Journal of Neuroscience Methods 246 (2015) 13–21

Contents lists available at ScienceDirect

Journal of Neuroscience Methods

journal homepage: www.elsevier.com/locate/jneumeth

Computational Neuroscience

Efficient semi-automatic 3D segmentation for neuron tracing in electron microscopy images

Cory Jones[a,b], Ting Liu[a,c], Nathaniel Wood Cohan[d], Mark Ellisman[d], Tolga Tasdizen[a,b,c,*]

[a] Scientific Computing and Imaging Institute, University of Utah, United States
[b] Department of Electrical and Computer Engineering, University of Utah, United States
[c] School of Computing, University of Utah, United States
[d] National Center for Microscopy and Imaging Research, University of California, San Diego, United States

HIGHLIGHTS

- Proposed a semi-automatic approach to proofreading segmentations of EM images.
- Achieved accuracy by experts approaching manual labeling quality.
- Showed a significant time reduction for dense segmentation of EM images by experts.

ARTICLE INFO

Article history:
Received 27 December 2014
Received in revised form 27 February 2015
Accepted 3 March 2015
Available online 10 March 2015

Keywords:
Semi-automatic segmentation
Image segmentation
Electron microscopy
Neuron reconstruction
3D segmentation
Connectomics

ABSTRACT

Background: In the area of connectomics, there is a significant gap between the time required for data acquisition and dense reconstruction of the neural processes contained in the same dataset. Automatic methods are able to eliminate this timing gap, but the state-of-the-art accuracy so far is insufficient for use without user corrections. If completed naively, this process of correction can be tedious and time consuming.
New method: We present a new semi-automatic method that can be used to perform 3D segmentation of neurites in EM image stacks. It utilizes an automatic method that creates a hierarchical structure for recommended merges of superpixels. The user is then guided through each predicted region to quickly identify errors and establish correct links.
Results: We tested our method on three datasets with both novice and expert users. Accuracy and timing were compared with published automatic, semi-automatic, and manual results.
Comparison with existing methods: Post-automatic correction methods have also been used in Mishchenko et al. (2010) and Haehn et al. (2014). These methods do not provide navigation or suggestions in the manner we present. Other semi-automatic methods require user input prior to the automatic segmentation such as Jeong et al. (2009) and Cardona et al. (2010) and are inherently different than our method.
Conclusion: Using this method on the three datasets, novice users achieved accuracy exceeding state-of-the-art automatic results, and expert users achieved accuracy on par with full manual labeling but with a 70% time improvement when compared with other examples in publication.

1. Introduction

1.1. Motivation

In the field of neuroscience, there is a long-standing interest in mapping the neural pathways that combine to create the networks that control the functions of most animals. One of the more prominent examples of studying a complete neural network is reconstruction of the *Caenorhabditis elegans* nematode (White et al., 1986; Varshney et al., 2011). With advances in technology, research has begun trying to extend this type of work to portions of more complex organisms such as the *Drosophila* (Cardona et al., 2010) and the mouse neuropil (Deerinck et al., 2010). In addition, a multi-institutional collaborative website funded by the NIH has recently been set up to facilitate mapping the human connectome (Toga et al., 2014).

Early work on neuronal network mapping (White et al., 1986) used electron micrographs as the imaging modality, while current work (Cardona et al., 2010; Deerinck et al., 2010; Toga et al., 2014)

* Corresponding author at: Scientific Computing and Imaging Institute, University of Utah, Salt Lake City, UT, United States. Tel.: +1 801 581 3539; fax: +1 801 585 6513.
E-mail address: tolga@sci.utah.edu (T. Tasdizen).

http://dx.doi.org/10.1016/j.jneumeth.2015.03.005
0165-0270/

generally uses some form of digital electron microscopy (Briggman and Bock, 2012). The datasets that we use in this paper were created using serial section transmission electron microscopy (ssTEM), serial block-face scanning electron microscopy (SBFSEM), and serial section scanning electron microscopy (ssSEM) with in-plane resolutions of 3–6 nm and section thicknesses of 30–50 nm. An 8-bit grayscale image stack of just 1 mm ×1 mm ×1 mm with a resolution of 6 nm × 6 nm × 50 nm requires over 500 terabytes of space to store. Complete manual labeling as was done for the original *C. elegans* (White et al., 1986) is impractical for a dataset this large. The anisotropy of the data, however, creates difficulty in developing fully automatic 3D approaches with sufficient accuracy. This difficulty can be seen in the results from the 2013 3D segmentation of Neurites in EM Images Challenge (Arganda-Carreras et al., 2013). In this paper, we introduce a method that utilizes the information contained in the automatic segmentation results to allow the user to quickly label a dataset of interest.

1.2. Related work

For our method, we require the user to verify both the 2D segmentation and 3D linking that are suggested by automated processing. In Chklovskii et al. (2010), several semi-automatic methods for both 2D segmentation and 3D linking are reviewed. Semi-automatic methods can be separated into two distinct classes: (1) pre-automatic user input methods (pre-auto) and (2) post-automatic user input methods (post-auto). The pre-auto methods require the user to give input prior to an automatic method taking over the segmentation. Some examples of these include (Jeong et al., 2009), which uses manual input with a level-set method, and (Cardona et al., 2010), which uses skeleton tracing. These methods do not use the automatic method to assist the user and are very different from the method we present here.

Post-auto methods are sometimes called proofreading methods as in Mishchenko et al. (2010) and Haehn et al. (2014). In Mishchenko et al. (2010), the authors use proofreading to complete the labeling of their dataset; however, the specific method used is not described. In Haehn et al. (2014), the authors present a method that requires the user to manually search for errors without specific guidance and then provides tools for correcting those errors; this differs from our method which specifically guides the user to review each segmentation. Another post-auto method is Eyewire (Seung, 2013). The method requires users to navigate through the volume and add regions to a selected cell until it is completely labeled within the provided volume. Users self-navigate and are required to move forward and backward regularly through the volume to ensure correctness and completeness. Our method, on the other hand, allows the user to navigate if needed, but provides a controlled navigation between cells automatically. In addition, whereas Eyewire focuses on labeling only one cell at a time, we proceed one 2D section at a time, i.e., we have the user completely label one section before moving onto the next sections.

In the following sections, we will describe the specific semi-automatic method that we use to completely label a dataset volume along with results and conclusions. More specifically, in Section 2 we describe both the 2D semi-automatic and 3D linking methods along with a description of timing considerations for those methods. In Section 3 we present our segmentation results for several datasets and users. Finally, in Section 4 we provide the conclusions we have been able to draw from these results.

