Multi-Scale Segmentation of Neurons Based on One-Class Classification

Paul Hernandez-Herrera [1], Manos Papadakis [1,2] and Ioannis A. Kakadiaris [1]

[1] *Computational Biomedicine Lab, Department of Computer Science, University of Houston, TX, 77204, USA*
[2] *Department of Mathematics, University of Houston, TX, 77204-3008, USA*

Abstract

Background: High resolution multiphoton and confocal microscopy has allowed the acquisition of large amounts of data to be analyzed by neuroscientists. However, manual processing of these images has become infeasible. Thus, there is a need to create automatic methods for the morphological reconstruction of 3D neuronal image stacks.

New Method: An algorithm to extract the 3D morphology from a neuron is presented. The main contribution of the paper is the segmentation of the neuron from the background. Our segmentation method is based on one-class classification where the 3D image stack is analyzed at different scales. First, a multi-scale approach is proposed to compute the Laplacian of the 3D image stack. The Laplacian is used to select a training set consisting of background points. A decision function is learned for each scale from the training set that allows determining how similar an unlabeled point is to the points in the background class. Foreground points (dendrites and axons) are assigned as those points that are rejected as background. Finally, the morphological reconstruction of the neuron is extracted by applying a state-of-the-art centerline tracing algorithm on the segmentation.

Results: Quantitative and qualitative results on several datasets demonstrate the ability of our algorithm to accurately and robustly segment and trace neurons.

Comparison with Existing Method(s): Our method was compared to state-of-the-art neuron tracing algorithms.

Conclusions: Our approach allows segmentation of thin and low contrast dendrites that are usually difficult to segment. Compared to our previous approach, this algorithm is more accurate and much faster.

Keywords: Neuron tracing, Segmentation, One-Class Classification

1. Introduction

Neurons are the main part of the nervous system. They allow processing and transmission of information. Thus, to understand the neuronal process at the cellular level, it is necessary to develop mathematical models allowing simulation of the neuronal function. In another direction, new research [1] suggests that anorexia nervosa affects the morphology structure of the neuron, such as the dendritic length and dendritic branches. Hence, it is necessary to trace the neuron from a 3D image stack to extract the morphology representation of the neuron. The first step of this process is the segmentation of the neuron from the background. Recent developments in confocal and multiphoton microscopy allow the acquisition of large volumes of neuronal images. Manual processing of these images is infeasible, as it would require an excessive amount of manual effort and would be likely to suffer from human errors. All of these reasons establish the need for the development of methods for the automatic segmentation of neurons from the 3D image stack. The main challenges to address when developing a segmentation algorithm are: (i) irregular cross-section (i.e., not semi-elliptical cross-sections such as those of vessels) of dendrites due to structures attached to the dendrites (spines); (ii) variability in the size of the dendrites to be segmented; (iii) thin dendrites can be as small as one voxel radii (depending on the voxel size); (iv) thin dendrites appear as low contrast objects; (v) contrast variation across different datasets due to different acquisition modalities; and (vi) noise.

The remainder of the paper is organized as follows: in Sec. 2, previous work in the segmentation of neurons is presented. In Sec. 3, an overview of our algorithm is presented, Sec. 4 our approach for neuron segmentation is presented with mathematical detail, Sec. 5 reports results in real datasets, and our conclusions are presented in Sec. 6.

2. Previous Work

Different approaches have been proposed to segment neurons from the background of the 3D image stack, including Machine Learning Algorithms (ML). The most common approach in ML is supervised learning where the user usually trains a support vector machine (SVM) classifier which allows separation of the training data in a high-dimensional space. The training data usually consist of two classes, positive samples corresponding to the structure of interest (dendrites) and negative samples corresponding to the background. The main difference between the various supervised learning approaches is the selection of the feature vector. Gonzalez *et al.* [2] used 3D steerable filters to create rotationally invariant feature vectors that are less sensitive to the irregularities of dendrites. Jimenez *et al.* [3]

proposed using isotropic low-pass, high-pass and Laplacian filters to compute a set of features that are computationally efficient. Santamaria-Pang *et al.* [4] used the eigenvalues of the Hessian matrix as descriptors to learn the local geometry of the dendrites. The main limitation of these approaches is the assumption that training and testing samples follow the same distribution, which may not be true due to the large variety in datasets (different imaging technologies and preparations with various resolutions and labeling methods [5]). These methods require re-training when the assumptions are not satisfied and a different model must be created for each dataset. This process is usually difficult since it requires properly selecting the positive and negative samples. In addition, the user needs previous knowledge in ML to set the correct parameters of the classifier. Furthermore, due to the variability in the size of the dendrites, these approaches usually have difficulty segmenting thin dendrites.

The segmentation process can also be implemented by thresholding. Janoos *et al.* [6] used a non-linear diffusion filter to de-noise the image. Thus, the neuron was segmented using a global threshold. Chothani *et al.* [7] proposed enhancing dendrites using a multi-scale center-surround filter and applying a threshold to the enhanced image to segment neurons. Then, a post-processing step was employed to remove regions with a small number of voxels. Xie *et al.* [8] used the triangle method over the histogram of the 3D image stack to compute a global threshold for segmentation. Then, a morphological closing operation was employed to fill artificial holes produced by the segmentation. Xiao *et al.* [9] used the average intensity of the 3D image stack as a threshold value to create an over-segmentation of the dendrites. These approaches assume that the staining is homogeneous, which is usually incorrect. As a consequence, these approaches are not robust to segmentation errors. In addition, it is difficult to know in advance the correct threshold value for segmentation. For a complete review of neuron segmentation algorithms see Meijering *et al.* [10] and Donohue *et al.* [11].

Recently, our team [12] proposed a semi-automatic method based on one-class classification for the segmentation of neurons. First, a monoscale isotropic Laplacian filter was proposed to detect a training set consisting of points belonging to the background. These points are used to train a single decision function that allows determining how similar an arbitrary point is to the points in the background class. Finally, the unlabeled points rejected as background are labeled as foreground. A limitation of [12] is that a-priori knowledge of the likely size of dendrites to be segmented is required to determine the suitable scale of the mono-scale isotropic Laplacian filters used in the training step. In addition, neurons with a high degree of heterogeneity in dendrite diameter are difficult to segment since a single scale will not cover the range of dendrite diameters.

In this paper, we propose a multi-scale approach that eliminates the drawbacks of our previous approach. The core philosophy of our approach is still the one-class classification segmentation approach [12], but the requirement of the user to provide the likely width of the neuron to segment is dropped. More specifically, our contributions in this work are the following:

(i) A multi-scale framework to segment the neurons.

(ii) A mathematically rigorous general approach for the normalization of the response of a multi-scale ensemble of linear filters motivated by the ad-hoc normalization of Gaussian low-pass filters used in [13].

(iii) A multi-scale framework to compute the Laplacian of the 3D image stack.

(iv) An approach to compute as many decision functions as the number of scales (one for each scale) used for segmentation.

(v) A mathematical justification for using different low-pass filters to compute the Laplacian and the Hessian matrix.

(vi) An extensive experimental evaluation of the performance of our approach on a number of datasets, including all of the DIADEM competition and the BigNeuron dataset.

Compared to [12], the method presented here (i) allows detection of neurons with a high degree of heterogeneity in dendrite diameter; (ii) allows a fair comparison of the response to the Laplacian filter at different scales; (iii) allows an automatic selection of samples belonging to the background of the 3D image stack using several scales and allows creation of a partition of the 3D image stack for each scale; and (iv) allows creation of specific decision functions for each scale. Due to these advantages over our previous approach, the Multi-scalE Segmentation Of Neuron (MESON) algorithm is less sensitive to the varying sizes of the dendrites to be segmented.

3. Materials

3.1. Data Acquisition and Segmentation

The first step of the analysis of neurons is the acquisition of the 3D image stacks. Neurons are imaged using microscopy and the voxels usually have anisotropic size where the $x-y$ dimensions have the same size but the z dimension usually has a different value than the $x-y$. This property creates an elliptical cross-section of the dendrites in the 3D image stack. Hence, models that assume circular cross-section usually fail to accurately detect dendrites. Another property due to the acquisition protocols is that dendrites usually have a decreasing intensity profile where the maximum value is reached at the center of the dendrite and the intensity decreases from the center to the boundary of the dendrite.

3.2. Segmentation

Figure 1 depicts an overview of our segmentation approach. Our algorithm consists of five steps to segment dendrites from a 3D image stack and it falls in the category of one-class classification algorithms.

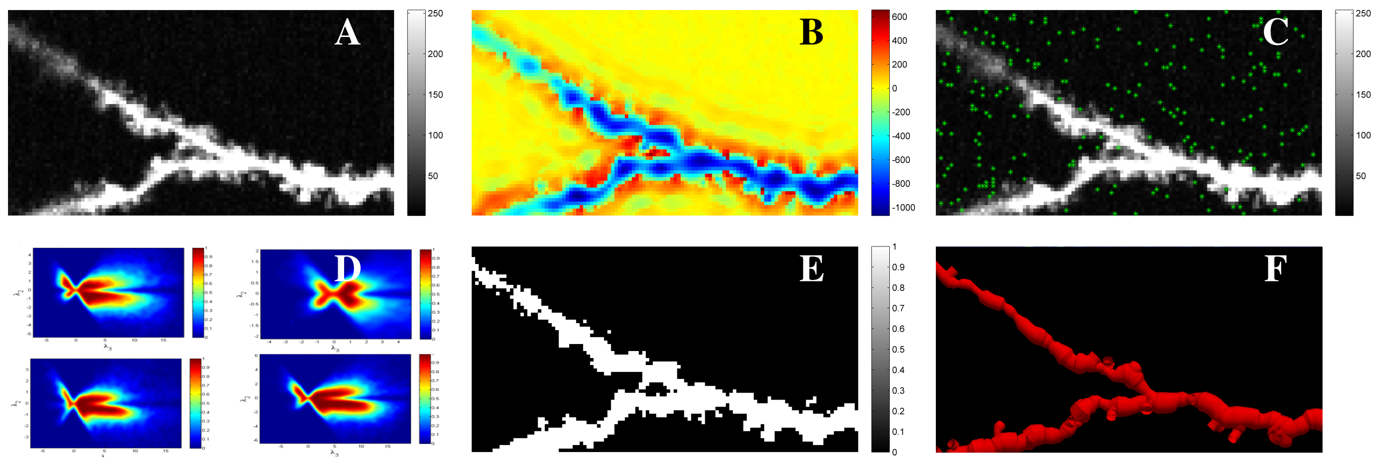


Figure 1: Algorithm overview. (A) depicts a slice of a 3D image stack; (B) the multiscale Laplacian calculated using the proposed method that allows to detect points in the background of the stack; (C) green points depict the automatically detected points (training set) in the background; (D) the training set allows automatic construction of a one-class classifier (decision functions) for each radii to detect; (E) the decision functions are used to classify whether points belong to the background; white depicts the rejected points that do not belong to the background; (F) the segmented volume is used to reconstruct the morphology of the neuron.

