Neuroinform (2016) 14:201–219
DOI 10.1007/s12021-015-9287-0

ORIGINAL ARTICLE

Fuzzy-Logic Based Detection and Characterization of Junctions and Terminations in Fluorescence Microscopy Images of Neurons

Miroslav Radojević[1] · Ihor Smal[1] · Erik Meijering[1]

Published online: 23 December 2015

Abstract Digital reconstruction of neuronal cell morphology is an important step toward understanding the functionality of neuronal networks. Neurons are tree-like structures whose description depends critically on the junctions and terminations, collectively called critical points, making the correct localization and identification of these points a crucial task in the reconstruction process. Here we present a fully automatic method for the integrated detection and characterization of both types of critical points in fluorescence microscopy images of neurons. In view of the majority of our current studies, which are based on cultured neurons, we describe and evaluate the method for application to two-dimensional (2D) images. The method relies on directional filtering and angular profile analysis to extract essential features about the main streamlines at any location in an image, and employs fuzzy logic with carefully designed rules to reason about the feature values in order to make well-informed decisions about the presence of a critical point and its type. Experiments on simulated as well as real images of neurons demonstrate the detection performance of our method. A comparison with the output of two existing neuron reconstruction methods reveals that our method achieves substantially higher detection rates and could provide beneficial information to the reconstruction process.

Keywords Neuron reconstruction · Junction detection · Bifurcation detection · Termination detection · Fuzzy logic · Fluorescence microscopy · Image analysis

✉ Miroslav Radojević
m.radojevic@erasmusmc.nl

Erik Meijering
h.meijering@erasmusmc.nl

[1] Biomedical Imaging Group Rotterdam, Departments of Medical Informatics and Radiology, Erasmus University Medical Center, Rotterdam, the Netherlands

Introduction

The complexity and functionality of the brain depend critically on the morphology and related interconnectivity of its neuronal cells (Kandel et al. 2000; Ascoli 2002; Donohue and Ascoli 2008). To understand how a healthy brain processes information and how this capacity is negatively affected by psychiatric and neurodegenerative diseases (Anderton et al. 1998; Lin and Koleske 2010; Šišková et al. 2014) it is therefore very important to study neuronal cell morphology. Advanced microscopy imaging techniques allow high-sensitivity visualization of individual neurons and produce vast amounts of image data, which are shifting the bottleneck in neuroscience from the imaging to the data processing (Svoboda 2011; Peng et al. 2011; Senft 2011; Halavi et al. 2012) and call for a high level of automation. The first processing step toward high-throughput quantitative morphological analysis of neurons is their digital reconstruction from the image data. Many methods have been developed for this in the past decades (Meijering 2010; Donohue and Ascoli 2011) but the consensus of recent studies is that there is still much room for improvement in both accuracy and computational efficiency (Liu 2011; Svoboda 2011).

A key aspect of any neuron reconstruction method is the detection and localization of terminations and junctions of the dendritic (and axonal) tree, collectively called

"critical points" in this paper (Fig. 1), which ultimately determine the topology and faithfulness of the resulting digital representation. Roughly there are two ways to extract these critical points in neuron reconstruction (Al-Kofahi et al. 2008; Meijering 2010; Basu et al. 2013). The most often used approach is to start with segmentation or tracing of the elongated image structures and then to infer the critical points, either afterwards or along the way, by searching for attachments and endings in the resulting subsets (Dima et al. 2002, Xiong et al. 2006, Narro et al. 2007, Vasilkoski and Stepanyants 2009, Bas and Erdogmus 2011, Chothani et al. 2011, Dehmelt et al. 2011, Ho et al. 2011, Choromanska et al. 2012, Xiao and Peng 2013). This approach depends critically on the accuracy of the initial segmentation or tracing procedure, which usually is not designed to reliably capture critical points in the first place and thus often produces very fragmented results, requiring manual postprocessing to fix issues (Peng et al. 2011; Luisi et al. 2011; Dercksen et al. 2014). The reverse approach is to first identify critical points in the images and then to use these as seed points for tracing the elongated image structures. Critical points can be obtained either by manual pinpointing, as in semiautomatic tracing methods (Meijering et al. 2004; Schmitt et al. 2004; Narro et al. 2007; Lu et al. 2009; Peng et al. 2010; Longair et al. 2011), or by fully automatic detection using sophisticated image filtering and pattern recognition methods (discussed in the next section). The latter approach has barely been explored for neuron reconstruction, but if reliable detectors can be designed, they provide highly valuable information to the reconstruction process.