2. Method

Consider an image volume $\mathcal{V}$ consisting of m image slices $\mathcal{I}_i$ that is to be segmented into a set of n true regions, t_l, such that the true segmentation is $\mathcal{T} = \{t_1, t_2, \ldots, t_n\}$. Each t_l in $\mathcal{T}$ has a unique integer label and consists of a set of pixels $v_{i,j,k} \subset \mathcal{V}$ that are 26-connected. We will produce a set of q predicted regions, p_k^F, such that the final predicted segmentation is $\mathcal{P}^F = \{p_1^F, p_2^F, \ldots, p_q^F\}$. Throughout the remainder of this paper, primary subscripts are used to indicate indices and superscripts and secondary subscripts are used to distinguish a label. Superscript T indicates a true version of the corresponding label, superscript F indicates a final prediction of the corresponding label, and superscript I indicates an intermediate prediction of the corresponding label.

2.1. Automatic segmentation

The semi-automatic segmentation method described in Section 2.2 depends on an automatic method that segments the image into r highly oversegmented superpixels, o_j. In an ideal segmentation, each region t_l is comprised of a set of these superpixels such that

$$t_l = \bigcup_{j \in \gamma_l^T} o_j \tag{1}$$

for each t_l in $\mathcal{T}$ where γ_l^T is the set of superpixel indices included in region l. Each o_j will be used exactly once in $\mathcal{T}$. Moving from the ideal to the predictive scenario

$$p_m^I = \bigcup_{j \in \gamma_m^{p_I}} o_j \tag{2}$$

for each intermediate region, p_m^I, in the intermediate segmentation $\mathcal{P}^I$ and each o_j are used exactly once. In addition, the automatic segmentation must also build a hierarchy of merged o_j for the semi-automatic segmentation to work efficiently. Fig. 1 shows each piece of the hierarchy where Fig. 1(a) represents $\mathcal{I}_i$ with superpixels, o_j, labeled with numbers; Fig. 1(b) shows a segmentation with each t_l highlighted; and Fig. 1(c) shows an example tree structure for building $\mathcal{T}$. Presently we use the modular hierarchical approach introduced in Liu et al. (2012) with some 2D refinements described in Liu et al. (2014).

The modular hierarchical approach uses a 2D classification to generate its initial oversegmented superpixels o_j by applying 2D watershed (Beare and Lehmann, 2006) from the ITK library (Yoo et al., 2002) to the results of a 2D cell membrane detection method such as the cascaded hierarchical model (Seyedhosseini et al., 2013) or deep neural networks (Ciresan et al., 2012). From this initial segmentation, the water level is gradually raised to merge neighboring superpixels together. Each merge represents a new node in an unbalanced binary tree consisting of the two merged nodes as children. This merging continues until all nodes have been merged into one large tree with the node from the final merge as the root and each o_j as the leaf nodes. Fig. 1 shows a toy example of this merging process from (Liu et al., 2012).

Using this tree, a set of features to be used for classification purposes is generated for every merge. These features include both geometry based features such as region area and boundary curvatures and intensity based features such as intensity histograms and texton histograms. The merges are then classified using a random forest classifier to assign the probability that a given merge is a true merge in the truth. For example, in Fig. 1(c), the merge of superpixels o_5 and o_6 to form node 8 should have a high probability and the merge of node 9 and node 10 to form node 11 should have a low probability.

Based on the results of this classification, a potential is generated for each node. This potential is computed by multiplying the probability that the current merge should happen with the probability that the merge forming the parent node should not happen. Referring again to Fig. 1(c), the potential for node 8 is computed by multiplying the probability that superpixels o_5 and o_6 merge with

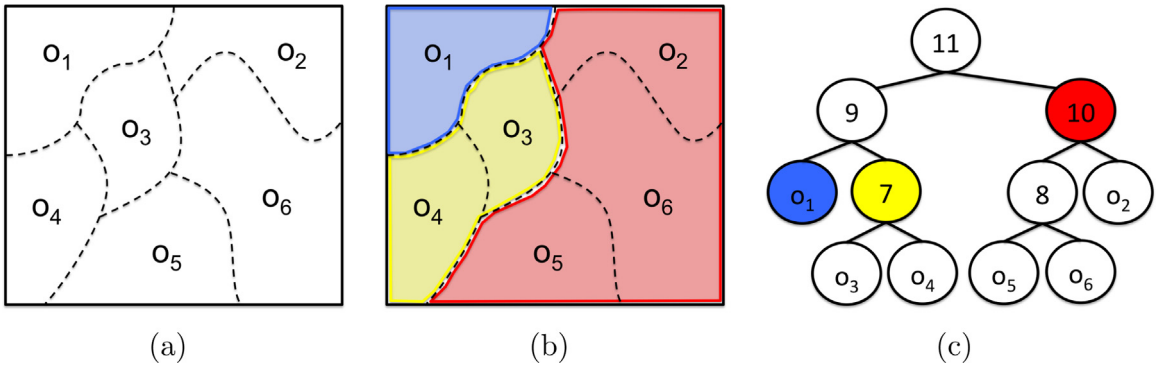

Fig. 1. (a) An example of an image $\mathcal{I}_i$ segmented into superpixels o_j represented by the numbered regions, (b) shows each o_j merged into the predicted segmentation regions p_k^I represented by the colored regions, and (c) shows the corresponding tree structure with labeled nodes. The colored nodes in (c) correspond to the colored regions in (b). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

the probability that node 8 and superpixel o_2 do not merge. This results in the nodes with the highest potential being those with the highest likelihood of being a true segmentation. This potential is what will determine the examination order for the semi-automatic segmentation.

2.2. Semi-automatic segmentation

For the semi-automatic segmentation, we assume that each $\mathcal{I}_i$ is sufficiently oversegmented into r superpixels, o_j, such that each true region, t_l, can be generated from o_j as in Eq. (1). Using this assumption, we seek to implement a method that reorganizes the initial segmentation, $\mathcal{P}^I$, predicted by the automatic method by utilizing the hierarchical structure, each superpixel o_j, and user input to generate the final predicted segmentation, $\mathcal{P}^F$. If the results are ideal then $\mathcal{P}^F = \mathcal{T}$. Due to the 2D nature of the automatic segmentation method used, both proofreading the 2D segmentation and linking the resulting 2D segments using automated suggestions will be necessary. By completing both of these steps simultaneously we seek to reduce the amount of time required from the user.

2.2.1. Implementation

The following method was implemented in C++ using the Visualization Toolkit (Schroeder et al., 2006) for both interaction and visualization. This library was chosen for easy integration with the automatic method, which utilizes the Insight Toolkit (Yoo et al., 2002) extensively. It is currently implemented as a stand-alone command line program with a windowed interface as described below. The program is compiled using CMake Inc. (2015) to allow for easier cross-platform compilation. It has been successfully compiled on both Mac OS X (10.6.8 and later) and Linux operating systems.

