mirrors the ground truth from Figs. 12(j) and 12(n) respectively.

It is especially interesting to note that the highly fragmented stems indeed give rise to a high number of medial smaller trees in step II-*D* in the previous section. Yet, the missing information (the connections between the medial trees) is intelligently reconstructed by Tree2Tree in step II-*H* for global connectivity determination in the previous section. This exhibits the strength of Tree2Tree in completing the missing information despite poor quality of data.

This kind of data is especially problematic for NeuronStudio. Since NeuronStudio adopts a seed following strategy, poor quality or missing data in portions of the neurons are capable of terminating the seed following completely. Figs. 12(l) and 12(p) demonstrate this point. Heavy background scatter in Fig. 12(i) throws the seed follower off the actual neuron in Fig. 12(l) and missing neuronal portions in Fig. 12(m) stops the seed following prematurely in Fig. 12(p).

D. Neurons with Heavy Background Clutter:

Figs. 12(q) and 12(u) show examples of two neurons with heavy background clutter. Notice that along with the neuronal central branches in these two images, the background is high in both the images. This contributes to erroneous classification into foreground voxels after the preprocessing in step II-*A* in the previous section.

The human observer can reliably estimate the background tissue clutter based on his/her pre-existing knowledge of background tissue shape and location (see the ground truth segmentation in red in Figs. 12(r) and 12(v)). However, distinguishing the same is a highly difficult task for an automated algorithm.

The Tree2Tree-traced neurons shown in Figs. 12(s) and 12(w) demonstrate good agreement with the manual segmentation of Figs. 12(r) and 12(v). This robust performance is due to the fact that the novel $\alpha - \beta$ graph pruning algorithm performs a graph based clutter removal.

The unique point here is posing the clutter problem in a graph theoretic paradigm where the graph nodes and their edge connectivity lie in a different space. Inference of unlikely nodes based on the amount of connectivity weight they introduce immediately empowers us with a tool that cannot be conceptualized in direct image filtering or seed tracing algorithms. We will elaborate on this point further in the next section.