3.3. *Detect a training set of background points*

Given a 3D image stack, the first step of our algorithm is to detect points in the 3D image stack that reliably belong to the background of the 3D image stack. The second property is used to accurately detect points in the 3D image stack.

The Laplacian operator is a second-order differential operator. It has been widely used to detect sharp changes of intensity as those that occur at the boundary (edges) of the dendrites. To avoid noisy responses and obtain better responses, the Laplacian operator is usually calculated by convolving the input image with a low-pass filter. The classical approach to computing the Laplacian is using the Gaussian filter and it is denoted as the well known Laplacian of Gaussian (LoG). We employ a more general approach to compute the Laplacian where the family of Hermite Distributed Approximating Functional (HDAF) is used as the low-pass filter and the LoG is a special case of the HDAF. Due to the high variability of dendrite sizes, the Laplacian has to be computed using several radii (scales). One novelty of this paper is a mathematically rigorous approach to normalize the Laplacian (see **Proposition 1**) of the 3D image stack and to calculate the Laplacian using several radii (multiscale Laplacian).

The multiscale Laplacian defined in this paper has important properties. Specifically, (i) it has negative values and is close to zero near the inside of the boundary of the dendrite; (ii) it has positive values close to the outside of the boundary of the dendrite; (iii) it has negative values inside the neuron; and (iv) it has oscillations between positive and negative values on the exterior of the neuron. Figure 1(A) depicts a slice of a 3D image stack where one dendrite is bifurcating. Figure 1(B) depicts the multiscale Laplacian of the stack. Note that the multiscale Laplacian satisfies the previous properties. Specifically, it has negative values inside the neuron depicted in blue, positive values near the outside of the boundary depicted in red, and positive and negative values outside the dendrite depicted in yellow. The previous properties allow us to ensure that all the points from the multiscale Laplacian with positive values belong to the background of the 3D image stack. Figure 1(C) depicts an example of some points (green points) with positive Laplacian value; all of them are in the background of the 3D image stack. Note that there are negative values inside and outside the neuron due to properties (iii) and (iv). Hence, we need to develop a method that allow us to differentiate which negative values belong to the exterior of the neuron and which belong to the interior of the neuron. In the following sections, a method to identify which negative values belong to the exterior of the neuron is presented.

3.4. *Feature extraction*

One-class classification is a machine learning algorithm that, given a set of training samples, tries to identify objects similar to the objects in the training set. The objects that are identified as dissimilar to the objects in the training set are called outliers. The one-class classification algorithm requires extraction of features such that similar objects have similar features. It is well known that the eigenvalues of the Hessian matrix are good features to differentiate between different structures such as noise, tubular structures, blob structures and plate-like structures. These features have been used to enhance dendrites, vessels, arteries, airways, etc. Hence, the eigenvalues of the Hessian matrix are used as the features for the one-class classification algorithm. To account for the anisotropic voxel of the acquired images, the eigenvalues are computed using an anisotropic Gaussian filter.

3.5. *Learning from training set*

Density estimation is a popular one-class classification algorithm. In this approach the algorithm learns the distribution of the feature vector from the training set and the user sets a threshold for acceptance as a similar object. Thus, a new object is accepted as similar to the objects in the training set if the features of the object have a distribution value higher than the threshold. We employ this approach to detect objects similar to the objects in the training set. The distribution of the feature vector from the training set is learned using histogram since it has some advantages over other approaches such as (i) it is easier to compute, (ii) it is a fast approach and (iii) it provides good approximations. The distribution is transformed in order to normalize the values and smoothed to have smooth transition of the values of the distribution between neighbouring bins. Figure 1(D) depicts the learned decision functions for four different scales.

3.6. Segmentation of neurons

The constructed decision functions from the training set allow us to differentiate which values with negative Laplacian value belong to the background and which belong to the neuron. To this end, the features are calculated for each point that has negative Laplacian value. Then, the distribution value is calculated for the feature vector. Every point with distribution value larger than the automatically detected threshold is assigned to the background class, while points with values smaller than or equal to the threshold are assigned to the neuron. Finally, a post-processing step is employed to removed small structures. Figure 1(E) depicts the rejected points that do not belong to the background of the 3D image stack (white), while the black represents the points that were accepted by the decision functions as belonging to the background.

3.7. Centerline tracing

An state-of-the-art neuron tracing algorithm is used to extract the centerline from the segmented volume. Figure 1(F) depicts a visualization of the morphological recontruction of the 3D image stack. The reconstruction of the neuron is exported in the SWC file format.

4. Methods

In this section, each step of the proposed segmentation algorithm is explained in detail. Algorithm 1 describes the main steps of the proposed MESON. In this section, each step of the proposed algorithm is explained in detail.

```
Algorithm 1 MESON
Input: A 3D image stack $I$ and scales $\sigma_r,\ r = 1, 2, \ldots, l$
Output: Label 0 for background voxels and 1 for voxels belonging to neurons
    Step 1: Detect a training set of background points
    Step 2: Extract local shape information (features) from the 3D image stack $I$
    Step 3: Estimate a decision function for each detected training set
    Step 4: Detect unlabeled points that are rejected as background class for each scale $\sigma_r$
    Step 5: Post-process the labeling in Step 4.
```

4.1. Detect a training set of background points

A popular model for the cross-section of a dendritic branch, with axis parallel to the x-axis, is the radial 2-D Gaussian $g_\sigma(y,z) = e^{-\frac{y^2+z^2}{2\sigma^2}}$, where σ can be used to model the radius of the branch at a given point. We train the proposed algorithm to detect background points using the fact that close to the boundary of the dendritic branch the values of a certain Laplacian operator that we construct in this subsection are positive. We begin by observing that this property is valid for the 2-D Laplacian of a cross-section of any dendritic branch. Indeed,

$$\Delta_{y,z} g_\sigma(r) = \frac{1}{r}\frac{\partial}{\partial r}\left(r \frac{\partial\left(e^{-\frac{r^2}{2\sigma^2}}\right)}{\partial r}\right) = \frac{e^{-\frac{r^2}{2\sigma^2}}}{\sigma^2}\left(\frac{r^2}{\sigma^2} - 1\right).$$

This calculation implies that the "sign-change point" of $\Delta_{y,z} g_\sigma$ in the radial direction is located at radius σ. This radius coincides with the distance from the origin of the circle of all points of the cross-sectional plane at which the sign of $\Delta_{y,z} g_\sigma$ *changes to plus*. On the other hand, in the annulus $[\sigma, 2\sigma]$ the intensity values of g_σ decrease smoothly and become small, thus corresponding to a region which can be considered to be the "skin" of the dendritic branch. This model suggests that, indeed, the set of positive values of $\Delta_{y,z} g_\sigma$ in a cross-sectional plane of the branch belongs to the exterior of the branch. Our goal is to use a set of isotropic Laplacian filters to detect those points.

The Laplacian operator used is defined by convolution with a radial filter defined in the frequency domain in the following way:

$$\widehat{F^L_{n,\sigma}}(\xi) = -\|\xi\|^2 \widehat{F_L}(c^2_{n,\sigma}\|\xi\|^2), \quad (1)$$

where $\widehat{F_L}(x) = p_n(x)\exp^{-x}$, $x > 0$ is a low-pass filter [14] belonging to the family of the Hermite Distributed Approximating Functionals (HDAFs), the "roof" ($\widehat{\ }$) over the notation of a function denotes its normalized Fourier transform, $p_n(x) = \sum_{k=0}^{n} x^k/k!$ is the n^{th} order Taylor polynomial for the exponential function and $c_{n,\sigma} := \sqrt{2n+1}/(\sqrt{2}K\sigma)$. The parameters n and σ play an important role in the design of the low-pass filter. The choice of the constant $c_{n,\sigma}$ places the inflection point of the radial profile of $\widehat{F_L}(c^2_{n,\sigma}\|\cdot\|^2)$ firmly at radius $K\sigma$ from the origin, regardless of the value of n. As n increases to ∞ [15, Remark 3.4] the width of the radial profile of the transition band of $\widehat{F_L}(c^2_{n,\sigma}\|\cdot\|^2)$ is proportional to $\frac{1}{n}$. This transition band also contains the inflection point of the radial profile of $\widehat{F_L}(c^2_{n,\sigma}\|\cdot\|^2)$, for every n. Moreover, as n grows, the values of $\widehat{F_L}(c^2_{n,\sigma}\|\cdot\|^2)$ tend to 1 at every point in the ball centered at the origin with radius $K\sigma$ [15, Theorem 3.7]. In other words, the low-pass filter $\widehat{F_L}(c^2_{n,\sigma}\|\cdot\|^2)$ asymptotically behaves like an isotropic ideal filter. Note that the traditional Gaussian low-pass filter is a special case of the HDAFs obtained for $n = 0$.

Following the mathematical analysis of Proposition 2.3 [16], we infer that if K is selected large enough, in practice $K = 3$, and n is sufficiently large (see Section 5 for values of K and n used in our experiments), then the values of $g_\sigma * F^L_{n,\sigma}$ faithfully (in the uniform norm sense) approximate the values of $\Delta_{y,z} g_\sigma$. Thus, up to a certain radius sufficiently distant from the origin the sign of the values of $g_\sigma * F^L_{n,\sigma}$ will be the same as the sign of the values of $\Delta_{y,z} g_\sigma$, while farther away the sign of the values of the former function is no longer positive, as the errors of the approximation of $\Delta_{y,z} g_\sigma$ dominate the outcome. The previous analysis also explains the behavior of the multi-scale isotropic Laplacian filtering we next introduce.