2.2.2. Interface

The interface presents the user with four images to assist in completing the semi-automatic segmentation. The first image, appearing in the top right of the interface as shown in Fig. 2, is a portion of the raw EM image for the image slice $\mathcal{I}_i$ being processed and is zoomed in and centered on the current proposed segmentation region, p_m^I. We highlight the border of p_m^I in one color and the interior of p_m^I in a different color, which makes it easy for the user to quickly identify which region is being considered. Additionally, each of the superpixels, o_j, in $\mathcal{I}_i$ are outlined in a third color to show the user all possible segmentations.

The second image displayed on the interface in Fig. 2 appears in the top left corner and is a portion of the raw EM image for the previous image slice, $\mathcal{I}_{i-1}$, with the same zoom and position as $\mathcal{I}_i$. On this image we highlight the border of the proposed link region, p_k^F, in one color and its interior in another color as was done for $\mathcal{I}_i$. As will be described in Section 2.2.3, each region, p_k^F, for the previous slice, $\mathcal{I}_{i-1}$, is completed prior to the final segmentation regions, p_k^F, for the current slice, $\mathcal{I}_i$, making it unnecessary to highlight each superpixel, o_j, on $\mathcal{I}_{i-1}$. Instead we highlight the border of each p_k^F from $\mathcal{I}_{i-1}$. This allows the user to select a different link region if the predicted region is incorrect or to select multiple regions if it is a branch merge point.

The third image displayed appears in the bottom right corner of Fig. 2 and is initially a portion of the raw EM image for the current image slice, $\mathcal{I}_i$, with the zoom and position matching the top left and top right images. This image does not have the proposed segmentation region, p_m^I, highlighted in any way. As the user generates new final segmentations, p_k^F, these segmentations will appear highlighted in a new color to show the user what he or she has already completed. As will be described in Section 2.2.3, the user has the ability to sequence this image forward or backward to see additional slices. When the user looks ahead to slices not yet processed, this image will be only the raw EM image for that slice with no highlighting. When the user looks backward to slices already completed, this image will be the raw EM image for that slice with the segmented region borders highlighted.

Finally, the last image appears in the bottom left corner of Fig. 2 and is initially a portion of the raw EM image for the previous image slice, $\mathcal{I}_{i-1}$, again with the zoom and position matching the other images in the interface. No highlighting of any kind appears on this image. As the user scrolls through the slices forward or backward, this image will display the raw EM image for the slice exactly one previous to the one on display in the third image on the bottom right. An example of this can be seen in the bottom left of Fig. 2. The overall resulting interface display has the images on the left being exactly one slice previous to the images on the right.

2.2.3. Process

The purpose of the semi-automatic method is to take advantage of as much information contained in the automatic segmentation as possible to limit the amount of input required from the user. We also reduce the user input to be single-clicks or single keystrokes to further minimize the amount of time required for a single response. The specific keystrokes used in our implementation can be found in the appendix in Table A.5. This reduction of user input results in the user being unable to split the individual superpixels, o_j, and thus a sufficient oversegmentation is necessary initially.

The process begins with the user being presented with a proposed 2D segmentation and recommended link. For the recommended 2D segmentation, the automatic segmentation node with the highest potential of being a true segmentation as described in Section 2.1 is presented to the user. This recommendation will

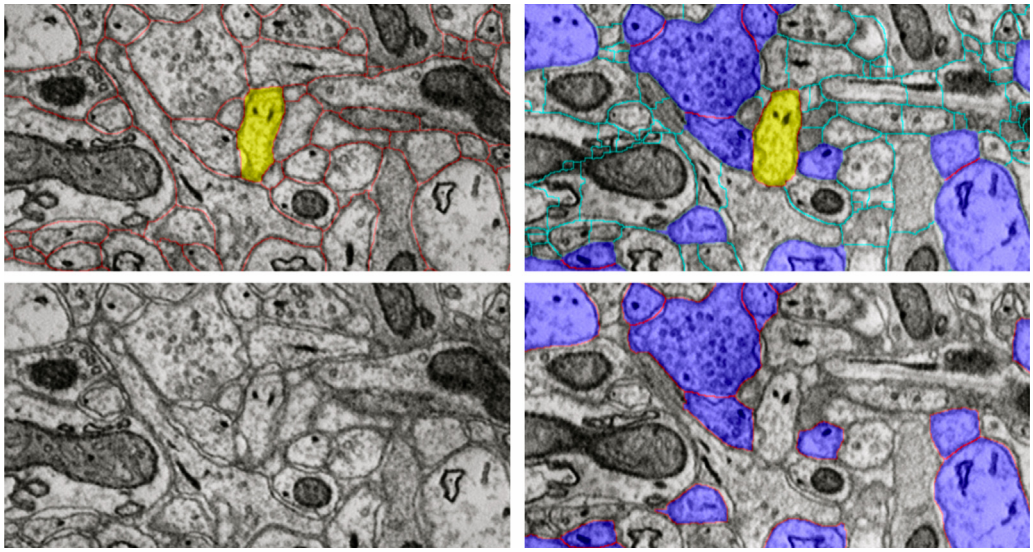

Fig. 2. Screen shot of the interface. The top left shows the image that highlights the proposed link in yellow with all other regions outlined in red. The image on the top right highlights the proposed segmentation in yellow, the o_j in light blue, and the resolved segmentations in dark blue. The image on the bottom left is the raw image of $\mathcal{I}_{i-1}$. The image on the bottom right is the raw image of $\mathcal{I}_i$ with the resolved segmentations highlighted in dark blue. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

result in the user first visiting the regions with the highest likelihood of being in the true segmentation and make it easier for the user to resolve the more difficult regions later on. The suggested 3D link that is simultaneously presented to the user is given as the region, p_k^F, in the previous image slice, $\mathcal{I}_{i-1}$, that shares the most overlap with the current proposed region, p_m^I.

The user will first consider the accuracy of the 2D segmentation. The four possible scenarios for the accuracy of the segmentation as seen in Fig. 3 are correct segmentation, oversegmentation, undersegmentation, and bad segmentation. To assist the user in difficult segmentations, we have included the ability to look to previously segmented images as well as the raw images of the next slices. By looking at previously segmented images, the user can see the general shift of the cell from slice to slice and also see the shape of a previous segmentation that may have provided better contrast, both of which can make the current decision easier. Looking to the unprocessed raw images can provide similar assistance but without a resolved segmentation to use as a baseline. We have also included the ability to zoom out and navigate to other areas of the image for correcting segmentations that may exceed the zoom window. The accuracy of the current proposed segmentation will affect how the user responds to the 3D linking. Those scenarios are described below.