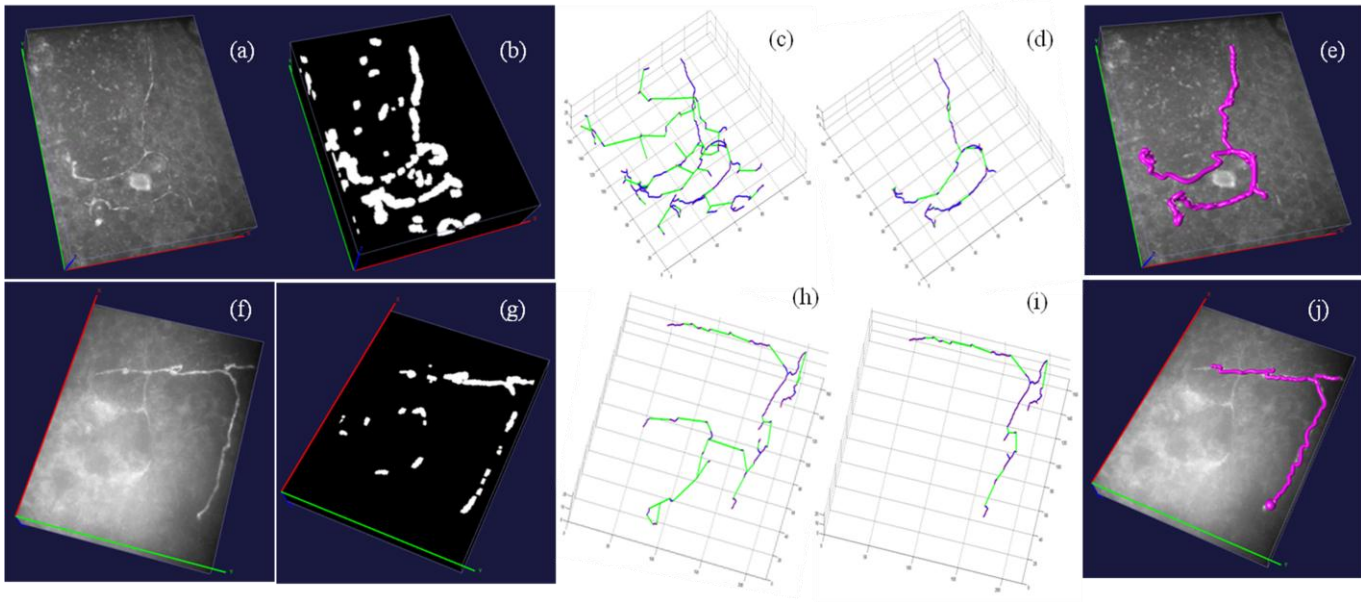


Fig. 13. Demonstration of the efficacy of the $\alpha - \beta$ graph pruning algorithm. The leftmost column shows two neuron images with heavy background clutter. The second column shows the respective neurons after the Hessian based preprocessing step in 1. The third column shows the global connectivity trees of the two neurons respectively after step 5. The fourth column shows the pruned trees after passing through the $\alpha - \beta$ graph pruning algorithm with $\alpha = 0.7$ and $\beta = 0.2$. The last column shows the final splined and traced neurons in pink.

The result of applying NeuronStudio to the neurons in Figs. 12(q) and 12(u) is shown in Figs. 12(t) and 12(x). The high background clutter from residual tissue in Fig. 12(g) again makes seed following especially problematic for NeuronStudio, as after a brief period of correct tracing the seed follower gets distracted by the closely lying background tissue. The performance of NeuronStudio on the neuron in Fig. 12(u) is comparatively better as in it follows the neuron closely for a longer length, but increasing loss of contrast at the terminal edges of the neuron end the tracing prematurely.

E. Efficacy of $\alpha - \beta$ Graph Pruning:

As mentioned in the previous paragraph, the $\alpha - \beta$ graph pruning algorithm poses the problem in a graph theoretic context such that inferring the validity of a neuronal node can be done by trading it off against the amount of edge weight it introduces. This concept is illustrated in Fig. 13.

Figs. 13(a) and 13(f) show the same neurons as in Figs. 12(m) and 12(p) respectively. After the Hessian based