Dendrites can be modeled locally as tubular structures of varying width where thick dendrites predominantly reside in low frequencies, while thin dendrites occupy higher frequencies in the frequency domain. Thus, the Laplacian filters can be customized to suit the anticipated sizes of dendrites by adjusting the cut-off frequency of the low-pass filter; by selecting a small scale $\sigma \in (0, 0.2)$, the filter keeps low frequencies and thus captures thick dendrites. Increasing the value of σ is more suitable for detection of thinner dendrites. To select the appropriate scale of the low-pass filter, the Laplacian filter (Eq. (1)) must be normalized to carry on a fair comparison of the response to the filter at different scales. In general, linear filtering is a convolution linear operator and as such is bounded. The operator norm determines the "magnitude" or, more precisely, the norm of the output. Thus, it is expected that linear operators with equal norms will have comparable outputs in the following sense: Given the same input, the "strength" of the output of linear filters compared with the same operator norm depends only on the qualitative properties of each filter. The operator norm of a linear filter acting on $L^2(\mathbb{R}^3)$, the space of all 3D images with finite energy, by $h \mapsto h * f$, with $h \in L^2(\mathbb{R}^3)$ is equal to $\|\hat{f}\|_\infty$, where $\|h\|_\infty = \sup\{|h(x)| : x \in \mathbb{R}^3\}$. Thus, to normalize the response of the linear filter induced by the kernel f, we use the kernel $f^N := \frac{f}{\|\hat{f}\|_\infty}$. To normalize the linear operators induced by the Laplacian filters $F^L_{n,\sigma}$ (Eq. (1)) we use the next proposition.

Proposition 1. *For every fixed integer $n \geq 0$, we have*

$$\|\widehat{F^L_{n,\sigma}}(\xi)\|_\infty = \frac{\sigma^2}{C}$$

where C is a constant that depends on n.

Since we fix n and we vary only σ we can treat C as a constant, at least during each experiment. The proof of Proposition 1 is in Appendix A.

The normalized isotropic Laplacian filtered output $L_{n,\sigma}(I)$ of the 3D image stack I at scale σ is given by:

$$\begin{aligned} L_{n,\sigma}(I) &= I * F^{L,N}_{n,\sigma} = \mathcal{F}^{-1}\{\widehat{F^{L,N}_{n,\sigma}}(\xi) \cdot \widehat{I}(\xi)\} \\ &= \mathcal{F}^{-1}\{\frac{\widehat{F^L_{n,\sigma}}(\xi)}{\sigma^2} \cdot \widehat{I}(\xi)\}, \end{aligned} \tag{2}$$

where $\mathcal{F}^{-1}$ is the inverse Fourier transform and $\widehat{I}$ is the Fourier transform of the 3D image stack I. Next, we define the multi-scale isotropic Laplacian of the 3D image stack I by:

$$L^M_n(I)(x) = L_{n,\sigma_{\max}(x)}(I)(x), \tag{3}$$

where

$$\sigma_{\max}(x) = \underset{\sigma\in\{\sigma_1,\sigma_2,\ldots,\sigma_l\}}{\operatorname{argmax}} \{|L_{n,\sigma}(I)(x)|\} \tag{4}$$

is the scale at which the response of the Laplacian filter reaches the maximum absolute value and $\{\sigma_1, \sigma_2, \ldots, \sigma_l\}$ are the scales at which the dendrites are expected to fall. Note that each position x in the 3D image stack I is assigned to a single scale $\sigma_{\max}(x)$. Hence, the operator L^M_n is not necessarily linear and the scale selected to compute the multi-scale Laplacian creates a partition of the 3D image stack I. This partition, which plays an important role in the extraction of the feature vectors and the construction of the decision function, is defined by:

$$P(\sigma_r) = \{x | \sigma_{\max}(x) = \sigma_r\}, r = 1, 2, \ldots, l. \tag{5}$$

Figure 2(a) depicts a 3D image stack I of a neuron acquired with multiphoton microscopy and the red square depicts a subvolume used to illustrate the steps of our approach. Figure 2(b) depicts the 2D maximum intensity projection of the subvolume along the $z-$axis. The multi-scale Laplacian (Eq. (3)) of the sub-volume is depicted in Figure 2(c) where four scales ($\sigma_1 = 1.5$, $\sigma_2 = 2$, $\sigma_3 = 3$, and $\sigma_4 = 4$) were used to compute the multi-scale Laplacian. Figure 2(d) depicts the partition (Eq. (5)) of the sub-volume where each color identifies the assigned partition for each scale. Note that thin dendrites are assigned to the smaller scale (blue) while thick dendrites are assigned to higher scales (red and yellow).

According to the theoretical analysis in the first half of this section, the multi-scale Laplacian $L^M_n(I)$ of the 3D image stack I (Eq. (3)) has the following properties: (i) its values are close to zero at the boundary of the neuron; (ii) it is negative inside the neuron; (iii) it is positive near the exterior of the neuron; and (iv) it produces oscillations between positive and negative values on the exterior of the neuron.

From properties (ii) and (iv), it follows that any x with positive response to the Laplacian operator belongs to the exterior of the neuron. In addition, properties (i) and (iii) imply that there are positive values close to the boundary of the neuron, which is important for having a representative training set of background points. From the previous observations, a training set B consisting of background points is selected by:

$$B = \{x \,| L^M_n(I)(x) > 0\}, \tag{6}$$

where x is a point (voxel) in I. Note that a training set of background points can be defined for each scale σ_r by

$$B(\sigma_r) = \{x \in B \ \wedge P(\sigma_r)\}, \ r = 1, 2, \ldots, l. \tag{7}$$

Figures 3(a-d) depict the detected training set $B(\sigma_r)$ (light green) and unlabeled samples (red) for each scale $\sigma_1, \sigma_2, \sigma_3, \sigma_4$, respectively. This partition corresponds to Figure 2(d).

4.2. *Feature extraction*

Dendrites locally resemble a tubular structure, hence we employ features capturing tubular shape information. The eigenvalues ($|\lambda_1| \leq |\lambda_2| \leq |\lambda_3|$) of the Hessian matrix H are employed as the feature vector since it is well known that they provide local geometric information of the structure. The Hessian matrix

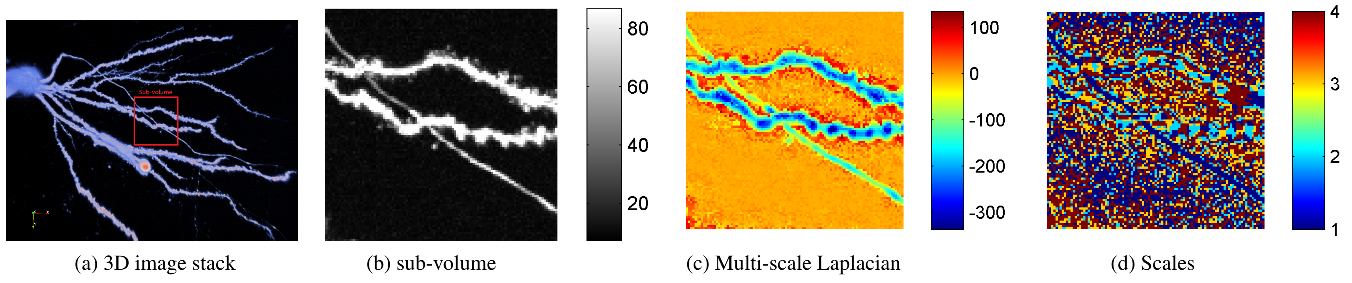


Figure 2: (a) Volume rendering of a 3D image stack I acquired with multiphoton microscopy; (b) Maximum Intensity Projection (MIP) of a sub-volume from the 3D image stack I; (c) illustration of the multi-scale Laplacian filter of the sub-volume; and (d) depiction of the scales at which the Laplacian reaches the maximum absolute response.

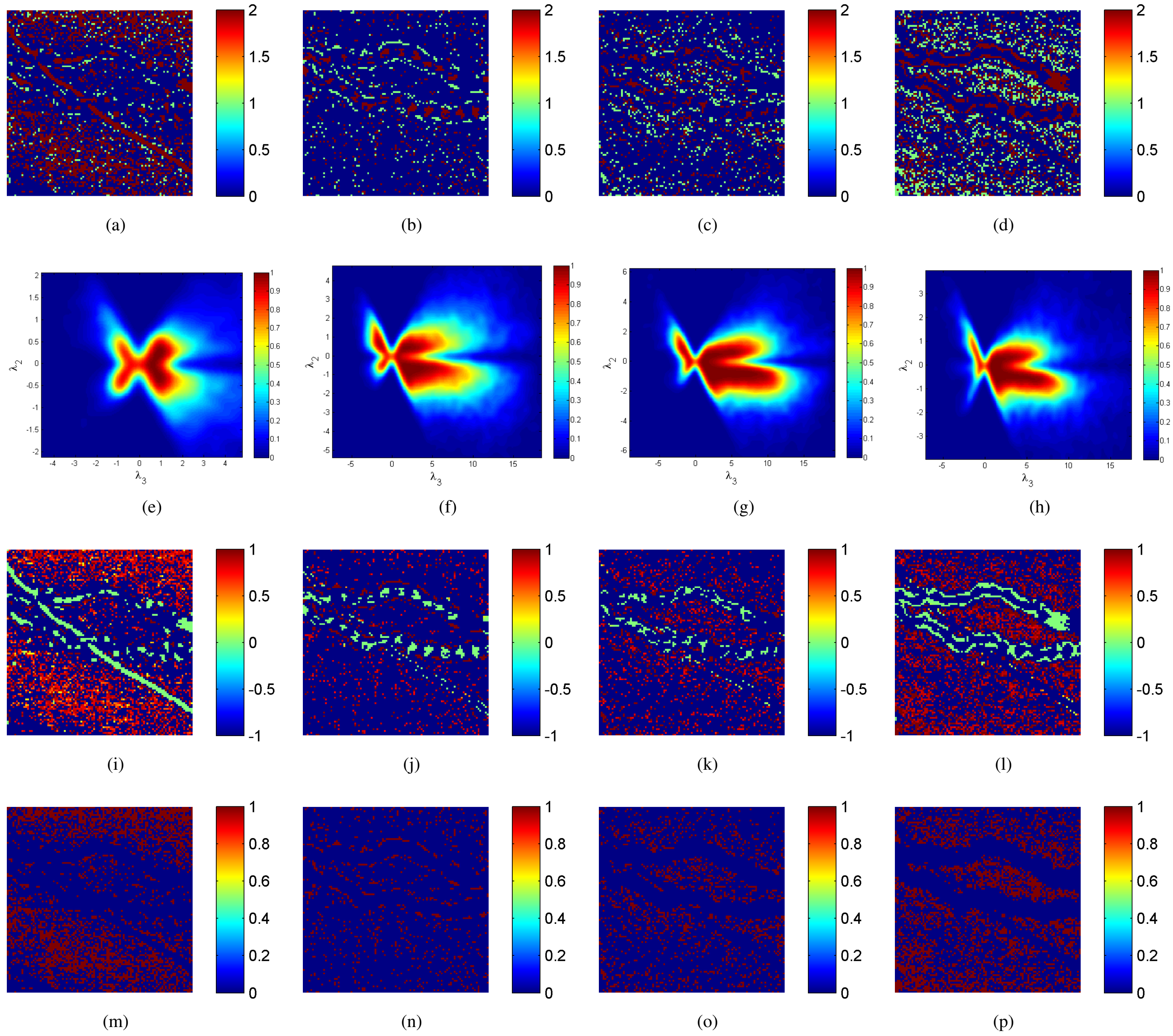


Figure 3: (a-d) Detected training set B_{σ_r} (light green) (Eq. (7)) and unlabeled points (red) for each partition P_{σ_r}; (e-h) learned decision function (Eq. (11)) for each training set B_{σ_r}; (i-l) the values of the similarity function $E_{P(\sigma_r)}(x)$ (Eq. (12)) for each partition P_{σ_r}; and (m-p) the detected background points (red) for each partition P_{σ_r}.