In the case of an accurate 2D segmentation where $p_m^I \approx t_l$, the user will ensure the correct 3D link and select the proper keystroke for a good segmentation. If the suggested 3D link is not correct, the user is able to add and remove individual regions, p_k^F, from the previous slice until the correct regions are linked. The proposed region p_m^I and all the linked regions are assigned the same label. If there are no linked regions for the proposed region, p_m^I, it is assigned a new label. In the tree structure, the node for p_m^I and all of its descendants and direct ancestors are removed from the tree and the region corresponding to the node with the next highest potential is presented to the user.

Oversegmentation refers to a scenario where the proposed segmentation, p_m^I, is such that

$$p_m^I = \bigcup_{j \in \gamma_{ml}^{P_l}} o_j \subset t_l \tag{3}$$

where t_l is the corresponding true segmentation region and $\gamma_{ml}^{P_l}$ is the set of indices for the superpixels to be included in the region. This scenario is handled with the user clicking additional superpixels, o_j, until $p_m^I \approx t_l$. The corresponding 3D link, as in the case of a good segmentation, is verified by the user and then the user indicates a good segmentation. In the tree, clicking regions will result in leaf nodes being removed and the tree being restructured. The restructuring happens with the parent node of each removed leaf node being replaced with the sibling node of that corresponding leaf node. The potentials and all other node information remain the same. In addition, the original recommended node, p_m^I and all of its descendants and direct ancestors are removed from the tree and the region corresponding to the node with the next highest potential is once again presented to the user. Fig. 4(b) shows a toy example of this oversegmentation process. In the first column is a segmentation where regions o_4 and o_3 make up the oversegmentation proposed by the automatic method and o_2 is clicked by the user, in the second column is the labeled result, in the third column is the tree structure for the first column, and in the fourth column is the tree structure remaining after the result.

Undersegmentation results when the proposed segmentation, p_m^I, is such that

$$p_m^I = \bigcup_{j \in \gamma_{ml}^{P_l}} o_j \supset t_l \tag{4}$$

where once again t_l is the corresponding true segmentation region and $\gamma_{ml}^{P_l}$ is the set of indices for the superpixels to be included in the region. In this scenario, the user will indicate an undersegmentation and the current node and all its ancestors are removed from the tree and the next region is presented. To simplify the visual processing for the user, we present the child node of the removed node that has the higher potential as the next proposed region. Because the user is already focused on resolving this region, the user is able to more quickly process what the correct response should be. This process continues until a correct segmentation or an oversegmentation is found in which case the procedure follows as described previously. 3D linking can be ignored for undersegmentation until a correct segmentation or oversegmentation result because no region labels are assigned. Fig. 4(c) shows a toy example of this undersegmentation process. In the first column is a proposed segmentation where

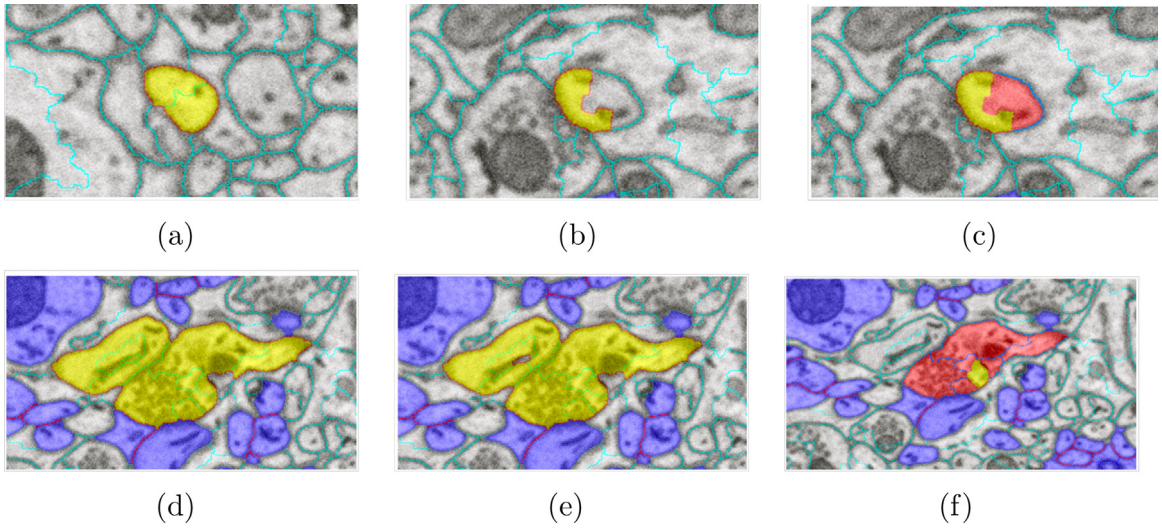

Fig. 3. (a) An example of a correct segmentation, (b) is an example of an oversegmentation with a portion of just one region suggested, (c) is the true segmentation for (b), (d) is an example of an undersegmentation with all of one region and a portion of another region suggested, (e) is an example of a bad segmentation with portions of multiple regions but no entire region suggested, and (f) is the true segmentation for both (d) and (e). (For color highlighted regions, the reader is referred to the web version of this article.)

regions o_4, o_3 and o_1 make up the undersegmentation proposed by the automatic method, in the second column is the new proposed segmentation after undersegmentation is indicated by the user, in the third column is the tree structure for the first column, and in the fourth column is the tree structure after the result.

Bad segmentation is when the segmentation is not correct and both Eqs. (3) and (4) fail to be satisfied. As seen in 3(e), it is a segmentation where portions of multiple regions are included, but no complete region is included. Although different from undersegmentation in that a correct segmentation cannot be obtained, we proceed in the same fashion as the undersegmentation with the current node and all its ancestors being removed from the tree and the child node with the higher potential being presented as the next proposed region. This process continues until an oversegmentation is found and the user is able to resolve as described above. Fig. 4(d) shows a toy example of this bad segmentation process. In the first column is a proposed segmentation where regions o_3, o_6, and o_2 make up the bad segmentation proposed by the automatic method, in the second column is the new proposed segmentation after bad segmentation is indicated by the user and o_5 is manually clicked, in the third column is the tree structure for the first column, and in the fourth column is the tree structure after bad segmentation is indicated by the user and o_5 is manually clicked.