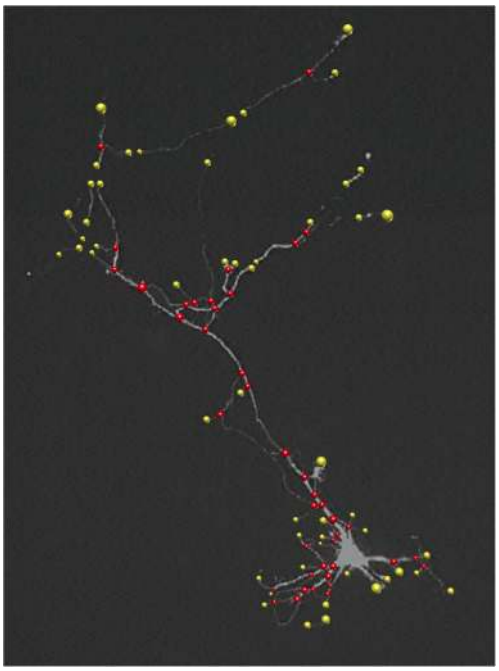

Fig. 1 Fluorescence microscopy image of a neuron with manually indicated junctions (*red circles*) and terminations (*yellow circles*). The radius of each annotated critical-point region reflects the size of the underlying image structure

Here we present a novel method – which we coin Neuron Pinpointer (NP) – for fully automatic detection and characterisation of critical points in fluorescence microscopy images of neurons. We describe and evaluate the method for studies where single (cultured) neurons are imaged in 2D although all aspects of the method can in principle be extended to 3D. The method may also be useful for reconstruction approaches based on 2D projections (Zhou et al. 2015). For computational efficiency the method starts with an initial data reduction step, based on local variation analysis, by which obvious background image regions are excluded. In the remaining set of foreground regions the method then explores the local neighborhood of each image pixel and calculates the response to a set of directional filters. Next, an iterative optimization scheme is used for robust peak selection in the resulting angular profile, and a set of corresponding features relevant for the detection task is computed. The feature set is then further processed to make a nonlinear decision on the presence of a critical point and its type (termination or junction) at each foreground image pixel. To conveniently deal with ambiguity and uncertainty in the data, the decision-making is carried out by a fuzzy-logic rule-based system using predefined rules specifically designed for this task. The presented work aims to facilitate the task of automatic neuron reconstruction by contributing a general scheme for extracting critical points that can serve as useful input for any tracing algorithm.

This paper is a considerably extended version of our recent conference report (Radojević et al. 2014). We have modified the filtering algorithms and fuzzy-logic rules so as to be able to detect both junction and termination points. In addition we here present the full details of our method and an extensive evaluation based on both manually annotated real neuron images and computer generated neuron images. To obtain the latter we here propose a new computational approach based on publicly available expert manual tracings. We start with a brief overview of related work on critical-point detection ("Related Work") and then present the underlying concepts ("Proposed Method"), implementational details ("Implementational Details"), and experimental evaluation ("Experimental Results") of our method, followed by a summary of the conclusions that can be derived from the results ("Conclusions").

Related Work

Detecting topologically critical points in images has been a long-standing problem in many areas of computer vision. Although an in-depth review of the problem and proposed solutions is outside the scope of this paper, we provide a brief discussion in order to put our work into context.

Examples of previous work include the design of filters to find image locations where either three or more edges join ("junctions of edges") (Sinzinger 2008; Hansen and Neumann 2004; Laganiere and Elias 2004) or three or more lines join ("junctions of lines") (Yu et al. 1998; Deschênes and Ziou 2000). In biomedical applications, the predominant type of junction is the bifurcation, with occasional trifurcations, as seen in blood vessel trees, bronchial trees, gland ductal trees, and also in dendritic trees (Koene et al. 2009; Iber and Menshykau 2013). Hence, research in this area has focused on finding image locations where three (or more) elongated structures join (Tsai et al. 2004; Agam et al. 2005; Bevilacqua et al. 2005; Bhuiyan et al. 2007; Zhou et al. 2007; Aibinu et al. 2010; Calvo et al. 2011; Obara et al. 2012b; Su et al. 2012; Azzopardi and Petkov 2013).

A common approach to find bifurcation points is to infer them from an initial processing step that aims to segment the elongated structures. However, the way these structures are segmented may influence the subsequent critical-point inference. Popular image segmentation methods use intensity thresholding and/or morphological processing, in particular skeletonization (Hoover et al. 2000; Dima et al. 2002; He et al. 2003; Weaver et al. 2004; Pool et al. 2008; Bevilacqua et al. 2009; Leandro et al. 2009; Aibinu et al. 2010), but these typically produce very fragmented results. Popular methods to enhance elongated image structures prior to segmentation include Hessian based analysis (Frangi et al. 1998; Xiong et al. 2006; Zhang et al. 2007; Al-Kofahi et al. 2008; Yuan et al. 2009; Türetken et al. 2011; Myatt et al. 2012; Basu et al. 2013; Santamaría-Pang et al. 2015), Laplacean-of-Gaussian filters (Chothani et al. 2011), Gabor filters (Bhuiyan et al. 2007; Azzopardi and Petkov 2013), phase congruency analysis (Obara et al. 2012a), and curvelet based image filtering approaches (Narayanaswamy et al. 2011). However, being tailored to elongated structures, such filters often yield a less optimal response precisely at the bifurcation points, where the local image structure is more complex than a single ridge.

Several concepts have been proposed to explicitly detect bifurcation points in the images without relying on an initial segmentation of the axonal and dendritic trees. Examples include the usage of circular statistics of phase information (Obara et al. 2012b), steerable wavelet based local symmetry detection (Püspöki et al. 2013), or combining eigen analysis of the Hessian and correlation matrix (Su et al. 2012). The problem with existing methods is that they often focus on only one particular type of critical point (for example bifurcations but not terminations), or they use rather rigid geometrical models (for example assuming symmetry), while in practice there are many degrees of freedom (Michaelis and Sommer 1994). Image filtering methods for bifurcation detection have also been combined with supervised machine-learning based approaches such as support vector machines (Türetken et al. 2011), artificial neural networks (Bevilacqua et al. 2009), or with multiple classifiers using AdaBoost (Zhou et al. 2007), but these lack flexibility in that they require a training stage for each application.