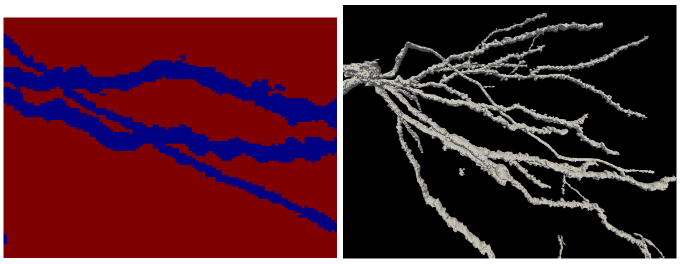


Figure 4: (a) Background segmentation (red) of the sub-volume and (b) depiction of the dendrite segmentation (gray) of the 3D image stack I.

of the 3D image stack I is computed as:

$$H_{ij}^{\sigma_w}(x) = \frac{\partial^2 (I * G_{\sigma_w})}{\partial_{x_i}\partial_{x_j}}(x), \tag{8}$$

where G_{σ_w} is a Gaussian low-pass filter at scale σ_w. Note that, similar to the previous section, the user must select the appropriate scale of the Gaussian filter at a given position x. The partition P (Eq. (5)) allows selection of the correct scale of the Gaussian low-pass filter, if $x \in P(\sigma_r)$, thus $\sigma_w = \sigma_r$.

Note that the Laplacian filter (Eq. (2)) and the Hessian matrix (Eq. (8)) are computed using different low-pass filters. By using the same low-pass filter to compute the Laplacian and Hessian matrix, the automatic selection of the training set is essentially conditioned on the features and thus the discriminative information to segment the dendrites will not be strong enough. Using the training set we compute the conditional density of the features and the segmentation problem is subsequently reformulated as a conditional expectation problem, where the expectation is computed using the conditional density learned from the training set. This prevents us from using the features whose conditional joint density we need to learn to define the training set. In other words, we cannot use a term we wish to define in the definition of the term. This is due to the following relation:

$$\begin{aligned} L(x) &= Tr(H^{\sigma_w}(x)) \\ &= H_{11}^{\sigma_w}(x) + H_{22}^{\sigma_w}(x) + H_{33}^{\sigma_w}(x) \\ &= \lambda_1(x) + \lambda_2(x) + \lambda_3(x). \end{aligned}$$

Hence, the training set B (Eq. (6)) depends on the feature vector by the relation

$$B = \{x | \lambda_1(x) + \lambda_2(x) + \lambda_3(x) > 0\}.$$

The relation between the training set and the feature vector no longer holds if different low-pass filters are used. This is the main motivation to use different low-pass filters to compute the Laplacian and the Hessian matrix.

4.3. *Learning decision function for each scale*

Most of the previous work created explicit functions from the feature vector to enhance tubular structures by using the property that, for an ideal tubular structure, $\lambda_1(x)$ is low and $\lambda_2(x)$ and, $\lambda_3(x)$ have large negative values ([17]). In this work, dendrites are indirectly enhanced by creating an implicit decision function from the feature vector that assigns values close to one to the background and values close to zero to the foreground. To this end, a three step approach is used to generate the decision function for each set $B(\sigma_r)$ in the partition of the training set B: (i) compute the empirical distribution of the eigenvalues corresponding to the training set $B(\sigma_r)$; (ii) transform the distribution using a monotonic function to create the decision function; and (iii) smooth the decision function.

Estimate background distribution: The distribution of the two eigenvalues[1] $\lambda_2(x)$ and $\lambda_3(x)$ from the training set $B(\sigma_r)$ provides information on which points should be assigned with high probability to belong to background. The higher the value of the distribution, the more likely the point belongs to the background. The distribution of the eigenvalues $\lambda_2(x)$ and $\lambda_3(x)$ from the training set $B(\sigma_r)$ is approximated using a 2D histogram. We prefer to use a histogram instead of a more complex representation such as kernel density estimation because: (i) it is easy to compute; (ii) the training set $B(\sigma_r)$ may have millions of points depending on the size of the 3D image stack I (computing the distribution using kernel density estimation is very slow. Histograms are very fast to compute even if there are millions of points); and (iii) it provides a good approximation of the distribution by selecting an appropriate bin size. The histogram is computed as:

$$D_{\sigma_r}(i, j) = c([b_{1,i}, b_{1,i+1}) \times [b_{2,j}, b_{2,j+1})), \tag{9}$$

where $c(R)$ represents the number of feature vectors that belong to the two-dimensional interval R, $b_{s,k} = m_s + k\frac{M_s - m_s}{N}$; N is number of bins for the histogram, and m_s, M_s are the minimum and maximum values of the $s-$feature in the training set $B(\sigma_r)$, respectively.

Transform the background distribution: The histogram D_{σ_r} has integer values since it counts the number of feature vectors falling in a given bin. The values of the histogram are transformed such that the minimum and maximum value in the histogram are zero and one, respectively. A decision function $D_{\sigma_r,T}$ from the background distribution D_{σ_r} is constructed by applying the transformation

$$D_{\sigma_r,T}(i, j) = \frac{1 - \exp^{-D_{\sigma_r}(i,j)}}{1 + \exp^{-D_{\sigma_r}(i,j)}}. \tag{10}$$

This transformation assigns values close to one to any bin with $D_{\sigma_r}(i, j) \geq 6$ while assigning value zero to bins with $D_{\sigma_r}(i, j) = 0$. The motivation behind this transformation is to ensure that if a two-dimensional interval has at least six feature vectors from the training set falling on it, any unlabeled feature vector falling in the interval would be assigned to background class. In addition, this transformation allows us to ensure that two-dimensional intervals with zero feature vector would be assigned value zero and therefore any unlabeled feature vector falling on it would not be assigned to background class.

[1]We use only the two largest eigenvalues since they provide enough shape information and we avoid the additional computational effort to compute the 3D distribution.

Smoothing the decision function: A disadvantage of histograms is that they result in piecewise constant functions. Hence, $D_{\sigma_r,T}$ is such a function. Therefore, there can be regions having value zero but they may have close regions with value one. To alleviate those discontinuities, $D_{\sigma_r,T}$ is smoothed by convolving it with a Gaussian kernel

$$D_{\sigma_r,s} = D_{\sigma_r,T} * G_{\sigma_s}. \quad (11)$$

Figures 3(e-h) depict the learned decision function $D_{\sigma_r,s}$ for each detected training set $B(\sigma_r)$. Note that the proposed decision function assigns high values to regions close to zero. This is mainly because, according to [17], the eigenvalues of noise samples (background) have low values.

4.4. *Segmentation of Neurons*

For each set $P(\sigma_r)$ in the partition, a function $E_{P(\sigma_r)}$ is created that measures how similar a point $x \in P(\sigma_r)$ is to the background

$$E_{P(\sigma_r)}(x) = \begin{cases} 1 & \text{if } x \in B(\sigma_r) \\ D_{\sigma_r,s}(Q(\lambda_2(x), \lambda_3(x))) & \text{otherwise} \end{cases}, \quad (12)$$

where $Q(\lambda_2(x), \lambda_3(x))$ indicates the bin position of $(\lambda_2(x), \lambda_3(x))$. Figures 3(i-l) depict the corresponding values of $E_{P(\sigma_r)}(x)$ for each $x \in P(\sigma_r)$. Note that points x in the background are assigned to values close to one (red) while points in the foreground are assigned values close to zero (light green) as expected.

A point $x \in P(\sigma_r)$ is assigned to the background class if the value of function $E_{P(\sigma_r)}$ is higher than a threshold T_{σ_r}:

$$S_{P(\sigma_r)} = \{x | E_{P(\sigma_r)}(x) > T_{\sigma_r}\}, \quad (13)$$

which the user can vary to optimize results. The threshold T_{σ_r} controls the accuracy of the classification on the training set $B(\sigma_r)$ by correctly classifying $x \in B(\sigma_r)$ if the value $D_{\sigma_r,s}$ associated to x is larger than T_{σ_r}, otherwise x is classified incorrectly. The threshold T_{σ_r} is automatically set by selecting the value that correctly classifies 99.9% of the training set as background. Note that the segmentation depends on the threshold T_{σ_r}; by selecting a threshold with less accuracy, more background would be classified incorrectly and noise would be incorporated in the segmentation. Thus, the set of points that belongs to the background of the 3D image stack is given by:

$$S_B = S_{P(\sigma_1)} \cup S_{P(\sigma_2)} \cup \cdots \cup S_{P(\sigma_l)}, \quad (14)$$

which is the union of all the points that were detected as background in each partition P_{σ_r}. Finally, the function used to segment the dendrites is defined as:

$$S_D = \neg S_B \vee (I > T_1), \quad (15)$$

where the first term detects dendrites as those points that are rejected as background points and the second term is a threshold operator ($T_1 = 0.90 \times \max_x \{I(x)\}$) that allows detection of dendrites that exceed the maximum radii provided for the scales. Note that the second term is necessary to segment thick dendrites in cases where few images are provided for the z-axis. For example, assume that n images are provided for the z-axis. Then, the maximum value allowed for the scales is n/y since any scale exceeding this value would exceed the number of images provided in the z-axis. Thus, any dendrite with radii larger than n/y would not be detected by the first term. However, it would be detected by the second term since the thicker the dendrite, the higher the intensity. Figures 3(m-p) depict the detected points in the background $S_{P(\sigma_r)}$ for each scale. Figure 4(a) depicts the segmentation of the background S_B. Figure 4(b) depicts segmentation of the neuron for the 3D image stack I. Note that thin and thick dendrites are detected and also low contrast dendrites are segmented.