The goal with each of these steps is to be as efficient as possible. When both the segmentation and 3D linking are clearly correct, a user typically requires approximately one second to assess the accuracy and respond. When the segmentation is correct, but the 3D linking is inaccurate, the time to complete is limited by how quickly the user is able to click the correct link regions. Because fixing the segmentation is often done in just one or two clicks, the time required is also minimal. Correcting the 2D segmentation, on the other hand, may require more time to complete depending on how close to accurate the segmentation was. For oversegmentation, the number of o_j that need to be added may be significant if the associated true region, t_l, is large and the time required to complete this correction may be tens of seconds. In the case of undersegmentation and bad segmentation, typically the number of responses to get to either a correct segmentation or an oversegmentation is small, and so the time required for these results is nearly the same as the time required for oversegmentation. Finally the last real limitation in time is related to the quality of the image set and the ease of determining an accurate segmentation.

3. Results

To test the effectiveness of our method, we applied it to three datasets with fully labeled 3D ground truths. For each dataset used, we split the data into training and testing for the automatic method and then applied the semi-automatic method to only the testing set. The justification for this is that in a live application the training set needed for the automatic method will be assumed to have full labels and not require any manual labeling. The accuracy of each test is measured using the adapted Rand error, which is an *F-score* error computed from the pairwise precision and pairwise recall scores as described in Liu et al. (2014) and also used in the International Symposium on Biomedical Imaging as the grading metric for the 2012 Segmentation of Neuronal Structures in EM Stacks Challenge (Arganda-Carreras et al., 2012) and the 2013 3D segmentation of Neurites in EM Images Challenge (Arganda-Carreras et al., 2013). This *F-score* error metric provides a robust 3D segmentation metric that emphasizes both topological accuracy of regions and geometric accuracy of membrane locations but with minimal dependence on accurate membrane thicknesses.

The tests were carried out on two different machines. For the Mouse Neuropil dataset, the tests were performed on a machine with 32 Intel Xeon CPU E5-2670 processors at 2.60 GHz with 126 GB of RAM running CentOS 6.4. The other datasets were completed on a machine with 32 Intel Xeon x7350 processors at 2.93 GHz with 196 GB of RAM running SUSE Linux Enterprise Server 11 (x86_64). Additionally, all of the datasets have been used on a machine with 2 6-Core Intel Xeon Processors at 2.66 GHZ with 32 GB of RAM running OS X 10.6.8, although this machine was not used for any of the complete testing results reported here. On each machine, there was no noticeable delay when moving from slice to slice as long as there was sufficient available RAM for the loaded dataset. For the largest of these datasets, the amount of free RAM required by the interface when the entire dataset was loaded was a little over 20 GB.

3.1. Drosophila *VNC*

As used in the ISBI 2012 segmentation challenge (Arganda-Carreras et al., 2012), this dataset is described as consisting of two stacks of 30 sections from a ssTEM dataset of the *Drosophila* first instar larva ventral nerve cord (VNC). The microcube measures approximately $2 \times 2 \times 1.5$ μm with a resolution

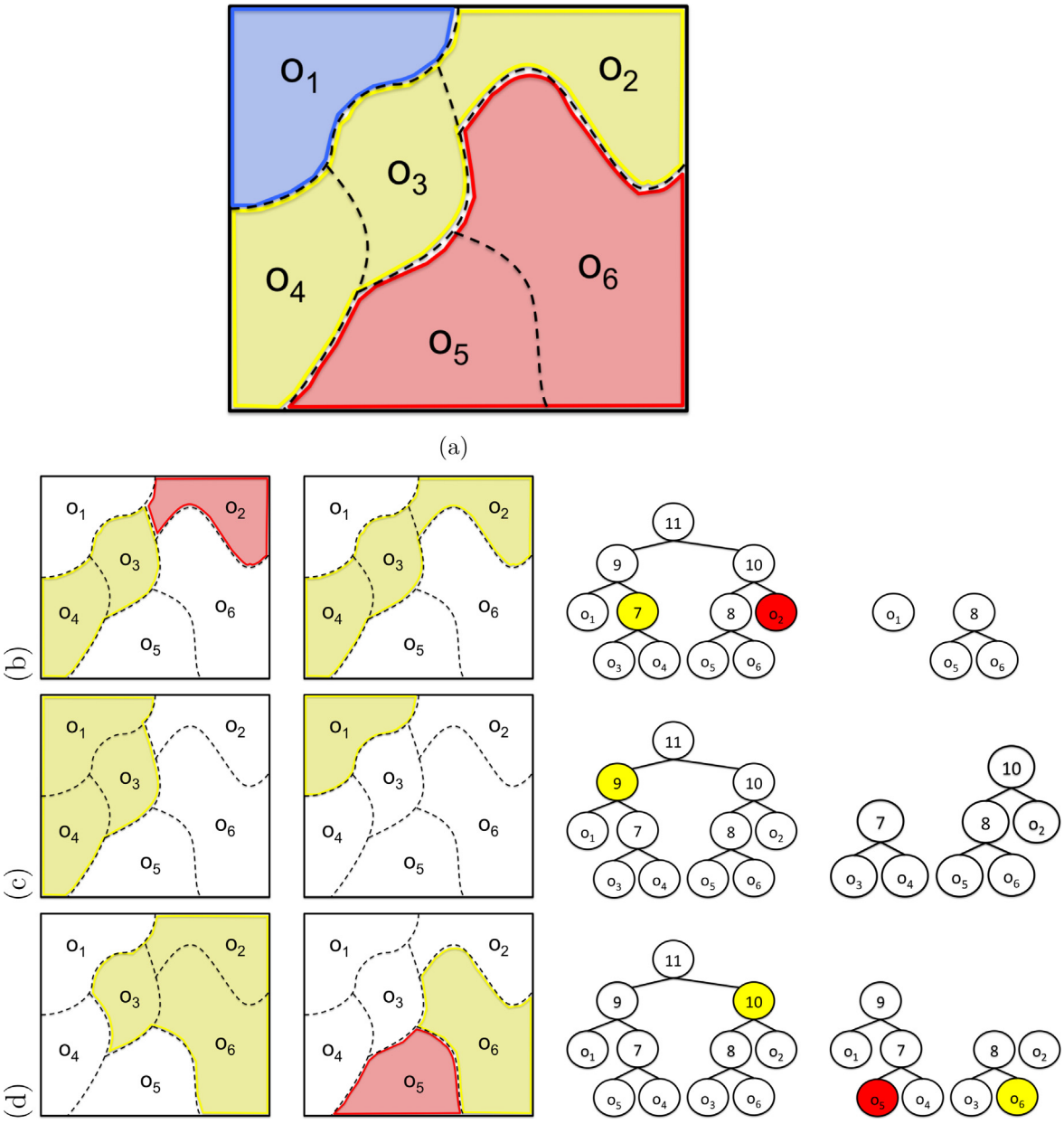

Fig. 4. (a) The truth image label where each color represents a true region for (b)–(d). In (b)–(d) the first column shows the initial proposed segmentation in yellow the second column shows the result of resolving the section as described in the method section, the third column is the tree associated with the first column, and the fourth column is the tree associated with the second column. (b) An oversegmentation example where yellow is the suggested and final segmentation and red is the manually clicked region. (c) An undersegmentation example where yellow is the suggested and final segmentation. (d) A bad segmentation example where yellow is the suggested segmentation and red is the manually clicked region. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

of $4 \times 4 \times 50$ nm/voxel, resulting in an image stack of size $512 \times 512 \times 30$ for both the training and testing stacks. The training set with corresponding 2D membrane labels and the testing set were downloaded from the challenge site (Arganda-Carreras et al., 2012).