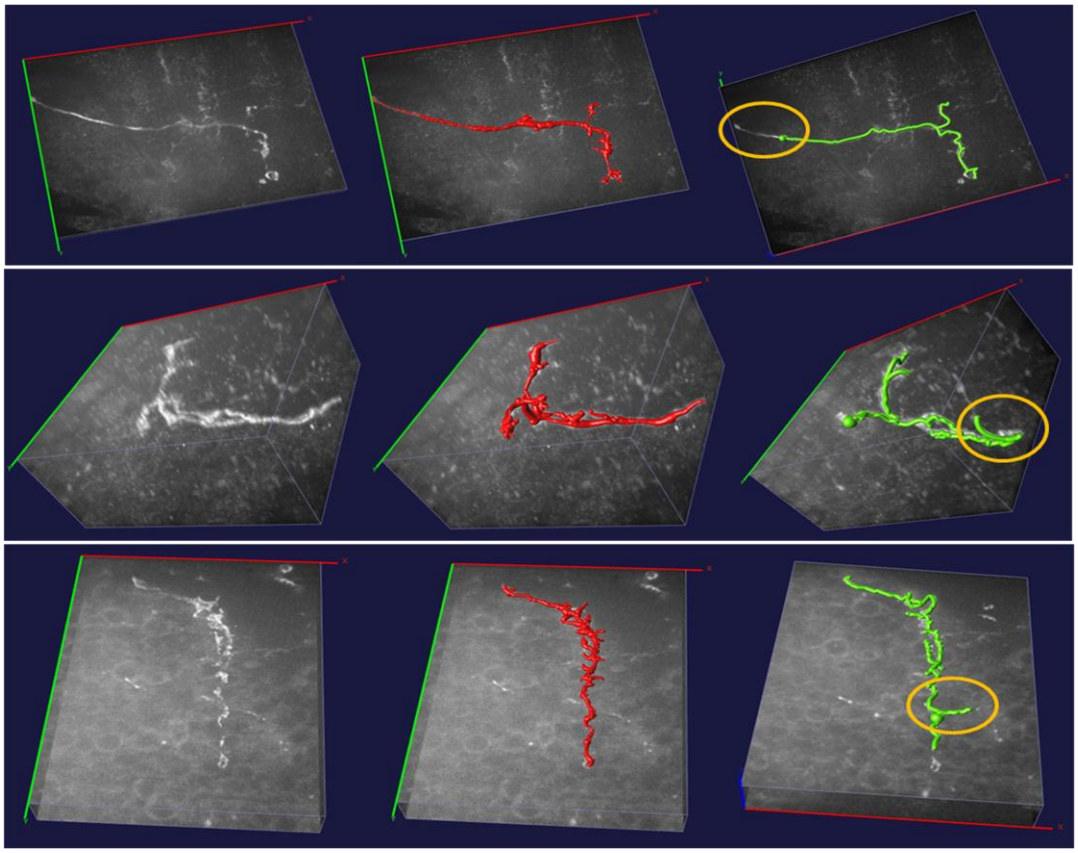


Fig. 14. Occasional underestimation or overestimation of branches by Tree2Tree. The three rows show Tree2Tree applied to three neurons. The leftmost column shows the original neuron images, the middle column shows the manually estimated ground-truth in red, and the rightmost column shows the neurons traced by Tree2Tree in green. The over(under) estimated branches are highlighted with yellow circles.

preprocessing and binary clustering applied to the two images as in section II-A, we obtain the two foreground images in Figs. 13(b) and 13(g) respectively. Note that the high clutter in the two neuron images contributes to the numerous small isolated foreground 'islands' or connected components in Figs. 13(b) and 13(g).

The preprocessing failed to remove these components because they satisfied the criteria of elongated tubular structures as permitted by step 1, although these were contributed by stray components from the background. Most direct image based clustering methods will fail to remove these components because they indeed look like neuronal parts in a geometric sense.

At the end of step 5, the global connectivity trees of the two neurons are shown in Figs. 13(c) and 13(h) respectively. Note how many extra components are present in these two figures. After pruning the two global connectivity trees in Figs. 13(c) and 13(h) respectively with the $\alpha - \beta$ graph pruning algorithm with $\alpha = 0.7$ and $\beta = 0.2$, we obtain the trees shown in Figs. 13(d) and 13(i) respectively.

The removal of the clutter is due to the fact that in this space of graph nodes (denoted by weight of neuronal connected components) connected by edges (denoted by the ease of geometric connection between nodes), we achieve a tradeoff between the weight of nodes and the cost of connecting them with edges. This approach, we believe, is a novel yet natural way to perform clutter removal in images of neurons.

F. Missed Branches and Overestimation:

The high complexity of branch connectivity, broken neuronal stems and background clutter invariably lead to tedious and difficult manual delineation in the process of constructing ground truth. It is therefore only natural for any automated neuron tracing algorithm to over or under-estimate neuronal branches in extreme cases.