Robust neuron tracing requires knowledge of not only the bifurcation points but also the termination points. Since each type of critical point may vary considerably in terms of geometry (orientation and diameter of the branches) and image intensity (often related to the branch diameter), designing or training a dedicated filter for each possible case is impractical, and a more integrated approach is highly desirable for both detection and characterization of the different types of critical points. To the best of our knowledge, no generic methods currently exist for critical-point detection in neuron images. The method proposed in this paper aims to fill this gap and to complement exploratory neuron reconstruction algorithms that initialize on a set of seed points.

Proposed Method

Our proposed method for detection and characterization of critical points consists of three steps: feature extraction ("Feature Extraction"), fuzzy-logic based mapping ("Fuzzy-Logic Based Mapping"), and, finally, critical-point determination ("Critical-Point Determination"). Here we describe each step in detail.

Feature Extraction

The aim of the feature extraction step is to compute a set of quantitative features of the local image structure at each pixel position that helps to discriminate between different types of critical points. Since the tree-like neuronal image structures typically cover only a small portion of the image, we avoid unnecessary computations by first performing a foreground selection step ("Foreground Selection"), which discards image locations that are very unlikely to contain neuronal structures and keeps only those regions that are worthy of further examination. Next, the angular pro-

Fig. 2 Example of foreground selection. The original image of 560 × 780 pixels is divided into background (*green transparent mask*) and foreground (*gray-scale regions without mask*) using $r_d = 8$ pixels and the 75$^{\text{th}}$ variation percentile as threshold. In this example, 25% of the total number of pixels is selected for further processing

file ("Angular Profile Analysis") of each foreground pixel is constructed, from which the quantitative features are computed.

Foreground Selection

To determine whether a pixel location (x, y) in a given image I should be considered as foreground or background, we analyze the local intensity variation $\rho(x, y)$ within a circular neighborhood of radius r_d centered at that location. To avoid making strong assumptions about the local intensity distribution we chose to use the difference between the 95$^{\text{th}}$ and the 5$^{\text{th}}$ percentile of the intensities within the neighborhood as the measure of variation:

$$\rho(x, y) = \mathcal{P}_{95}(\mathcal{I}_{xy}) - \mathcal{P}_{5}(\mathcal{I}_{xy}) \tag{1}$$

$$\mathcal{I}_{xy} = \left\{ I(m, n) \mid (m - x)^2 + (n - y)^2 \le r_d^2 \right\} \tag{2}$$

$$x, m \in [0, W - 1] \text{ and } y, n \in [0, H - 1] \tag{3}$$

where W and H denote, respectively, the width and the height of I in pixels. The value of ρ is relatively low within more or less homogeneous regions (background but also the soma) but relatively high in regions containing neuronal branches. Consequently, the histogram of ρ computed over the entire image contains two clusters (representing foreground and background pixels), which can be separated using simple percentile thresholding (Doyle 1962). The percentile should be chosen such that background pixels (true negatives) are removed as much as possible while at the same time the foreground pixels (true positives) are retained as much as possible (in practice this implies allowing for false positives). We found that in our applications a percentile of around 75 is a safe threshold (Fig. 2). Small gaps in the foreground region are closed by morphological dilation. The resulting set of foreground pixel locations is denoted by F. In our applications the parameter r_d is typically set to the diameter of the axonal and dendritic structures observed in the image.

Angular Profile Analysis

For each selected foreground location, a local angular profile is computed and analyzed. The key task here is to assess the presence and properties of any curvilinear image structures passing through the given location. To this end we correlate the image with a set of oriented kernels distributed evenly over a range of angles around that location (Radojević et al. 2014). The basic kernel used for this purpose is of size $D \times D$ pixels and has a constant profile in one direction and a Gaussian profile in the orthogonal direction (Fig. 3):

$$G(x, y) = \mathrm{e}^{-x^2/2\sigma_D^2} / S \tag{4}$$

where S is a normalization factor such that the sum of $G(x, y)$ over all kernel pixels is unity. We chose the Gaussian both because we observed that the cross-sectional profile of axons and dendrites in our applications is approximately Gaussian-like and because the Gaussian is a theoretically well-justified filter for regularization purposes. The parameters D and σ_D determine the size and shape of the

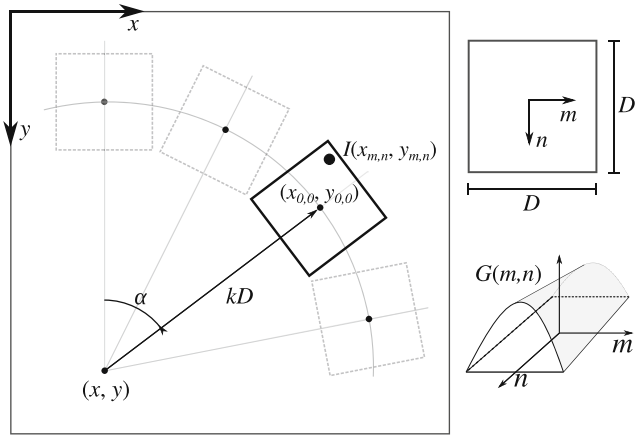

Fig. 3 Geometry involved in the computation of the angular profile. In effect, the value of $p(x, y, \alpha, k, D)$ is the correlation of the image $I(x, y)$ with the kernel $G(m, n)$ of size $D \times D$ pixels, after rotating the kernel patch over angle α and shifting it over kD with respect to (x, y)

kernel profile and should correspond to the expected branch diameter.