For this dataset we were able to obtain the 3D labels necessary to compute the accuracy of the tests for only the 30 training images. As a result our tests were carried out using a split of the stack such that the last 20 images in the stack were used as training images and the first 10 images in the stack were used as testing images. The automatic method was trained on the 20 training images and then applied to the 10 testing images. We then used both the semi-automatic method and automatic 3D linking as described in Liu et al. (2014) for comparison. In addition, the semi-automatic method was completed twice by a user who was familiar with EM images and segmentations, but was not an expert in neuroanatomy. In the first semi-automatic test, the user completed the 3D segmentation without any extra assistance, but for the second semi-automatic test the user completed the 3D segmentation using the 3D ground truth as a guide. The purpose of the first test is to show the results that are achieved by a novice user and the second test is to show the best results that could be achieved by an expert neuroanatomist using our method. Table 1 shows the segmentation results and Fig. 5 gives a 3D view of a few neurites from the novice segmentation of this dataset.

In Table 1 the semi-automatic method with a novice user shows a small decrease in the pair precision but a significant improvement in the pair recall, resulting in a 2.3% improvement in the error value. The slight decrease in the pair precision indicates more assignments of pixels within the same region that should belong

Table 1
The 3D accuracy results on the *Drosophila* VNC dataset for the automatic method without user input, the semi-automatic method with a novice user, and the semi-automatic with a simulated expert user. Note the automatic results differ from the challenge results (Arganda-Carreras et al., 2012) because the challenge was for 2D results and here we report 3D results.

No.	Approach	Testing error	Pair precision	Pair recall
1	Automatic (Liu et al., 2014)	0.131	0.908	0.834
2	Semi-automatic (novice)	0.108	0.896	0.887
3	Semi-automatic (expert)	0.014	0.990	0.982

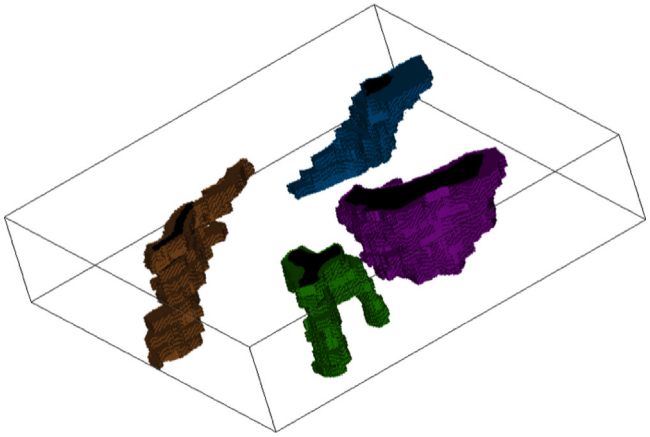

Fig. 5. The above figure is a 3D view of a few neurites selected from the novice segmentation of the *Drosophila* VNC dataset.

in different regions, whereas the improvement in pair recall indicates fewer assignments of pixels to different regions that should belong to the same region. Therefore, we conclude that the effect of the user interaction in this case is to largely correct oversegmentation errors. The expert user is largely able to correct the remaining errors as the results shown in Table 1 indicate a 1.4% total error.

3.2. Mouse cortex

The second dataset comes from the ISBI 2013 3D segmentation challenge (Arganda-Carreras et al., 2013). It consists of two stacks of 100 images to be used as training and testing sets. Both stacks come from a mouse cortex and are acquired using ssSEM, respectively. The microcube is approximately $6 \times 6 \times 3$ μm at a resolution of $6 \times 6 \times 30$ nm/voxel, resulting in an image stack of size $1024 \times 1024 \times 100$ for both the training and testing stacks. These stacks were downloaded from the challenge site (Arganda-Carreras et al., 2013).

For this dataset we once again applied the semi-automatic method to a portion of the training stack not used in training the automatic method and completed it with both a novice user and a simulated expert user as described for the *Drosophila* VNC dataset. We split the training set so that the first 50 images of the stack were used for training and the last 50 images of the stack were used for testing. Table 2 shows the results of the novice and

Table 2
The 3D accuracy results on the mouse cortex dataset for the automatic method without user input, the semi-automatic method with a novice user, and the semi-automatic with a simulated expert user. Note the automatic results are worse than the challenge results (Arganda-Carreras et al., 2013) because the difficulty is greater for the training set than for the testing set.

No.	Approach	Testing error	Pair precision	Pair recall
1	Automatic (Liu et al., 2014)	0.239	0.922	0.647
2	Semi-automatic (novice)	0.131	0.913	0.897
3	Semi-automatic (expert)	0.051	0.980	0.920

Table 3
The 3D accuracy results as reported on the challenge site (Arganda-Carreras et al., 2013) with the semi-automatic results inserted into the rankings.

No.	Group	Testing error
1	Human	0.060
2	Our approach (novice)	0.081
3	Team Gala	0.100
4	Automatic approach (Liu et al., 2014)	0.124
5	FlyEM (Nunez-Iglesias et al., 2013)	0.125
6	rll	0.131
7	Rhoana (Kaynig et al., 2013)	0.148
8	shahab	0.167

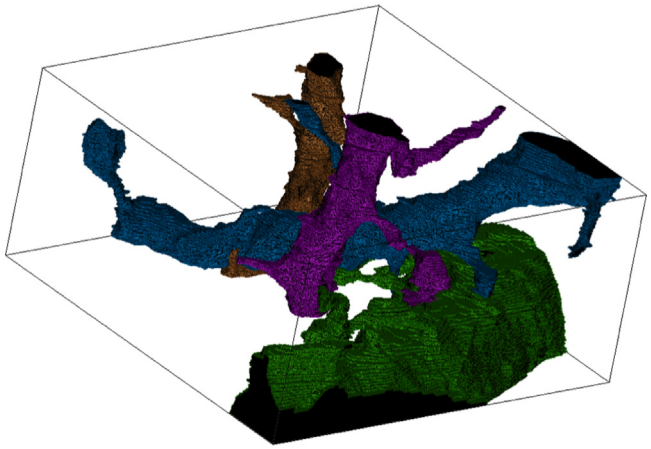

Fig. 6. A 3D view of a few neurites selected from the segmentation of the mouse cortex testing stack.

expert users compared to the automatic method for this test. These results are consistent with the *Drosophila* experiment where the novice user showed a slight drop in the precision and a modest improvement in the recall and the expert user showed a significant improvement in both precision and recall. In addition, we applied the semi-automatic method to the testing stack with a novice user for submission to the challenge to compare against the state-of-the-art methods. A domain expert was not available for this dataset. Table 3 shows the results for this semi-automatic method along with other top results listed on the challenge site. For the challenge results, the pair precision and pair recall are not available. Fig. 7 shows a 3D view of a few neurites obtained from segmenting the testing stack of this dataset.