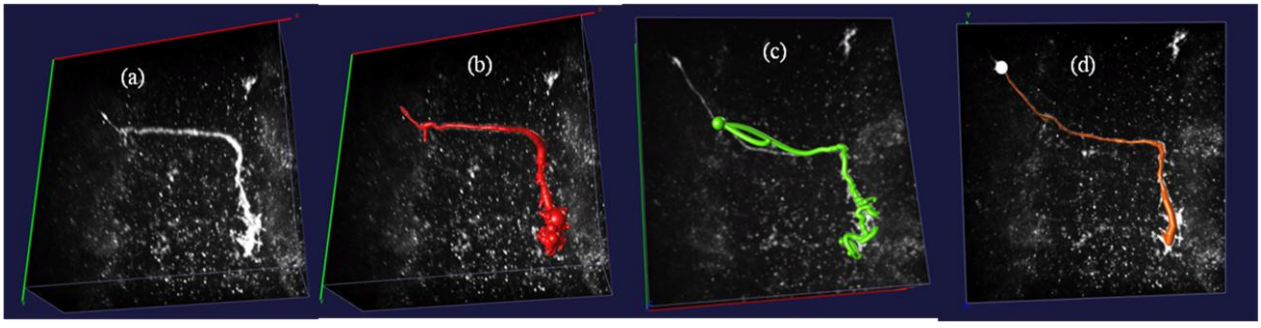


Fig. 15. A 3D neuron image where NeuronStudio performs better than Tree2Tree. 15(a) shows the original neuron image, and 15(b) shows the manual segmented ground-truth in red. 15(c) shows the tracing by Tree2Tree in green and 15(d) shows the trace by NeuronStudio with seed point shown in white. The spotted debris close to the neuron puts considerable load on Tree2Tree's graph based clutter removal algorithm, but a correct initialization of the seed point along the neuron in NeuronStudio helps in a relatively distraction free seed following.

We consider a branch to be underestimated if the missing length of the branch is more than 10% of that particular branch when compared to the same branch in the ground truth. Similarly, we consider a branch to be over-estimated if that branch is totally absent in the corresponding ground-truth and the total length of the branch is more than 10% of the longest branch in that neuron.

Fig. 14 shows the branch over(under)-estimation by Tree2Tree in some cases. In the top row in Fig. 14, the very narrow dimension of the neuronal branch at the very end of the neuron image (leftmost in top row) misguides the Hessian clustering algorithm to overlook it. Compared to the ground truth image in top row middle column (in red), note the missed end portion in the traced neuron (last image in top row, in green) circled in yellow.

A similar observation shows that in both the cases depicted in the middle and bottom rows in Fig. 13, clutter, low contrast, broken neuronal stems and their irregular geometric shapes misguide Tree2Tree to overestimate branches (circled in yellow).

G. Visual Comparison with Existing Neurite Tracers

Section III-*A* has explained the reasons for the choice of NeuronStudio and SNT as comparisons with Tree2Tree. Fig. 12 has shown the difficulty of NeuronStudio in employing a strategy of seed tracing when the image data has background clutter and missing information along the branches (possibly because of difficulty in specimen preparation). We have found NeuronStudio to outperform Tree2Tree in one neuron image (Fig. 15(a)) in our dataset where the image had a consistently bright central neuronal branch but had high density of bright ‘spots’ lying very close to the neuronal branch. Tree2Tree can avoid background clutter to a certain extent (Fig. 15(b)) but a correct manual seed placement for NeuronStudio helps in reliable seed tracking as the central bright filament avoids

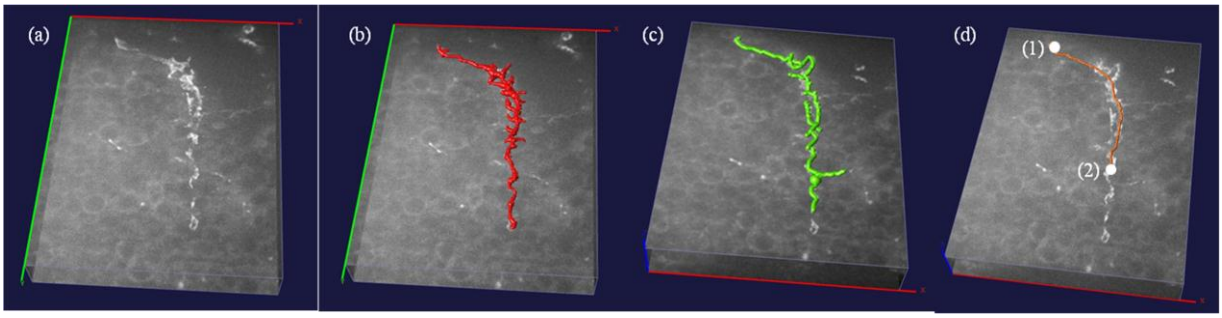


Fig. 16. An example showing performance of Tree2Tree versus SNT [32]. 16(a) shows the original neuron image, and 16(b) shows the manual segmented ground-truth in red. 16(c) shows the tracing by Tree2Tree in green and 16(d) shows the trace by SNT (in orange) with the path end points shown in white. (1) in 16(d) is the first end point and (2) in 16(d) is the second end point of the traced path. End point (1) has been kept constant and end point (2) has been gradually moved away from (1) along the neuron till the point when SNT breaks down.