Using the kernel we compute the local angular profile at any pixel location (x, y) in the given image I as:

$$p(x, y, \alpha, k, D) = \sum_m \sum_n I(x_{m,n}, y_{m,n})\, G(m, n) \tag{5}$$

where the transformed image coordinates are obtained as:

$$\begin{bmatrix} x_{m,n} \\ y_{m,n} \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + kD \begin{bmatrix} \sin\alpha \\ -\cos\alpha \end{bmatrix} + \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} m \\ n \end{bmatrix} \tag{6}$$

and the summation is performed over all (m, n) for which the kernel is defined. That is, $p(x, y, \alpha, k, D)$ is the correlation of the image with the kernel patch rotated over angle α and shifted over a distance kD with respect to (x, y) in the direction corresponding to that angle (Fig. 3). In practice, p is calculated for a discrete set of angles, $\alpha_i = i/(2\pi N_\alpha)$, $i = 0, \ldots, N_\alpha - 1$, where N_α is automatically set such that the circle with radius kD is sampled with pixel resolution. The parameter k is typically set slightly larger than 0.5 so as to scan the neighborhood around the considered pixel (x, y). To obtain the image intensity at non-integer transformed locations $(x_{m,n}, y_{m,n})$, linear interpolation is used.

In contrast with previous works, which used differential kernels for directional filtering and profiling (Yu et al. 1998; Can et al. 1999; Zhang et al. 2007), we employ the matched kernel (4), which avoids noise amplification. Although applying a set of rotated kernels is computationally more demanding than Hessian or steerable filtering based methods, it provides more geometrical flexibility in matching the kernels with the structures of interest while retaining excellent directional sensitivity. In our framework, the computational burden is drastically reduced by the foreground selection step, and further reduction is possible since the filtering process is highly parallelizable.

After computing the angular profile we further process it in order to extract several features (Fig. 4) relevant for critical-point detection and characterization:

Peaks At each foreground pixel location we first determine how many and in which direction line-like image structures pass through it. This is done by finding the local maxima ("peaks") in the angular profile at that location. Since the oriented kernels act as low-pass filters, the profile is sufficiently smooth to extract the peaks reliably using the iterative line searching algorithm (Flannery et al. 1992). The found peaks correspond to angles $\hat{\alpha}_i, i = 1, \ldots, N_{\hat{\alpha}}$, in which directions the image intensities are the highest. Here $N_{\hat{\alpha}} \leq 4$ to accommodate terminations, normal body points, and junctions (bifurcations and crossovers).

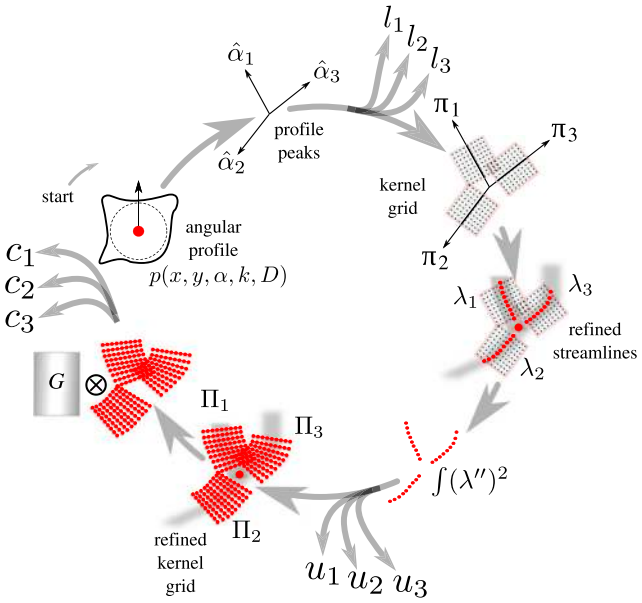

Fig. 4 Flowchart of the feature extraction scheme. The example showcases a bifurcation but the same scheme is used also for terminations. The scheme, which starts with the angular profile $p(x, y, \alpha, k, D)$ and is executed clockwise, is applied to each pixel in the selected foreground regions and results in the set of features l_i, u_i, and c_i, where i indexes the streamlines. See main text for details

Likelihood For each $\hat{\alpha}_i$ we calculate a likelihood $l_i \in [0, 1]$ from the angular profile according to:

$$l_i = \frac{p(x, y, \hat{\alpha}_i, k, D) - p_{\min}}{p_{\max} - p_{\min}} \tag{7}$$

where $p_{\min}$ and $p_{\max}$ denote, respectively, the minimum and maximum of $p(x, y, \alpha, k, D)$ over α.