4.5. *Post-processing*

The segmentation S_D may still identify false dendrites, which usually have few voxels. They are detected by identifying connected components using a 26-connected neighborhood and disregarding components with fewer than c_m voxels.

4.6. *Centerline tracing*

The morphological reconstruction of the neuron is achieved by applying the centerline tracing algorithm of [4] to our segmentation. Note that the morphological reconstruction of the neuron is represented in the standard SWC file format [18].

5. Results

In this section, qualitative and quantitative results of the performance of the proposed approach are presented on multiple datasets. The publicity available DIADEM [5] and the BigNeuron [19, 20, 21] datasets were used to evaluate our algorithm. The DIADEM dataset was release in 2012 and the 3D image stacks are provided from different laboratories with a variety of acquisition modalities, resolutions and labeling methods. Hence, it is a representative dataset of the challenges to be faced by neuron morphology tracing algorithms. The BigNeuron dataset is a recent dataset (2015), which incorporates new challenges to be addressed by neuron tracing algorithms. Furthermore, one additional dataset acquired with confocal microscopy is used to compare our MESON algorithm against our previous approach (OCCEN).

Qualitative results are presented for four (Cerebellar Climbing Fiber (CF), Hippocampal CA3 Interneuron (CA3), Neocortical Layer 1 Axons (NL), Visual Cortical Layer 6 Neuron (VC6)) of the six DIADEM datasets, six datasets (chick_uw, frog_scrippts, fruitfly_larvae_gmu2, human_allen_confocal, human_culturedcell_Cambridge and mouse_RGC_uw) of the BigNeuron dataset, and one confocal dataset. In addition, quantitative results are presented in two (Neuromuscular Projection Fibers (NP), and Olfactory Projection Fibers (OP)) DIADEM datasets.

Parameter Settings: Our approach requires four parameters n, N, σ_s and c_m. These parameters were fixed to $n = 60, N =$

500 and $\sigma_s = 5$ for all the experiments. We choose $n = 60$ because it provides a small transition band. The parameter N depends on the size of the training set B_{σ_r}: the larger the training set, the smaller the bin size to obtain a good approximation of the distribution. Hence, 1M samples from B_{σ_r} were randomly selected and N was fixed to 500; we set $\sigma_s = 5$ because it is an adequate value for the size of the histogram. The parameter $c_m = (2\times\sigma_{r_{\max}})^3$ where $\sigma_{r_{\max}}$ is the larger scale used for segmentation. The number of scales used for segmentation affects the segmentation; increasing the number of scales would improve the segmentation. However, there are usually no large differences between the segmentations beyond ten scales. Therefore, ten scales are usually used in our experiments.

5.1. Qualitative results

In this section qualitative results are presented in the DIADEM, BigNeuron and Confocal datasets to demonstrate the robustness of our approach to handle different datasets. The Confocal dataset is used to qualitatively compare our algorithm against our previous approach (OCCEN).

CF - DIADEM dataset: The 3D image stacks from this dataset have three channels. The third channel is used as our input image I since the blob structures are attenuated in this channel. Furthermore, this dataset is highly anisotropic since the z-distance between successive images within an image is 8.8 pixels, creating a spacing of [1 1 8.8] voxels for the x, y and z-axis, and only 34 images are provided for the z-axis. Figure 5(a) depicts the Maximum Intensity Projection (MIP) along the z-axis of a 3D image stack from this dataset, the manual reconstruction (green), and the reconstruction from our software (red). The total reconstruction time was *1h 11min*. Note that our reconstruction highly agrees with the manual reconstruction. In addition, our reconstruction is able to detect branches that were not traced in the manual reconstruction (blue squares). The scales for the segmentation are $\sigma_r = \{2, 3, 4, \ldots, 10, 11, 12\}$.

CA3 - DIADEM dataset: The neurons from this dataset appear as black structures in the raw images, hence the intensity of the images is inverted to depict neurons as bright structures. Furthermore, some dendrites appear as a sequence of unconnected dendrites due to gaps. In addition, the dendrites are very elongated along the z-axis due to the anisotropic ratio 1:1:8. Figure 5(b) depicts a comparison between the reconstruction from our algorithm (red) and the manual reconstruction (green). Note that both reconstructions are very similar; this is mainly because our segmentation algorithm is able to detect the bright tubular structures in the 3D image stack. In addition, our method is able to detect some dendrites (blue squares) missed in the manual reconstruction. However, note that our algorithm missed a branch traced in the manual reconstruction (yellow square). This is mainly because there is a very large gap in the dendrite. The scales used for segmentation are $\sigma_r = \{1.5, 1.6, 1.7, \ldots, 3.0\}$. Due to the highly anisotropic ratio, some false dendrites are detected along the z-axis that are not removed by the default parameters of **Step 5**. Hence, for this dataset the parameter C_m is manually selected to remove those false dendrites. The total reconstruction time was *3h 55min*. Note that this is a big stack with size $4824 \times 2655 \times 110$ voxels.

VC6 - DIADEM dataset: The neuron from this dataset was acquired with transmitted light bright-field microscopy and the z-distance between successive images is 2.93 pixels. Figure 6(a) depicts the minimum intensity projection of the 3D image stack from the VC6 dataset and the reconstruction from our algorithm where it has been shifted to better visualize the neuron, blue squares identify low contrast dendrites that were accurately segmented and traced by our algorithm, while the yellow square depicts a thick dendrite accurately detected by our algorithm. The computational time was *14m 49s*.

NL - DIADEM dataset: This is a challenging dataset because dendrites are crossing and there are gaps in the axons due to low signal-to-noise ratio; this may lead to topological errors in the reconstruction such as incomplete tracings and incorrect labeling of axons at cross-over. Figure 6(b) depicts a maximum intensity projection of a 3D image stack from the NL dataset; the labeling and tracing of each axon is depicted using different colors. Our algorithm is able to accurately extract the topology of each axon due to the centerline tracing algorithm because the tracing algorithm from [4] is designed to correct this type of topological error. The reconstruction time was *1m 33s*.

BigNeuron dataset: The BigNeuron dataset is a new publicly available dataset and it should be used as a baseline for testing neuron tracing algorithms. Figure 7 depicts the results of the neuron tracing for six different datasets of the BigNeuron. Red lines correspond to the reconstruction from the proposed algorithm. The results are fairly good for detecting the dendrites of the datasets. The proposed approach is able to detect thin and thick dendrites for each of the datasets. Additional visualization of the tracings for the different stacks of the BigNeuron dataset can be found in Supplementary Figures 2-11. Given a dataset, the scales used for our segmentation algorithm were fixed for each stack of the dataset to the minimum and maximum radii to detect.

MESON against OCCEN - CONFOCAL dataset: To demonstrate the advantage of using a multi-scale approach rather than a single approach, MESON is compared to OCCEN in a 3D image stack with a high degree of heterogeneity in dendrite diameter. Figure 8(a) depicts a volume rendering of the 3D image stack that includes the following properties: (i) high degree in dendrite diameter, (ii) high and low contrast dendrites, and (iii) high and low curvature of dendrites. Figure 8(b) depicts the segmentation from MESON where the scales used for segmentation were $\sigma_r = \{ 0.5, 1.0, 1.5, 2.0, \ldots, 4.5, 5.0\}$. Figures 8(c–e) depict the segmentation from OCCEN where the scales used for segmentation were $\sigma = 1, 1.5$, and 2, respectively. Note that MESON is able to segment thin dendrites (red squares) while OCCEN fails to segment them. In addition, by increasing the scale of OCCEN, more dendrites are not segmented. This is the main weakness of our previous approach since given a single scale σ, it has difficulty segmenting dendrites with radii smaller than σ. Furthermore, it has difficulties segmenting dendrites with radii much larger than σ since it creates a hollow at thick dendrites. Then, thick dendrites are traced as two parallel dendrites, which results in incorrect tracings.

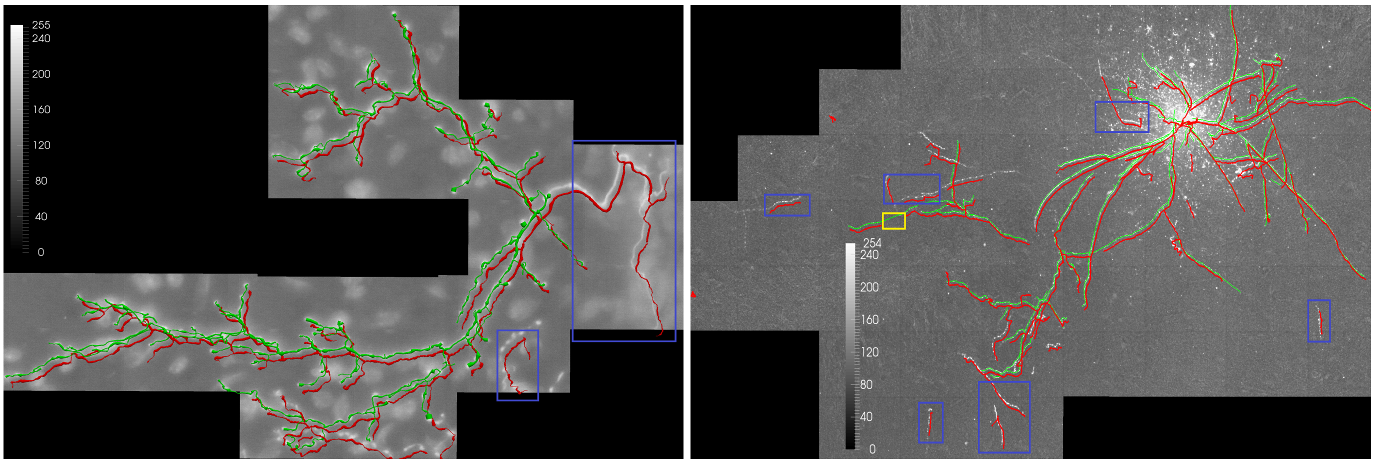

(a) DIADEM CF (b) DIADEM CA3

Figure 5: Qualitative results. (a,b) MIP along the z-axis of the 3D image stacks from DIADEM CF and CA3, respectively. Manual reconstruction (green), the reconstruction from our software (red), branches detected by our algorithm but missed in the manual reconstruction (blue squares), and a part of a dendrite missed by our algorithm (yellow square).