distractions successfully. In all the other fifteen neurons, Tree2Tree outperforms NeuronStudio in a visual check between the trace and the available ground truth. The low contrast, missing segments, heavy background clutter and irregular branch geometry in our neuron images make seed following in NeuronStudio difficult.

SNT cannot automatically estimate bifurcation or junctions and requires manual intervention in placing end points of paths, and therefore is not an automatic neurite tracer with regards to the scope of Tree2Tree as described in section III-*A*. Yet, we have compared Tree2Tree with SNT because SNT performs quite well in tracing a path between two end points when two appropriate end points are manually chosen by the user. Fig. 16 shows the performance of SNT on one neurite in our dataset. Fig. 16(a) shows the original neurite image, and it can be seen that this neurite does not have any major bifurcation from the central stem and therefore can be considered to be one single path. Fig. 16(b) shows the manually segmented ground truth in red.

Tree2Tree traces the neuron in Fig. 16(a) with reasonable accuracy in Fig. 16(c) (shown in green). Two end points designated by (1) and (2) in Fig. 16(d) have been manually placed on the neuron and SNT has been applied to trace the neuron. End point (1) has been kept fixed and then end point (2) has been manually moved farther and farther

Neuron #	Overestimated Branches by Tree2Tree	Underestimated Branches by Tree2Tree	Tree2Tree Mean Absolute Error (μ) %	NeuronStudio [] Mean Absolute Error (μ) %
1	1	1	14.56	12.32
2	1	None	3.41	13.36
3	Too many/small branches	Too many/small branches	1.77	18.8
4	None	None	1.42	38.37
5	Too many/small branches	Too many/small branches	2.23	8.76
6	None	1	2.91	17.01
7	None	1	2.72	35.17
8	None	None	2.02	9.62
9	1	None	2.72	15.66
10	None	None	1.43	12.89
11	None	None	4.01	28.24
12	None	None	0.99	17.49
13	None	None	2.52	24.93
14	None	1	2.93	14.73
15	None	None	2.02	16.93
16	1	None	1.46	21.79

Table 1. Overall performance of Tree2Tree and NeuronStudio [Neuronstudio] on our dataset of sixteen neurons from the Drosophila VNC showing the mean absolute error percent (μ), number of over-estimated and underestimated branches.

away from (1) along the neuron till SNT cannot trace the intermediate path any more. From Fig. 16(d), it can be seen that SNT can trace a considerable path of the neuron (in orange), but it cannot obviously estimate the smaller bifurcations along the central path and also gives a shorter total length of the neuron as compared to Tree2Tree in Fig. 16(c).

H. Quantitative Summary of Neuron Tracing Results:

In the analysis of the results, the ground truth trace is represented by the set $\mathcal{G}$ consisting of n landmarks, whereas the Tree2Tree trace is represented by set $\mathcal{T}$ consisting of m graph nodes. For a landmark $g \in \mathcal{G}$ we denote $C_{\mathcal{T}}(g) \in \mathcal{T}$ as the closest point in $\mathcal{T}$ with respect to g and for a landmark $t \in \mathcal{T}$ we denote $C_{\mathcal{G}}(t) \in \mathcal{G}$ in the same way. Here, $C_{\mathcal{T}}(g)$ and $C_{\mathcal{G}}(t) \in \mathcal{G}$ is calculated as

$$C_{\mathcal{T}}(g) = \arg\min_{p \in \mathcal{T}} |p - g|_{\mathbb{L}_1} \text{ and}$$
$$C_{\mathcal{G}}(t) = \underset{q \in \mathcal{G}}{arg\,min}\, |q - t|_{\mathbb{L}_1}. \tag{14}$$