Energy Next we consider the local grid $\pi_i(x, y, \hat{\alpha}_i, k, D)$ for each $\hat{\alpha}_i$ (Fig. 4), consisting of the transformed coordinates $(x_{m,n}, y_{m,n})$ corresponding to $\alpha = \hat{\alpha}_i$ (6), and we extract a refined centerline point set λ_i (or "streamline") on this grid by finding for each n the local maximum over m:

$$\lambda_i = \left\{(x_{\hat{m}_n,n}, y_{\hat{m}_n,n})\right\}_{n \in [-D/2, D/2]} \tag{8}$$

$$\hat{m}_n = \underset{m \in [-D/2, D/2]}{\arg\max} I(x_{m,n}, y_{m,n}) \tag{9}$$

We quantify how much the streamline deviates from a straight line by estimating its bending energy $u_i \geq 0$ as:

$$u_i = \frac{1}{\Delta m} \sum_n \left(\hat{m}_{n-1} - 2\hat{m}_n + \hat{m}_{n+1}\right)^2 \tag{10}$$

where Δm is the pixel spacing in the direction of m and the summation extends over all n for which the summand can be evaluated. This calculation is a discrete approximation of the integral squared second-order derivative of the centerline function if it were continuously defined.

Correlation Given a streamline λ_i we estimate the orthogonal direction at each point in the set by averaging the orthogonal directions of the two neighboring streamline segments corresponding to that point (that is, from the point to the next point, and from the point to the previous point). Using these direction estimates we sample a refined local grid $\Pi_i(x, y, \hat{\alpha}_i, k, D)$, consisting of image coordinates $(\tilde{x}_{m,n}, \tilde{y}_{m,n})$ relative to the streamline (Fig. 4), and compute a normalized cross-correlation (Lewis 1995) score $c_i \in [-1, 1]$ as:

$$c_i = \frac{\sum_m \sum_n \left[I(\tilde{x}_{m,n}, \tilde{y}_{m,n}) - \bar{I}\right]\left[G(m,n) - \bar{G}\right]}{\sqrt{\sum_m \sum_n \left[I(\tilde{x}_{m,n}, \tilde{y}_{m,n}) - \bar{I}\right]^2 \sum_m \sum_n \left[G(m,n) - \bar{G}\right]^2}} \quad (11)$$

where, similar to the angular profile calculation (5), the summations extend over all (m, n) for which the kernel is defined, and $\bar{I}$ and $\bar{G}$ denote the mean of the image intensities and of the kernel values, respectively. Effectively c_i quantifies the degree to which the template G matches a straightened version of the streamline. To cover a range of possible scales (radii of the underlying image structures), we take the largest score of a set of templates with standard deviations of the Gaussian profile model (Su et al. 2012) covering $\left\{1, \ldots, \lfloor \frac{D}{2} \rfloor\right\}$ set of values measured in pixels.

Fuzzy-Logic Based Mapping

The feature values extracted at each foreground image location subsequently need to be processed in order to assess the presence of a critical point and its type. Recognizing that in practice everything is "a matter of degree" (Zadeh 1975), and allowing for nonlinear input-output mappings, we chose to use fuzzy logic for this purpose. Fuzzy logic has been successfully used in many areas of engineering (Mendel 1995) but to the best of our knowlege has not been explored for neuron critical-point analysis. We briefly describe the basics of fuzzy logic ("Basics of Fuzzy Logic") and then present our specific fuzzy-logic system for calculating critical-point maps of neuron images ("Termination and Junction Mapping").

Basics of Fuzzy Logic

In a fuzzy-logic system (Fig. 5), numerical inputs are first expressed in linguistic terms (the fuzzification step), and are then processed based on predefined rules to produce linguistic outputs (the inference step), which are finally turned back into numerical values (the defuzzification step).

Fuzzification Given an input scalar value $s \in \mathbb{R}$, the fuzzification step results in a vector $\tilde{\mathbf{s}}$ whose elements express the degree of membership of s to input fuzzy sets, each corresponding to a linguistic term describing s. A fuzzy set is defined by a membership function $\mu : \mathbb{R} \to [0, 1]$ quantifying the degree to which s can be described by the corresponding linguistic term. Commonly used membership functions are trapezoidal, Gaussian, triangular, and piecewise linear (Mendel 1995). As an example, we may have linguistic terms LOW and HIGH, representing the subjective notions "low" and "high", respectively. The degrees to which "s is low" (which in this paper we will write as s = LOW) and "s is high" (s = HIGH) are given by membership values $\mu_{\text{LOW}}(s)$ and $\mu_{\text{HIGH}}(s)$, respectively. The output of the fuzzification step thus becomes $\tilde{\mathbf{s}} = [\mu_{\text{LOW}}(s), \mu_{\text{HIGH}}(s)]^T$.