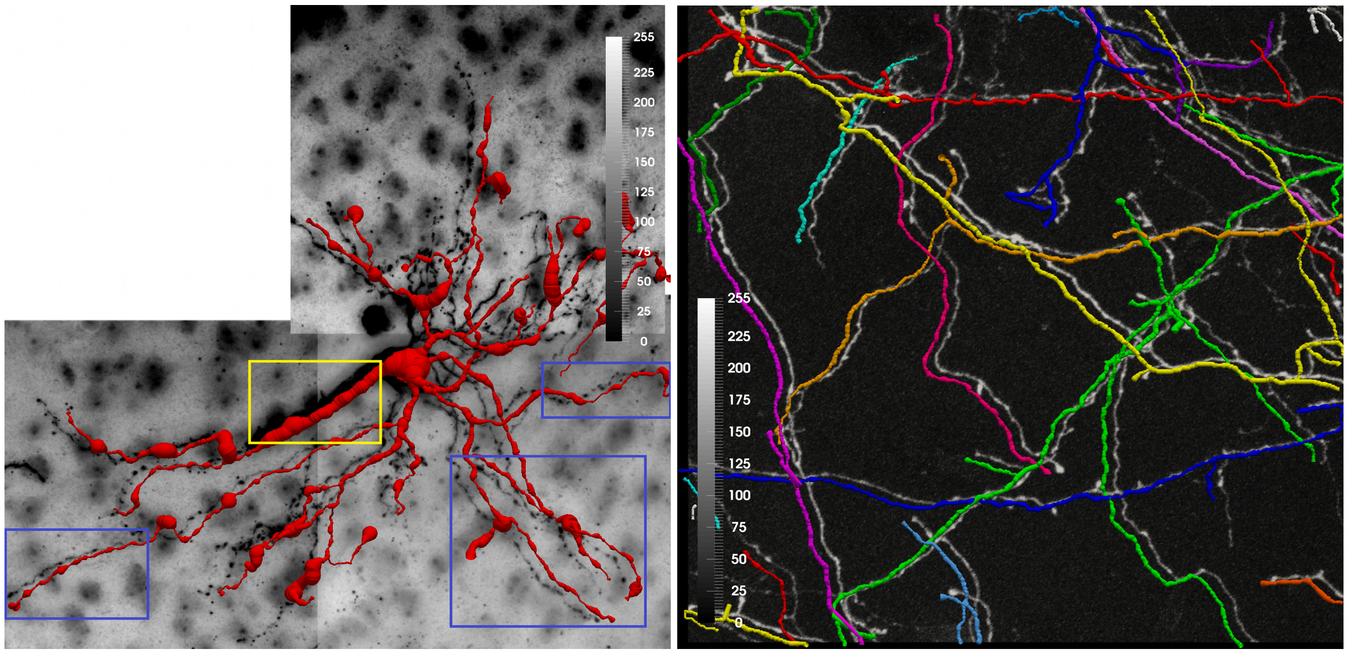
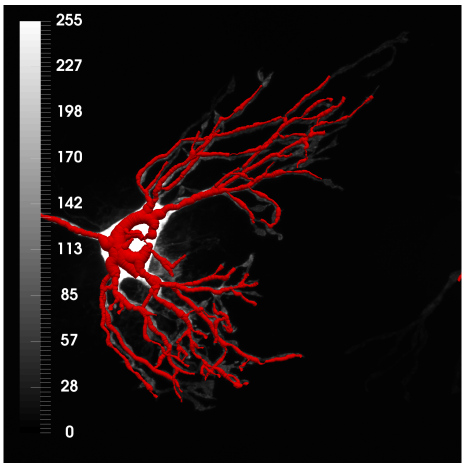

(a) DIADEM VC6 (b) DIADEM NL

Figure 6: Qualitative results. (a) Minimum Intensity Projection along the z-axis of a 3D image stack from the DIADEM VC6 and reconstruction by our software (red), blue squares identify low contrast dendrites acurately detected by our algorithm and the yellow square identifies a thick dendrite; (b) MIP along the z-axis of a 3D image stack from the NL dataset, different tracings labeled with different colors.

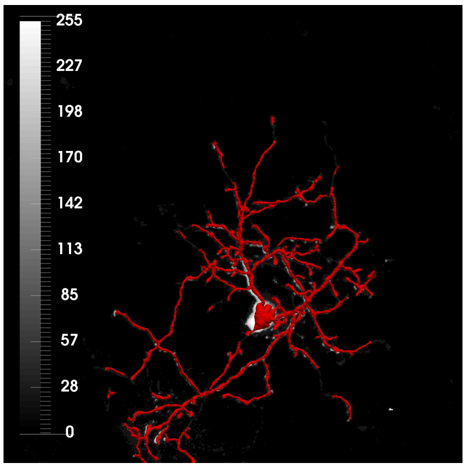
5.2. *Quantitative results*

Our method was quantitatively compared to state-of-the-art neuron reconstruction algorithms (Neurostudio ([22]), APP2 ([9]), Neutube ([23]), ORION 2[3], ORION [4] and OCCEN [12]). The default parameters were used for each of the algorithms. The parameter settings for our segmentation algorithm were fixed as described previously, and we did not include manual interaction. Even though tuning some of the parameters can increase the accuracy of our segmentation for each dataset, such as the value c_m used for removing small structures or the threshold values T_{σ_r} and T_l used for the detection of dendrites (Supplementary Figure 1 depicts an example of reconstruction with default parameters and reconstruction with the parameters tuned).

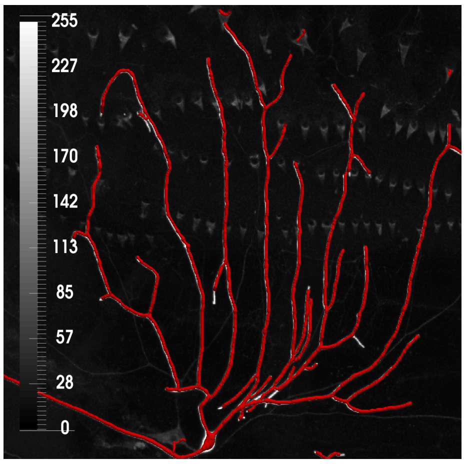
To quantitatively measure the performance of our proposed algorithm, the morphological reconstruction of the neuron (traced centerline) is used for comparison. The metrics used for comparison are Precision (P), Recall (R) and Miss-Extra-Score



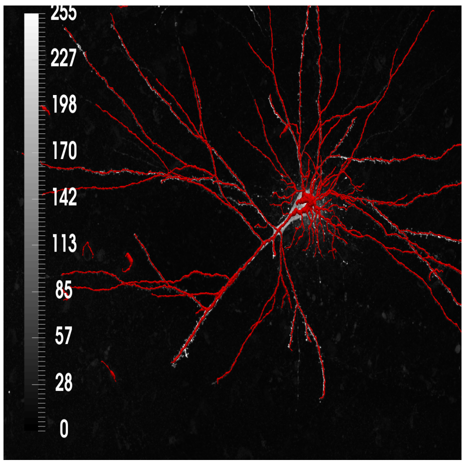
(a) checked6 chick uw case1 dataset - BigNeuron



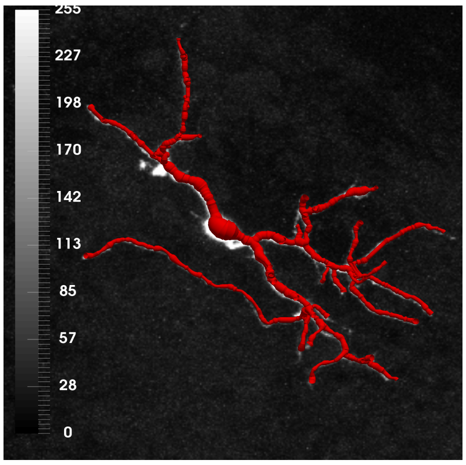
(b) checked6 frog scrippts dataset - BigNeuron



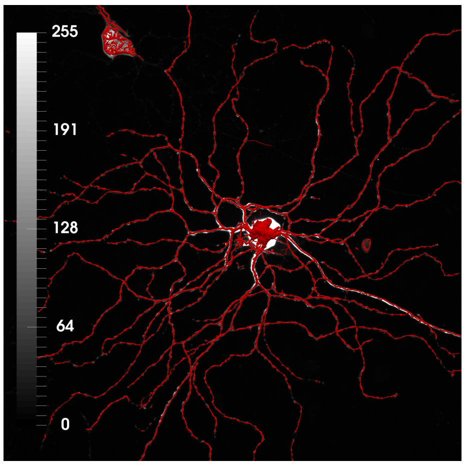
(c) checked6 fruitfly larvae dataset - BigNeuron



(d) checked6 human allen dataset - BigNeuron

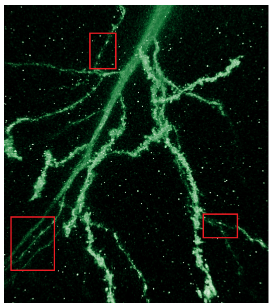


(e) checked6 human cultured cell dataset - BigNeuron

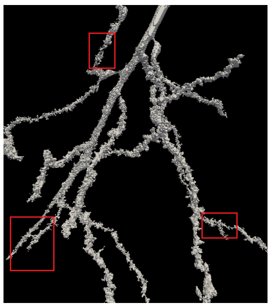


(f) checked6 mouse RGC uw dataset - BigNeuron

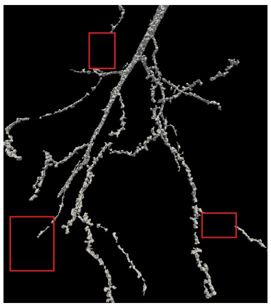
Figure 7: Qualitative results BigNeuron dataset. (a-f) Tracing results of case1 slide2, Recon112012, 1_CL-I_X_OREGON_R_ddaD_membrane-GFP, in_house1, image7, and ho_091202c2 stacks from the BigNeuron Project.