In Table 2 our approach with a novice user on the split training set was able to do considerably better than the automatic method, with an even more significant improvement in accuracy achieved by an expert user. For the challenge submission results shown in Table 3 our approach with a novice user is able to achieve accuracy exceeding the current state of the art automatic method by nearly 2%. We anticipate that having an expert user complete the full challenge testing set would result in an improvement such that our error would rival the error in two human experts manually labeling the same dataset but with our method requiring significantly less effort (Fig. 6).

3.3. Mouse neuropil

This dataset was acquired at the National Center for Microscopy and Imaging Research at the University of California, San Diego using SBFSEM. The complete dataset consists of a stack of 400 images each of size 4096×4096 at a resolution of $10 \times 10 \times 50$ nm/voxel (Deerinck et al., 2010). For evaluation purposes, we use a subset of this dataset that has been manually labeled

Table 4
The 3D accuracy results on the mouse neuropil dataset for the automatic method without user input and the semi-automatic method with an expert from the National Center for Microscopy and Imaging Research.

No.	Approach	Testing error	Pair precision	Pair recall
1	Automatic (Liu et al., 2014)	0.366	0.867	0.499
2	Semi-automatic (expert)	0.106	0.892	0.896

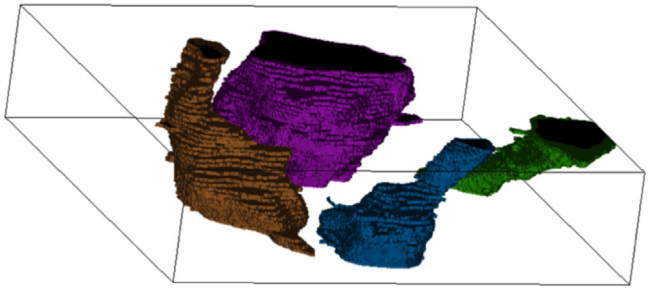

Fig. 7. A 3D view of a few neurites selected from the segmentation by an expert of the mouse neuropil dataset.

in 2D by expert neuroanatomists. This subset is $7 \times 7 \times 3.5\,\mu m$, resulting in 70 images with 700×700 voxels/image.

In the following results, the automatic method was trained for 2D using the method described in Liu et al. (2012) with a representative sample of 14 slices from throughout the dataset. Seven of the slices used in training do come from among the 35 slices used in the final 3D testing set because the automatic 2D segmentation performed poorly when trained on consecutive slices. The poor performance in 2D segmentation was due to the dataset having an uneven distribution of large structures. The 3D automatic linking used the last 30 slices in the dataset as training and used the method described in Liu et al. (2014). Finally the testing for both the automatic method and the semi-automatic method used the first 35 slices of the dataset. There were five slices in the middle that were discarded due to errors in the 2D ground truth labeling. Table 4 shows the final segmentation results. In spite of the significant error found in the automatic results, an expert [1] using this proofreading method was able to largely correct these errors and achieve a more acceptable result. Fig. 7 provides a 3D view of a few select neurites from the expert labeling of this dataset.

3.4. Timing analysis

For the mouse cortex dataset in Section 3.2, the novice user was able to complete each slice of the image stack in just under 30 min on average. Extending this time per slice out over the entire dataset, the total time required to fully label 100 slices of this data would be approximately 50 h or equivalently 8 min 15 s to do 3D labeling of 10 slices of a 1 μm^2 section. Using a similar analysis for the *Drosophila* VNC dataset in Section 3.1, the novice user required just over 12 min for each slice of the image stack or equivalently 30 min 16 s to do 3D labeling of 10 slices of a 1 μm^2 section. Finally, for the mouse neuropil dataset in Section 3.3, the expert user required just over 11 min for each slice of the image stack or equivalently 2 min 15 s to do 3D labeling of 10 slices of a 1 μm^2 section. By comparison, the proofreading done in Mishchenko et al. (2010) reports a time of 160 h, including training, for a non-expert user to complete the proofreading of a 167 μm^3 volume or equivalently 25 min 48 s to complete 10 slices of a 1 μm^2 section. In Mishchenko et al. (2010), the authors also cite significant time improvements for completion by an expert to 8 min 2 s for 10 slices of a 1 μm^2 section. These indicate that proofreading by a novice user using this method compares favorably with Mishchenko et al. (2010), and the time required by an expert user improves upon the time required by the expert in Mishchenko et al. (2010) by over 70%.

[1] The expert for this dataset is a researcher at the University of California, San Diego.

4. Conclusion

In this paper, we presented a semi-automatic method that can be used to perform 3D segmentation of neurites in EM image stacks. First, an automatic method creates a hierarchical structure for recommended merges of superpixels. The user then visits each node in this structure from highest accuracy potential to lowest potential until all nodes have been visited or removed. At each node the user interacts with the semi-automatic method via mouse clicks or single keystrokes to indicate the quality of or to correct the 2D segmentation and the 3D linking simultaneously.

When completed by a novice user, we were able to demonstrate significant improvement over using automatic methods. We were also able to demonstrate accuracy that is approximately the same as manual labeling for three different datasets when our method is used by an expert in neuroanatomy. Additionally, we have been able to demonstrate that our timing is better than currently published timing for other proofreading methods with a novice user and comparable to other methods with an expert user. This timing improvement is achieved by utilizing the information contained in the automatic method.

The test datasets presented here are relatively small portions of much larger datasets. The entire Mouse Neuropil dataset for example is $4096 \times 4096 \times 400$. The limiting factor in applying this method to the full dataset directly is memory usage. Completing the entire dataset could be done by breaking it into smaller memory-manageable blocks and then stitching them together. Additionally, as with any method relying on user input, user fatigue may contribute to judgement errors as these blocks get larger. Limiting the length of time a user spends completing the proofreading in a single sitting can help to eliminate these errors.