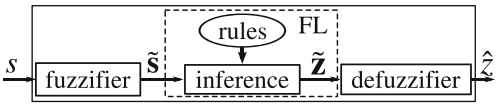

Fig. 5 Scheme of a single input/output fuzzy-logic (FL) system. A scalar input value s is converted to a vector $\tilde{\mathbf{s}}$ containing the memberships of s for each of the input fuzzy sets, resulting in a vector $\tilde{\mathbf{z}}$ containing the memberships of z for each of the output fuzzy sets

Inference The input fuzzy set memberships are processed by the inference engine to produce a fuzzy output based on rules expressing expert knowledge. The rules can be either explicitly defined or implicitly learned by some training process, and may express nonlinear input-output relationships and involve multiple inputs. In engineering applications, the rules are commonly given as IF-THEN statements about the input and output linguistic terms. For example, the output terms could be OFF, NONE, and ON, indicating whether a certain property of interest is "off", "none" (expressing ambiguity), or "on". A rule could then be:

$$\begin{aligned} R_i: \ & \text{IF } (s_1 = \text{HIGH}) \ \wedge \ (s_2 = \text{LOW}) \\ & \text{THEN } (z = \text{OFF}) \end{aligned} \quad (12)$$

where $z \in \mathbb{R}$ is the variable over the output range. This is not a binary logical statement, where the input and output conditions can be only true or false, but a fuzzy logical statement, where the conditions are expressed in terms of memberships, in this case $\mu_{\text{HIGH}}(s_1)$, $\mu_{\text{LOW}}(s_2)$, and $\mu_{\text{OFF}}(z)$. Input conditions are often combined using the operators $\wedge$ (denoting fuzzy intersection) or $\vee$ (denoting fuzzy union), which are commonly defined as, respectively, the minimum and maximum of the arguments (Mendel 1995). In our example, the IF-part of R_i (12) would result in the following intermediate value (degree of verity):

$$\upsilon_i = \min\{\mu_{\text{HIGH}}(s_1), \mu_{\text{LOW}}(s_2)\} \quad (13)$$

This value is then used to constrain the fuzzy set corresponding to the output linguistic term addressed by R_i, in this case OFF, resulting in the output fuzzy set:

$$\Upsilon_i(z) = \min\{\mu_{\text{OFF}}(z), \upsilon_i\} \quad (14)$$

In practice there may be many rules $R_i, i = 1, \ldots, N_R$, which are aggregated by the inference engine to produce a single output fuzzy set Υ. The common way to do this (Mendel 1995) is by means of a weighted fuzzy union:

$$\Upsilon(z) = \max\left\{w_1\Upsilon_1(z), \ldots, w_{N_R}\Upsilon_{N_R}(z)\right\} \tag{15}$$

Although it is possible to assign a different weight to each rule by setting $w_i \in [0, 1]$, in our applications this is not critical, and therefore we simply use $w_i = 1$ for all i.

Defuzzification In the final step of the fuzzy-logic system, the fuzzy output Υ is converted back to a scalar output value. Although there are many ways to do this, a common choice is to calculate the centroid (Mendel 1995):

$$\hat{z} = \frac{\int z\Upsilon(z)dz}{\int \Upsilon(z)dz} \tag{16}$$

With this value we can finally calculate the vector of output fuzzy set memberships: $\tilde{\mathbf{z}} = [\mu_{\text{OFF}}(\hat{z}), \mu_{\text{NONE}}(\hat{z}), \mu_{\text{ON}}(\hat{z})]^T$.

Termination and Junction Mapping

To determine the presence and type of critical point at any foreground image location, we use a cascade of two fuzzy-logic systems, representing two decision levels (Fig. 6). The first level takes as input vectors $\mathbf{s}_i = [l_i, u_i, c_i]$, $i = 1, \ldots, 4$, which contain the features for each of the streamlines extracted in the angular profile analysis step at the image location under consideration ("Angular Profile Analysis"). For each streamline (Fig. 7), the features are fuzzified (μ) and processed by the first fuzzy-logic module (FL_1), which determines the degree to which the streamline indeed represents a line-like image structure (ON), or not (OFF), or whether the image structure is ambiguous (NONE). In cases where less than four streamlines were found by the angular profile analysis step,

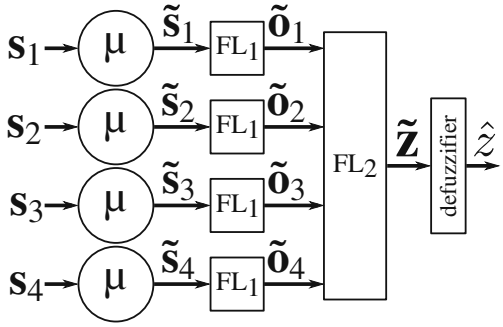

Fig. 6 Architecture of the proposed fuzzy-logic system for critical-point detection. A cascade of two fuzzy-logic modules (FL_1 and FL_2) is used, where the first determines the degree to which streamlines (up to four) are present at the image location under consideration, and based on this information the second determines the degree to which that location corresponds to the possible types of critical points

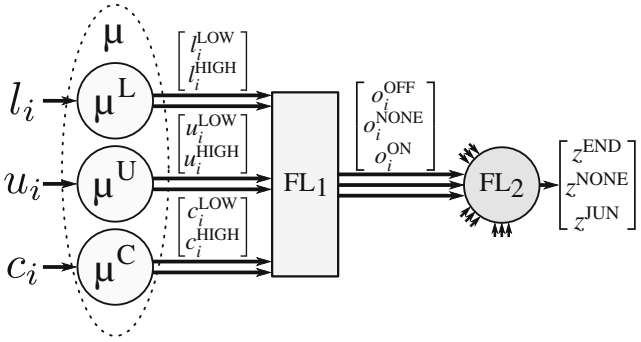

Fig. 7 Architecture of the proposed fuzzy-logic system for processing the information of one streamline. Input feature values are fuzzified into linguistic terms LOW and HIGH, which are translated by the first fuzzy-logic module (FL_1) into intermediate linguistic terms OFF, NONE, ON, which are finally translated by the second fuzzy-logic module (FL_2) into linguistic terms END, NONE, JUN

the feature vectors of the nonexisting streamlines are set to $\mathbf{0}$. The fuzzy output for all four streamlines together forms the input for the second decision level, where another fuzzy-logic module (FL_2) determines the degree to which the image location corresponds to a junction (JUN), or a termination (END), or neither of these (NONE).