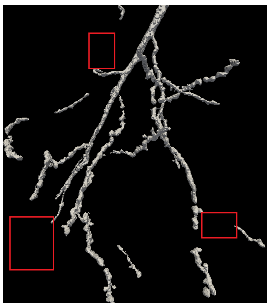
(a) 3D image stack



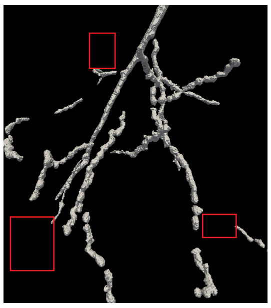
(b) MESON



(c) OCCEN $\sigma = 1$



(d) OCCEN $\sigma = 1.5$



(e) OCCEN $\sigma = 2$

Figure 8: Comparison of MESON over OCCEN. (a) Volume rendering of a 3D image stack with high degree of heterogeneity in dendrite radii. (b) Segmentation result of MESON where red squares depict thin dendrites that are correctly segmented. (c,d,e) Segmentation result of MESON where the scales used for segmentation were $\sigma = 1, 1.5$ and 2, respectively.

(MES):

$$P = \frac{S_C}{S_t}, \quad R = \frac{S_C}{S_C + S_m} \quad \text{and MES} = \frac{S_G - S_m}{S_G + S_e},$$

where S_C is the total length of the correct segments from the traced centerline, and S_t and S_G are the total length of the traced centerline and manual reconstruction, respectively. The total length of extra segments in the traced centerline is denoted by S_e. The total length of missing segments from the ground truth is denoted by S_m. Given a point p_i in the correct segments from the traced centerline, the displacement error of p_i is defined as the minimum Euclidian distance between the point p_i and all the points in the ground truth. The metric Precision measures the percentage of the centerline that is correctly detected, Recall is a metric that penalizes the traced centerline if it misses segments (the algorithm fails to detect dendrites), and MES penalizes the extra and missing parts from the centerline.

It is important to point out that since the automatic reconstruction is performed on a 3D grid there may be small errors between the manual reconstruction and the traced centerline. To account for this error, a point p_e in the centerline is classified correctly if there is a point p_g in the manual reconstructions such that $\|p_e - p_g\| \leq C$, where C is a constant defined by the user. The constant C used for computing the metrics is the same as those used in [3]. For the OP dataset, the value $C = 3$ is used while $C = 7$ is used for the NP dataset.

MESON against OCCEN. NP - DIADEM dataset: First, the proposed algorithm to trace neurons is quantitatively compared against our previous approach [12] in the NP dataset. Table 1 presents the average value for Precision, Recall and MES for 30 image stacks from the NP dataset. We only report results on 30 out of 152 stacks, since the gold standard misses some of the fibers for the remaining stacks as pointed out by [3] and this affects the value of the metrics. Methods A1-A6 represent our previous approach where the scale used for segmentation was $\sigma = 1.5, 2.0, 2.5, 3.0, 3.5$ and 4.0, respectively. Method D corresponds to our multi-scale approach where the scales used for our segmentation algorithm were $\sigma_r = \{1.5, 1.75, 2.0, ..., 5.0\}$. Note that our multi-scale framework has the highest value for the metrics Precision and MES. The highest value for the metric Recall is achieved by OCCEN with $\sigma = 1.5$. However, OCCEN with $\sigma = 1.5$ produces the lowest value for the metric Precision indicating that the tracing is creating the most spurious branches among the other reconstructions. Furthermore, the metric MES for OCCEN with $\sigma = 1.5$ is lower than the multi-scale approach, indicating that it produces more branches and it missed more branches than our multi-scale framework. Hence, the quantitative results demonstrate that the multi-scale approach outperforms the single scale. The computational time to trace the NP dataset was $5h$ $27min$, $4h$ $54min$, $5h$ $07min$, $4h$ $41min$, $5h$ $10min$ and $6h$ $11min$ by the OCCEN algorithm for each scale. Hence, the total time to extract the 180 (six for each image stack since six scales are analyzed in the single scale) reconstructions was $31h$ $30min$. The multi-scale approach took $1h$ $21min$ to reconstruct the 30 image stack. In addition to the quantitative advantage of our multi-scale scale, our approach is faster than the single scale because tracing the centerline from the segmentation is a time consuming task and tracing the same stack six times (for the OCCEN algorithm) takes a large amount of time. The average size of the stacks is $512 \times 512 \times 96$.

NP - DIADEM dataset: Table 2 summarizes the performance result for each algorithm on the NP dataset. The scales used for our segmentation approach are $\sigma_r = \{1.5, 1.75, 2.0, \ldots, 4.0\}$. The computational time to trace the 30 stacks was 1h 3m where the average size of the stacks was $512 \times 512 \times 96$. The algorithm of [12] was tested using six scales. Then, six traces were returned for each 3D image stack. The trace with the highest value of Precision, Recall and MES was selected to compute Table 2. Neurostudio achieves the best average performance for the metric Precision, indicating that Neurostudio produces fewer incorrect branches. However, our approach has better results for the metrics Recall and MES, thus indicating that the centerline traced from our algorithm is more similar to the ground truth than the methods of Neurostudio, APP2, Neutube, [3] and [4]. Our results are identical to our previous approach [12], indicating that both algorithms perform equally well. However, an advantage over our previous approach is that given the scales for segmentation, our algorithm generates only one reconstruction (tracing of the neuron) since all the scales are incorporated in our multi-scale framework, while [12] produces one reconstruction for each scale. Hence, the method of [12] requires visual inspection of the traced reconstruction for each scale and selection of the reconstruction with the best result, which is highly subjective. Specifically, [12] has an average Precision, Recall and MES of $\{0.95, 0.95, 0.89\}$, $\{0.96, 0.95, 0.90\}$, and $\{0.97, 0.92, 0.89\}$ for $\sigma = 1.5, 2$, and 2.5, respectively. Our multi-scale framework outperforms any of these results.

OP - DIADEM dataset: Table 3 summarizes the performance results of each algorithm on the OP dataset. The scales used for our segmentation approach are $\sigma_r = \{1.0, 1.2, 1.4, \ldots, 2.0\}$. The computational time to trace the 8 stacks was $8min$ $26s$ where the average size of the stacks was $512 \times 512 \times 77$. Our method achieves the best average performance for the metrics Recall and MES, hence our approach is able to detect more parts of the dendrites and creates fewer erroneous branches. Hence, our method produces more accurate tracings than state-of-the-art neuron reconstruction methods. The results from our method and [4] are very similar, indicating that both algorithms perform equally well. This is mainly because there is not large radii variation in the dendrites of this dataset, hence the segmentation method of [4] produces good segmentation results and therefore tracings. However, [4] has difficulty segmenting dendrites where there are large radii variations such as the NP dataset, as can be seen from Table 2. The quantitative results show that our algorithm clearly outperforms our previous approach since our previous approach produces incomplete tracings due to gaps in the 3D image stack. Our method is able to correct this error and then produce complete tracings.

6. Discussion

We have presented a novel, one-class classification approach to automatically segment the neurons from a 3D image stack that uses a multi-scale approach to detect a training set consisting of samples belonging to the background. Then, it learns a decision function for each scale and dendrites are detected as outliers (points that are rejected as background by the decision function). Our approach allows segmentation of thin and low contrast dendrites that are usually difficult to segment. In addition, the morphology of the dendrite is traced from the segmentation using a state-of-the-art centerline algorithm [4].

Our approach was evaluated in all the datasets from the publicly available DIADEM dataset. The qualitative and quantita-

Table 1: Performance evaluation NP dataset - single scale against multi-scale (A1-A6 [12] $\sigma = 1.5, 2.0, 2.5, 3.0, 3.5$, and 4.0, respectively and D: Our method.)

	Precision							Recall							MES						
Stack\method	A1	A2	A3	A4	A5	A6	D	A1	A2	A3	A4	A5	A6	D	A1	A2	A3	A4	A5	A6	D
Average	0.95	0.95	0.97	**0.98**	**0.98**	**0.98**	**0.98**	**0.99**	0.98	0.97	0.94	0.90	0.84	0.98	0.91	0.92	0.94	0.93	0.89	0.85	**0.96**

Table 2: Performance evaluation on 30 volumes from the NP dataset (A: [22], B: [9], C: [23] ,D: [3], E: [4], F: [12], G: Our method.)

	Precision							Recall							MES						
Stack\method	A	B	C	D	E	F	G	A	B	C	D	E	F	G	A	B	C	D	E	F	G
Average	**1.00**	0.99	0.98	0.98	0.97	0.98	0.98	0.89	0.88	0.93	0.95	0.92	**0.98**	**0.98**	0.93	0.86	0.92	0.94	0.90	**0.96**	**0.96**

Table 3: Performance evaluation on the OP dataset (A: [22], B: [9], C: [23], D: [3], E: [4], F: [12], G: Our method.)

	Precision							Recall							MES						
Stack\method	A	B	C	D	E	F	G	A	B	C	D	E	F	G	A	B	C	D	E	F	G
1	**0.99**	0.97	0.96	**0.99**	0.98	**0.99**	**0.99**	0.95	0.94	0.96	0.94	0.96	0.96	**0.97**	0.94	0.92	0.92	0.93	0.94	0.94	**0.96**
3	0.95	0.95	0.94	**0.98**	0.96	0.96	0.96	0.89	0.87	0.97	0.74	**0.98**	0.92	0.97	0.86	0.84	0.90	0.74	**0.94**	0.88	0.93
4	0.93	0.85	**0.94**	0.92	0.92	0.87	**0.94**	0.86	0.85	0.87	0.88	0.90	0.92	**0.93**	0.81	0.74	0.84	0.82	0.83	0.80	**0.87**
5	**0.98**	0.95	0.96	0.97	0.97	0.93	0.93	0.92	0.91	0.93	0.74	**0.97**	0.84	0.95	0.91	0.87	0.90	0.71	**0.94**	0.78	0.88
6	0.98	**0.99**	0.98	**0.99**	0.95	0.98	0.98	0.95	0.91	0.91	0.95	0.96	**0.98**	0.97	0.94	0.90	0.90	0.94	0.91	**0.96**	0.95
7	0.99	0.97	0.99	**1.00**	0.97	**1.00**	0.99	**0.98**	0.93	0.93	0.94	0.97	0.95	**0.98**	0.98	0.91	0.93	0.94	0.93	0.94	**0.97**
8	0.99	0.97	0.99	**1.00**	**1.00**	0.98	**1.00**	0.96	0.93	0.94	0.98	0.96	**0.99**	0.97	0.96	0.91	0.93	**0.97**	0.96	**0.97**	**0.97**
9	0.96	0.93	0.93	0.95	0.92	0.91	**0.97**	0.86	0.74	0.72	0.82	0.86	**0.91**	0.90	0.84	0.72	0.69	0.78	0.80	0.81	**0.87**
Average	**0.97**	0.95	0.96	**0.97**	0.96	0.95	**0.97**	0.92	0.88	0.91	0.87	0.95	0.93	**0.96**	0.90	0.85	0.88	0.85	0.91	0.89	**0.93**

tive results demonstrate that our algorithm is able to correctly trace and label neurons. In addition, the quantitative results indicate that our approach results in more accurate segmentation and centerline tracing of neurons compared with state-of-the-art neuron tracing algorithms. We demonstrated that a multi-scale approach to computing the Laplacian can be incorporated with a one-class classification to segment neurons. The main advantages of our approach are: (i) it works for a wide range of acquisition modalities without requiring tuning of the parameter settings; (ii) it can detect thin dendrites as in the OP dataset and thick dendrites as in the NP dataset; and (iii) it requires little user intervention (the user only has to set the scales for segmentation).