Also, assuming that the neuron image has height, width and elevation of h, w and e respectively, we define our normalized mean absolute error μ as

$$\mu = \frac{1}{n}\sum_{g \in \mathcal{G}} \frac{|g - C_{\mathcal{T}}(g)|_{\mathbb{L}_1}}{h + w + e} + \frac{1}{m}\sum_{t \in \mathcal{T}} \frac{\left|t - C_{\mathcal{G}}(t)\right|_{\mathbb{L}_1}}{h + w + e}. \tag{15}$$

(15) simply collects all the minimum absolute ($\mathbb{L}_1$) distances of every point in both sets $\mathcal{G}$ and $\mathcal{T}$ to their respective closest point in the other set, and averages them by the cardinality of their respective sets and normalizes by the maximum dimensions of the image. In table 1, we have multiplied the value of μ by 100 to present a percentage mean absolute error.

A human expert is assigned the task to determine whether there is any significant over- or under-estimation of traced neuronal branches. The criterion for testing the significance is set arbitrarily and from experience. We consider a branch to be underestimated if the missing length of the branch is more than 10% of that particular branch when compared to the corresponding branch in the ground truth.

Similarly, we consider a branch to be over-estimated if that branch is totally absent in the corresponding ground-truth and the total length of the branch is more than 10% of the longest branch in that neuron. The assignment of the

correspondence between similar branches in the manually segmented and Tree2Tree traced images are currently done by an expert observer, as there is no satisfactory automated way to do it.

Table 1 gives a overview of the neuronal traces obtained by applying Tree2Tree to our database of sixteen 3D neuronal images from the Drosophila VNC. Note that μ is expressed as a percentage value in Table I. From Table I we note that Tree2Tree segments within a normalized mean absolute error of 2.75% of the ground truth for the images, despite most of the images being affected by lack of contrast, background clutter, amorphous and irregular stem shapes, fragmentation and complex branching.

The last column in table 1 gives the corresponding percentage mean absolute error by NeuronStudio when applied to the same dataset. Reiterating from the discussion in section III-*A*, in each of the sixteen neurons the manual starting seed point has been placed at ten different positions along the neuron before applying NeuronStudio and the longest trace that visually compares best with the ground truth has been retained. It can be seen that except for the first neuron where NeuronStudio outperforms Tree2Tree (this case has been shown in Fig. 15), Tree2Tree consistently outperforms NeuronStudio in all the other cases. We believe that Tree2Tree is especially well-suited for building a complete neuron trace for heavily fragmented and cluttered images with highly irregular stem geometry and complex branching, and seed following algorithms like NeuronStudio are not ideally designed for these scenarios.

IV. STRENGTHS AND LIMITATIONS

In this paper, we have presented an automatic 3D neuron segmentation and morphology generation algorithm that can satisfactorily trace irregularly shaped neurons in low contrast confocal microscopy images. We summarize the strengths and limitations of our algorithm below:

Strengths

(1) In contrast to seed following algorithms, Tree2Tree can track neurons with highly fragmented branches with irregularly shaped neuronal stems.

(2) Tree2Tree can perform in low contrast images with heavy background clutter. This is quite difficult with current directional filtering algorithms, because they depend on relatively clutter free background and high contrast between foreground and background for computing reliable directional gradients.

(3) Tree2Tree handles complicated neuronal branches with ease because the branching is intrinsic and natural to the graph theoretic definition.

Limitations

(1) Tree2Tree currently cannot distinguish clutter that lies close to the actual neuronal branches and are suitably oriented.

(2) Tree2Tree currently does not exploit the radial thicknesses of neurites in computing connected component weights. We estimate that it would be conceptually straightforward to incorporate algorithmic measurement of radial thickness in a future version of Tree2Tree.

Our current work is focused on the development of a morphological matching function that utilizes the segmentation and tracing provided by Tree2Tree [34].

REFERENCES

[1] C. Koch and I. Segev, "The role of single neurons in information processing", *Nat. Neurosci. 3 (Suppl)*, pp. 1171-1177, 2000.