The input streamline features, l_i, u_i, c_i, are expressed in linguistic terms LOW and HIGH using membership functions μ_{LOW} and μ_{HIGH} defined for each type of feature. In our application we use trapezoidal membership functions, each having two inflection points, such that μ_{LOW} and μ_{HIGH} are each other's complement (Fig. 8). For example, the degrees to which $l_i = \text{LOW}$ and $l_i = \text{HIGH}$, are given by $l_i^{\text{LOW}} = \mu^L_{\text{LOW}}(l_i)$ and $l_i^{\text{HIGH}} = \mu^L_{\text{HIGH}}(l_i) = 1 - l_i^{\text{LOW}}$, respectively, and because of this complementarity we often simply write μ^L to refer to both membership functions (Fig. 7). Similarly, the membership degrees of u_i and c_i are given by μ^U and μ^C, respectively. Summarizing, we use the following notations and definitions for the fuzzification step:

$$\begin{aligned}
\mu^L\colon\ l_i &\rightarrow \tilde{\mathbf{l}}_i = \left[l_i^{\text{LOW}},\ l_i^{\text{HIGH}}\right]^T \\
\mu^U\colon\ u_i &\rightarrow \tilde{\mathbf{u}}_i = \left[u_i^{\text{LOW}},\ u_i^{\text{HIGH}}\right]^T \\
\mu^C\colon\ c_i &\rightarrow \tilde{\mathbf{c}}_i = \left[c_i^{\text{LOW}},\ c_i^{\text{HIGH}}\right]^T
\end{aligned} \tag{17}$$

and the lower and higher inflection points of μ^L are denoted by L_{LOW} and L_{HIGH}, and similarly U_{LOW} and U_{HIGH} for μ^U, and C_{LOW} and C_{HIGH} for μ^C (Fig. 8).

Taken together, the input to FL_1 is the matrix of memberships $\tilde{\mathbf{s}}_i = [\tilde{\mathbf{l}}_i, \tilde{\mathbf{u}}_i, \tilde{\mathbf{c}}_i]$, and the output is the vector $\tilde{\mathbf{o}}_i$ of memberships to the linguistic terms OFF, NONE, ON:

$$\text{FL}_1\colon\ \tilde{\mathbf{s}}_i \rightarrow \tilde{\mathbf{o}}_i = \left[o_i^{\text{OFF}},\ o_i^{\text{NONE}},\ o_i^{\text{ON}}\right]^T \tag{18}$$

To calculate these memberships we introduce scalar variable o, where $o = 0$ corresponds to OFF, $o = 1$ to NONE, and $o = 2$ to ON, and we define Gaussian membership