The ORION3 software and documentation can be requested at http://cbl.uh.edu/ORION/research/software (no fee license). Our software is written in Matlab R2015 and ITK. It is developed for Windows users; hence some routines have to be compiled for other OS.

Acknowledgement

P. Hernandez-Herrera is supported by a CONACYT graduate scholarship, M. Papadakis was supported in part by NSF-DMS 1320910; I.A. Kakadiaris was supported in part by NSF-DMS 1320910 and by the Hugh Roy and Lillie Cranz Cullen Endowment Fund.

Appendix A Proof of Proposition 1

The Laplacian filter $F^L_{n,\sigma}$ is defined in $\mathbb{R}^3$. Since the linear filter induced by $F^L_{n,\sigma}$ is a linear operator acting on $L^2(\mathbb{R}^3)$, the operator norm of this operator is equal to $||\widehat{F}^L_{n,\sigma}||_\infty$. Since $\widehat{F}^L_{n,\sigma}$ is radial, it is enough to find the maximum absolute value of $\widehat{F}^L_{n,\sigma}$ on any of the positive half-axes. So,

$$\begin{aligned} ||\widehat{F}^L_{n,\sigma}||_\infty &= \sup\{|\widehat{F}^L_{n,\sigma}(\xi)| : \xi \geq 0\} \\ &= \sup\{-\xi^2 \widehat{F}_L(c^2_{n,\sigma}\xi^2) : \xi \geq 0\}. \end{aligned}$$

To find this maximum value, we need to find the critical points of $\widehat{F}^L_{n,\sigma}$. We have,

$$\begin{aligned} \frac{d\widehat{F}^L_{n,\sigma}(\xi)}{d\xi} &= \frac{-2\xi}{e^{c^2_{n,\sigma}\xi^2}} \left(\frac{(c^2_{n,\sigma}\xi^2)^{n+1}}{n!} - p_n(c^2_{n,\sigma}\xi^2) \right) \\ &= 0. \end{aligned}$$

Hence, $\xi = 0$ is the obvious critical point, where $|\widehat{F}^L_{n,\sigma}|$ takes its minimum value. The remaining critical points given by

$$\frac{(c^2_{n,\sigma}\xi^2)^{n+1}}{n!} - p_n(c^2_{n,\sigma}\xi^2) = 0 \tag{16}$$

are the interesting ones, since at one of them $|\widehat{F}^L_{n,\sigma}|$ attains its maximum value, because $|\widehat{F}^L_{n,\sigma}|$ is differentiable. Next, set $\omega = c^2_{n,\sigma}\xi^2$. Since $\omega(\xi) = c^2_{n,\sigma}\xi^2$ and $\xi > 0$, the function ω is increasing and its range covers all of $\mathbb{R}^+$, we conclude that the non-zero critical points of $|\widehat{F}^L_{n,\sigma}|$ are determined by the roots of the polynomial equation

$$\frac{\omega^{n+1}}{n!} - p_n(\omega) = 0. \tag{17}$$

Let r_1 be the root of the polynomial in the right-hand side of Eq. (16) corresponding to the maximum of $|\widehat{F}^L_{n,\sigma}|$. Consequently, $\sqrt{r_1/c^2_{n,\sigma}}$, is a value of ξ at which $|\widehat{F}^L_{n,\sigma}|$ attains its maximum, which is equal to

$$\begin{aligned}\widehat{F}^L_{n,\sigma}\left(\sqrt{\frac{r_1}{c^2_{n,\sigma}}}\right) &= \left(\sqrt{\frac{r_1}{c^2_{n,\sigma}}}\right)^2 p_n\left(c^2_{n,\sigma}\left(\sqrt{\frac{r_1}{c^2_{n,\sigma}}}\right)^2\right) e^{-c^2_{n,\sigma}\left(\sqrt{\frac{r_1}{c^2_{n,\sigma}}}\right)^2} \\ &= \frac{r_1}{c^2_{n,\sigma}}\left(p_n(r_1)e^{-r_1}\right) \\ &= \frac{(2n+1)}{2\sigma^2 K^2}\left(r_1 p_n(r_1)e^{-r_1}\right) \\ &= \frac{1}{\sigma^2}\left(\frac{(2n+1)r_1 p_n(r_1)e^{-r_1}}{2K^2}\right) \\ &= \frac{1}{\sigma^2}C,\end{aligned} \tag{18}$$

where $C := \frac{(2n+1)r_1 p_n(r_1)\exp^{-r_1}}{2K^2}$ is a fixed constant that depends only on n, since r_1 only depends on the degree of the polynomial.

7. References

[1] T. G. Chowdhury, N. C. Barbarich-Marsteller, T. E. Chan, C. Aoki, Activity-based anorexia has differential effects on apical dendritic branching in dorsal and ventral hippocampal CA1, Brain Structure and Function 219 (6) (2014) 1935–1945.

[2] G. Gonzalez, F. Aguet, F. Fleuret, M. Unser, P. Fua, Steerable features for statistical 3D dendrite detection, in: Proc. Medical Image Computing and Computer-Assisted Intervention, London, UK, 2009, pp. 625–632.

[3] D. Jimenez, D. Labate, I. A. Kakadiaris, M. Papadakis, Improved automatic centerline tracing for dendritic and axonal structures, Neuroinformatics 13 (2) (2015) 227–244. `doi:10.1007/s12021-014-9256-z`.

[4] A. Santamaria-Pang, P. Hernandez-Herrera, P. M., S. P., I. A. Kakadiaris, Automatic morphological reconstruction of neurons from multiphoton and confocal microscopy images using 3D tubular models, Neuroinformatics 13 (3) (2015) 297–320.

[5] K. Brown, G. Barrionuevo, A. Canty, V. Paola, J. Hirsch, G. Jefferis, J. Lu, M. Snippe, I. Sugihara, G. Ascoli, The DIADEM data sets: representative light microscopy images of neuronal morphology to advance automation of digital reconstructions, Neuroinformatics 9 (2-3) (2011) 143–157.

[6] F. Janoos, K. Mosaliganti, X. Xu, R. Machiraju, K. Huang, S. T. Wong, Robust 3D reconstruction and identification of dendritic spines from optical microscopy imaging, Medical Image Analysis 13 (1) (2009) 167–179.

[7] P. Chothani, V. Mehta, A. Stepanyants, Automated tracing of neurites from light microscopy stacks of images, Neuroinformatics 9 (2-3) (2011) 263–278.

[8] J. Xie, T. Zhao, T. Lee, E. Myers, H. Peng, Anisotropic path searching for automatic neuron reconstruction, Medical Image Analysis 15 (5) (2011) 680–689.

[9] H. Xiao, H. Peng, APP2: automatic tracing of 3D neuron morphology based on hierarchical pruning of a gray-weighted image distance-tree, Bioinformatics 29 (11) (2013) 1448–1454.

[10] E. Meijering, Neuron tracing in perspective, Cytometry Part A 77A (7) (2010) 693–704. `doi:10.1002/cyto.a.20895`.

[11] D. Donohue, G. Ascoli, Automated Reconstruction of Neuronal Morphology: An Overview, Brain Research Reviews 67 (2011) 94–102.

[12] P. Hernandez-Herrera, M. Papadakis, I. A. Kakadiaris, Segmentation of neurons based on one-class classification, in: Proc. IEEE International Symposium in Biomedical Imaging: From Nano to Macro, Beijing, China, 2014, pp. 1316–1319.

[13] T. Lindeberg, Edge detection and ridge detection with automatic scale selection, International Journal of Computer Vision 30 (2) (1998) 77–116.

[14] D. K. Hoffman, N. Nayar, O. A. Sharafeddin, D. J. Kouri, Analytic banded approximation for the discretized free propagator, The Journal of Physical Chemistry 95 (21) (1991) 8299–8305.

[15] B. Bodmann, D. Hoffman, D. Kouri, M. Papadakis, Hermite distributed approximating functionals as almost-ideal low-pass filters, Sampling Theory in Image and Signal Processing 7 (2007) 15–38.

[16] B. Ozcan, D. Labate, D. Jiménez, M. Papadakis, Directional and non-directional representations for the characterization of neuronal morphology, Proc. SPIE 8858 (2013) 1–11.

[17] A. Frangi, W. Niessen, K. Vincken, M. Viergever, Multiscale vessel enhancement filtering, in: Proc. Medical Image Computing and Computer Assisted Intervention, Vol. 1496, Cambridge, MA, 1998, pp. 130–137.

[18] R. Cannon, D. Turner, G. Pyapali, H. Wheal, An on-line archive of reconstructed hippocampal neurons, Journal of Neuroscience Methods 84 (1) (1998) 49–54.

[19] H. Peng, M. Hawrylycz, J. Roskams, S. Hill, N. Spruston, E. Meijering, G. A. Ascoli, Bigneuron: large-scale 3D neuron reconstruction from optical microscopy images, Neuron 87 (2) (2015) 252–256.

[20] H. Peng, E. Meijering, G. A. Ascoli, From diadem to bigneuron, Neuroinformatics (2015) 1–2.

[21] Bigneuron. Http://bigneuron.org [online] (Nov. 2015).

[22] S. Wearne, A. Rodriguez, D. Ehlenberger, A. Rocher, S. Henderson, P. Hof, New techniques for imaging, digitization and analysis of three-dimensional neural morphology on multiple scales, Neuroscience 136 (3) (2005) 661–680. `doi:10.1016/j.neuroscience.2005.05.053`.

[23] L. Feng, T. Zhao, J. Kim, neuTube 1.0: A new design for efficient neuron reconstruction software based on the SWC Format, eNeuro 2 (1) (2015) 1–10.