[2] G. A. Ascoli, D. E. Donohue and M Halavi, "NeuroMorpho.Org: a central resource for neuronal morphologies", *Journal of Neuroscience*, vol. 27, pp. 9247-51, 2007.

[3] G. A. Ascoli, "Neuroinformatics grand challenges", *Neuroinformatics*, vol. 6, pp. 1-3, 2008.

[4] J. G. White, E. Southgate, J. N. Thomson and S. Brenner, "The structure of the nervous system of the Nematode Caenorhabditis Elegans," *Philosophical Transactions of the Royal Society of London, Series B, Biological Sciences*, vol. 314, pp. 1-340, 1986.

[5] B. D. Pfeiffer *et al*., "Tools for neuroanatomy and neurogenetics in Drosophila," *Proc. Natl Acad Sci USA*, vol. 105(28), pp. 9715-9720, 2008.

[6] A. Schmid, A. Chiba and C.Q. Doe, "Clonal analysis of Drosophila embryonic neuroblasts: neural cell types, axon projections and muscle targets," *Development*, vol. 126, pp. 4653-89, 1999.

[7] J. Lu, J. C. Fiala and J. W. Lichtman, "Semi-automated reconstruction of neural processes from large numbers of fluorescence images" *PLoS One*, vol. 4 (5), 2009.

[8] K. A. Al-Kofahi, A. Can, S. Lasek, D. H. Szarowski, N. Dowell-Mesfin, W. Shain, J. N. Turner and B. Roysam, "Median based robust algorithms for tracing neurons from noisy confocal microscope images", *IEEE Trans. Information Technology in Biomedicine*, vol. 7, no. 4, pp 302-16, 2003.

[9] K.A. Al-Kofahi, S. Lasek, D.H. Szarowski, C.J. Pace, G. Nagy, J. N. Turner and B. Roysam, "Rapid automated three-dimensional tracing of neurons from confocal image stacks", *IEEE Trans. Information Technology in Biomedicine* , vol. 6 (2), pp. 171–187, 2002.

[10] G. J. Streekstra and J. van Pelt, "Analysis of tubular structures in three-dimensional confocal images", *Network*, vol. 13 (3), pp. 381–395, 2002.

[11] A. Wolf, A. Herzog, S. Westerholz, B. Michaelis and T. Voigt, "Improving fuzzy-based axon segmentation with genetic algorithms", in *Proc. IEEE Congress on Evolutionary Computation*, pp. 1025-31, 2009.

[12] H. Cai, X. Xu, J. Lu, J. Lichtman, S. P. Yung and S. T. C. Wong, "Shape-constrained repulsive snake method to segment and track neurons in 3D microscopy images," in *Proc. 3rd IEEE Intl. Symposium on Biomedical Imaging*, pp. 538-541, 2006.

[13] T. F. Chan and L. A. Vese, "Active Contours without Edges", *IEEE Trans. on Image Processing*, vol. 10(2), pp. 266-277, 2001.

[14] J. F. Evers, S. Schmitt, M. Sibila and C. Duch, "Progress in functional neuroanatomy: precise automatic geometric reconstruction of neuronal morphology from confocal image stacks", *J. Neurophysiol*, vol. 93 (4), pp. 2331–2342, 2005.

[15] H. Cuntz, F. Forstner, J. Haag and A. Borst, "The morphological identity of insect dendrites", *PLoS Comput. Biol.* vol. 4 (12), 2008.

[16] A. Dima, M. Scholz and K. Obermayer, , "Automatic segmentation and skeletonization of neurons from confocal microscopy images based on the 3-D wavelet transform," *IEEE Trans. on Image Processing*, vol.11(7), pp. 790- 801, 2002.

[17] C. M. Weaver, P. R. Hof, S. L. Wearne and W. B. Lindquist, "Automated algorithms for multiscale morphometry of neuronal dendrites", *Neural Computation*, vol. 16(7), pp. 1353-1383, 2004.