Our method also relies on the assumption that initial superpixels are sufficiently oversegmented such that they do not need to be split. We have found this to not always be the case. We are researching efficient ways to correct superpixels that are undersegmented. We anticipate that with the addition of this ability, an expert user would be able to create near perfect segmentations of any dataset. Additionally, further research into identifying nodes that could be skipped because the automatic method is correct could result in further time improvements as skipping nodes reduces the amount of interaction required from the user.

Acknowledgments

This work was supported by NIH 1R01NS075314-01 (TT, MHE) and NSF IIS-1149299 (TT). For providing the *Drosophila* VNC dataset we thank the Albert Cardona Lab at the Howard Hughes Medical Institute Janelia Farm Research Campus, for providing the mouse neuropil dataset and an expert user we thank the National Center for Microscopy and Imaging Research at the University of California, San Diego, for providing the mouse cortex dataset we thank the Jeff Lichtman Lab at Harvard University, and for providing the deep neural networks membrane detection probability maps we thank Alessandro Guisti and Dan Ciresan at the Dalle Molle Institute for Artificial Intelligence.

Table A.5
Summary of possible user input.

Input:	Usage
p:	View previous slice
n:	View next slice
g:	Indicate suggested region is acceptable
u:	Indicate suggested region is an undersegmentation or bad segmentation
Left Click:	Recenter view on clicked point
Shift + Left Click:	Add or removed user selected supervoxel to region
+/=:	Increase zoom
_/-:	Decrease zoom
0:	Recenter current region in view areas
1:	Reset to original view for the current region
s:	Save volume and exit
d:	Indicate the user has completed the current slice and is ready to move to the next slice
q	Quit without saving

Appendix A. Summary of user input

Table A.5 summarizes the possible input to the program and the reason for using each input.

References

Arganda-Carreras I, Seung HS, Cardona A, Schindelin J. Segmentation of neuronal structures in EM stacks challenge – ISBI 2012; 2012 http://brainiac2.mit.edu/isbi_challenge/ [accessed 10.03.14].

Arganda-Carreras I, Seung HS, Vishwanathan A, Berger D. 3D segmentation of neurites in EM images challenge – ISBI 2013; 2013 http://brainiac2.mit.edu/SNEMI3D/ [accessed 10.03.14].

Beare R, Lehmann G. The watershed transform in ITK – discussion and new developments. Insight J 2006, http://hdl.handle.net/1926/202

Briggman KL, Bock DD. Volume electron microscopy for neuronal circuit reconstruction. Curr Opin Neurobiol 2012;22(1):154–61, http://dx.doi.org/10.1016/j.conb.2011.10.022.

Cardona A, Saalfeld S, Preibisch S, Schmid B, Cheng A, Pulokas J, et al. An integrated micro- and macroarchitectural analysis of the *Drosophila* brain by computer-assisted serial section electron microscopy. PLoS Biol 2010;8(10):e1000502.

Chklovskii DB, Vitaladevuni S, Scheffer LK. Semi-automated reconstruction of neural circuits using electron microscopy. Curr Opin Neurobiol 2010;20(5):667–75, http://dx.doi.org/10.1016/j.conb.2010.08.002.

Ciresan D, Giusti A, Gambardella LM, Schmidhuber J. Deep neural networks segment neuronal membranes in electron microscopy images. Adv Neural Inf Process Syst 2012;25:2852–60.

K. Inc. Cmake; 2015 http://www.cmake.org [accessed 23.02.15].

Deerinck TJ, Bushong EA, Lev-Ram V, Shu X, Tsien RY, Ellisman MH. Enhancing serial block-face scanning electron microscopy to enable high resolution 3-D nanohistology of cells and tissues. Microsc Microanal 2010;16(S2):1138–9.

Haehn D, Knowles-Barley S, Roberts M, Beyer J, Kasthuri N, Lichtman J, et al. Design and evaluation of interactive proofreading tools for connectomics. IEEE Trans Vis Comput Graph 2014;20(12):2466–75, http://dx.doi.org/10.1109/TVCG.2014.2346371.

Jeong W-K, Beyer J, Hadwiger M, Vazquez A, Pfister H, Whitaker R. Scalable and interactive segmentation and visualization of neural processes in EM datasets. IEEE Trans Vis Comput Graph 2009;15(6):1505–14, http://dx.doi.org/10.1109/TVCG.2009.178.

Kaynig V, Vazquez-Reina A, Knowles-Barley S, Roberts M, Jones TR, Kasthuri N, et al. Large-scale automatic reconstruction of neuronal processes from electron microscopy images; 2013, arXiv Preprint arXiv:1303.7186.

Liu T, Jurrus E, Seyedhosseini M, Ellisman M, Tasdizen T. Watershed merge tree classification for electron microscopy image segmentation. In: 21st International conference on pattern recognition (ICPR); 2012. p. 133–7.

Liu T, Jones C, Seyedhosseini M, Tasdizen T. A modular hierarchical approach to 3D electron microscopy image segmentation. J Neurosci Methods 2014;226(0):88–102, http://dx.doi.org/10.1016/j.jneumeth.2014.01.022.

Mishchenko Y, Hu T, Spacek J, Mendenhall J, Harris KM, Chklovskii DB. Ultrastructural analysis of hippocampal neuropil from the connectomics perspective. Neuron 2010;67(6):1009–20, http://dx.doi.org/10.1016/j.neuron.2010.08.014.

Nunez-Iglesias J, Kennedy R, Parag T, Shi J, Chklovskii DB. Machine learning of hierarchical clustering to segment 2D and 3D images. PLOS ONE 2013;8(8):e71715.

Schroeder W, Martin K, Lorensen B. The visualization toolkit: an object-oriented approach to 3D graphics. 4th ed. Kitware; 2006.

Seung HS. Eyewire; 2013 http://www.eyewire.org [accessed 18.12.14].

Seyedhosseini M, Sajjadi M, Tasdizen T. Image segmentation with cascaded hierarchical models and logistic disjunctive normal networks. In: Proceedings of the IEEE international conference on computer vision (ICCV 2013); 2013.

Toga AW, Rosen B, Wedeen VJ. The human connectome project; 2014 http://www.humanconnectomeproject.org [accessed 18.12.14].

Varshney LR, Chen BL, Paniagua E, Hall DH, Chklovskii DB. Structural properties of the *Caenorhabditis elegans* neuronal network. PLoS Comput Biol 2011;7(2):e1001066.

White JG, Southgate E, Thomson JN, Brenner S. The structure of the nervous system of the nematode *Caenorhabditis elegans*. Philos Trans R Soc Lond B 1986;314:1–340.

Yoo TS, Ackerman MJ, Lorensen WE, Schroeder W, Chalana V, Aylward S, et al. Engineering and algorithm design for an image processing API: a technical report on ITK – the insight toolkit. Stud Health Technol Inform 2002:586–92.