[18] M. L Narro, F. Yang, R. Kraft, C. Wenk, A. Efrat and L. L. Restifo, "NeuronMetrics: software for semi-automated processing of cultured neuron images", *Brain Res*. Vol. 1138, pp. 57–75, 2007.

[19] M. Oberlaender, R. M. Bruno, B. Sakmann and P.J. Broser, "Transmitted light brightfield mosaic microscopy for three-dimensional tracing of single neuron morphology", *J. Biomed. Opt*, vol. 12 (6), 2007.

[20] M. Pool, J. Thiemann, A. Bar-Or and A.E. Fournier, "NeuriteTracer: a novel ImageJ plug-in for automated quantification of neurite outgrowth", *J. Neurosci. Methods*, vol. 168 (1), pp. 134–139, 2008.

[21] J. Selinummi, P. Ruusuvuori, A. Lehmussola, H. Huttunen, O.Yli-Harja and R. Miettinen, "Three-dimensional digital image analysis of immunostained neurons in thick tissue sections", in *Proc. Conf. IEEE Eng. Med. Biol. Soc.*, vol. 1, pp. 4783–4786, 2006.

[22] J. Chen and B. G. Condron, "Branch architecture of the fly larval abdominal serotonergic neurons", *Developmental Biology*, vol. 320(1), pp. 30-38, 2008.

[23] S. Basu, A. Aksel, B. Condron and S.T. Acton, "Tree2Tree: neuron segmentation for generation of neuronal morphology," *Proc. IEEE Int. Symp. of Biomed. Imag*., pp. 548-551, 2010.

[24] N. Otsu, "A threshold selection method from gray-level histograms", *IEEE Trans. Systems, Man and Cybernetics*, vol. 9(1), pp. 62-66, 1979.

[25] A. F. Frangi, W. J. Niessen, K. L. Vincken and M. A. Viergever, "Muliscale vessel enhancement filtering", in *Proc. First Int. Conf. on Medical Image Computing and Computer-Assisted Intervention* (MICCAI '98), pp. 130-137, 1998.

[26] K. Palágyi, E. Sorantin, E. Balogh, A. Kuba, C. Halmai, B. Erdôhelyi and K. Hausegger, "A sequential 3D thinning algorithm and its medical applications", in *Proc. 17th Int. Conf. Information Processing in Medical Imaging, published in Lecture Notes in Computer Science*, vol. 2082, pp. 409-415, 2001.

[27] W. Xie, R. P. Thompson and R. Perucchio, "A topology-preserving parallel 3D thinning algorithm for extracting the curve skeleton", *Pattern Recognition*, vol. 36, pp. 1529-1544, 2003.

[28] S. Lobregt, P. W. Verbeek and F. C. A. Groen, "Three-dimensional skeletonization: principle and algorithm," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 2, pp. 75 - 77, 1980.

[29] R. Diestel, *Graph theory* (*Graduate Texts in Mathematics*), Springer, 3rd edition, 2006.

[30] E. Meijering, "Neuron tracing in perspective," *Cytometry Part A*, vol. 77(7), pp. 693–704, 2010.

[31] S. L. Wearne, A. Rodriguez, D. B. Ehlenberger, A. B. Rocher, S. C. Hendersion, and P. R. Hof, "New techniques for imaging, digitization and analysis of three-dimensional neural morphology on multiple scales," *Neuroscience*, vol. 136, pp. 661-680, 2005.

[32] M. Longair, *Simple Neurite Tracer*, http://pacific.mpi-cbg.de/wiki/index.php/Simple_Neurite_Tracer.

[33] H. Peng, Z. Ruan, F. Long, J. H. Simpson and E. Myers, "V3D enables real-time 3D visualization and quantitative analysis of large-scale biological image data sets," *Nature Biotechnology*, vol. 28, pp. 348-353, 2010.

[34] S. Basu, B. Condron and S.T. Acton, "Path2Path: hierarchical path-based analysis for neuron matching," to appear in *Proc. IEEE Int. Symp. of Biomed. Imag*., Chicago, the United States, March 30-April 2, 2011.