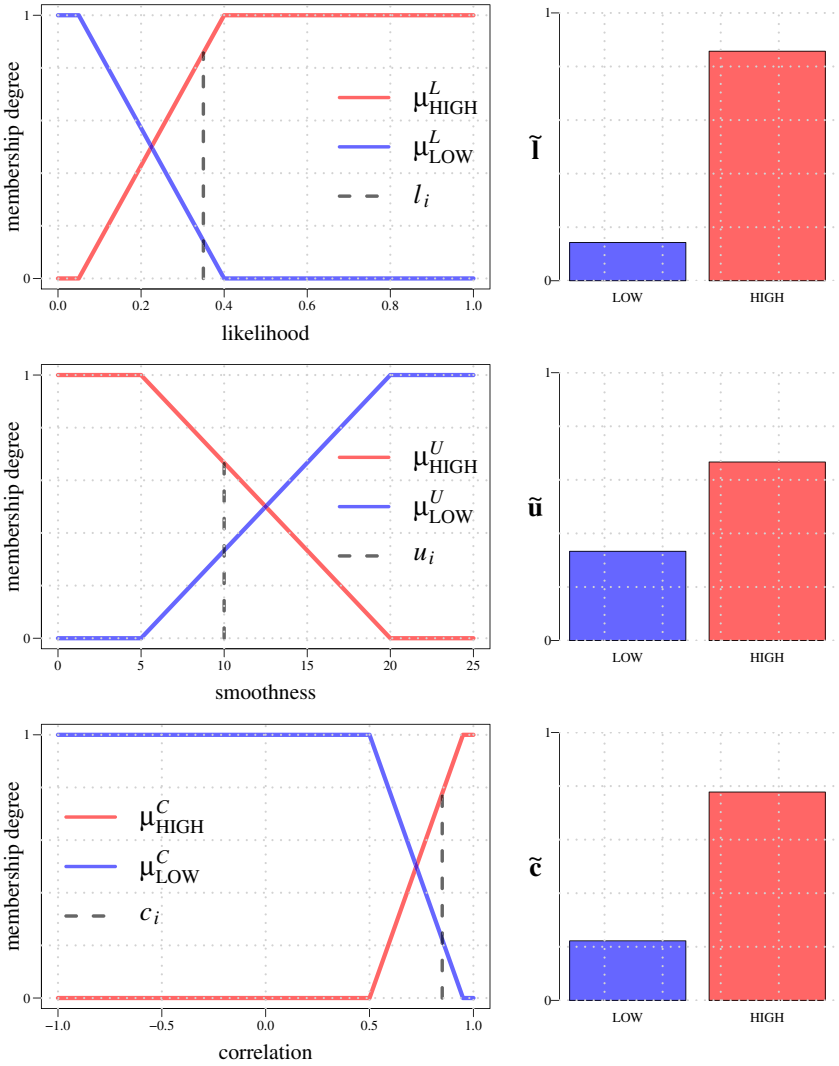

Fig. 8 Input membership functions used in the fuzzification step for FL_1. Example LOW and HIGH membership values are shown (*right column*) for input values (*dashed vertical lines in the plots on the left*) $l_i = 0.35$ (*top row*), $u_i = 10$ (*middle row*), and $c_i = 0.85$ (*bottom row*). The inflection points of the membership functions are, respectively, $L_{\text{LOW}} = 0.05$ and $L_{\text{HIGH}} = 0.4$ for μ^L, $U_{\text{HIGH}} = 5$ and $U_{\text{LOW}} = 20$ for μ^U, and $C_{\text{LOW}} = 0.5$ and $C_{\text{HIGH}} = 0.95$ for μ^C. Notice that features u_i (the centerline bending energies of the streamlines) are reinterpreted here to express the degree of smoothness (hence the inverted membership functions as compared to the other two)

functions μ^O_{OFF}, μ^O_{NONE} and μ^O_{ON}, centered around 0, 1, and 2, respectively (Fig. 9), and with fixed standard deviation 0.4 so that they sum to about 1 in the interval [0, 2]. The rules used by FL_1 to associate the input terms LOW and HIGH to the output terms OFF, NONE, and ON, are given in Table 1. They are based on the heuristic assumption that a line-like image structure exists (ON) if the evidence represented by all three features support it (HIGH). By contrast, if the likelihood is LOW and at least one other feature is also LOW, this indicates that no such structure exists (OFF). In all remaining cases, some structure may exist, but it is not line-like (NONE). As an example, rule R_8 (Table 1) is given by:

$$R_8: \quad \text{IF} \quad (l = \text{HIGH}) \ \wedge \ (u = \text{HIGH}) \ \wedge \ (c = \text{HIGH}) \quad \text{THEN} \ (o = \text{ON}) \tag{19}$$

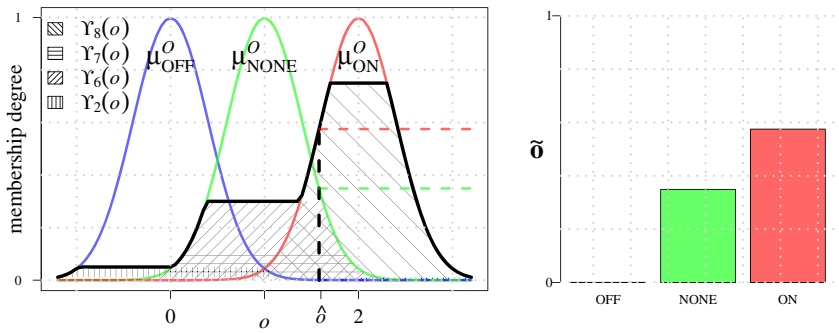

Fig. 9 Output membership functions used in module FL_1. Example output fuzzy sets Υ_i corresponding to rules R_i from Table 1 are shown as the textured areas. Value $\hat{o}$ (*left panel*) represents the centroid of the aggregated output fuzzy sets. The resulting output membership values (*right panel*) serve as input for module FL_2

Table 1 The set of rules employed by FL_1

R_i	l	u	c	o
1	LOW	LOW	LOW	OFF
2	LOW	LOW	HIGH	OFF
3	LOW	HIGH	LOW	OFF
4	LOW	HIGH	HIGH	NONE
5	HIGH	LOW	LOW	NONE
6	HIGH	LOW	HIGH	NONE
7	HIGH	HIGH	LOW	NONE
8	HIGH	HIGH	HIGH	ON

which results in the verity value:

$$\upsilon_8 = \min\left\{\mu^L_{\text{HIGH}}(l), \mu^U_{\text{HIGH}}(u), \mu^C_{\text{HIGH}}(c)\right\} \tag{20}$$

and the output fuzzy set:

$$\Upsilon_8(o) = \min\left\{\mu^O_{\text{ON}}(o), \upsilon_8\right\} \tag{21}$$

All the rules are resolved and combined as:

$$\Upsilon(o) = \max\{\Upsilon_1(o), \ldots, \Upsilon_8(o)\} \tag{22}$$

and centroid defuzzification then results in a scalar output value $\hat{o}$. This procedure is repeated for each streamline, yielding $\hat{o}_i, i = 1, \ldots, 4$, from which the output of each FL_1 (18) is calculated using the membership functions:

$$\tilde{\mathbf{o}}_i = \left[\mu^O_{\text{OFF}}(\hat{o}_i),\ \mu^O_{\text{NONE}}(\hat{o}_i),\ \mu^O_{\text{ON}}(\hat{o}_i)\right]^T \tag{23}$$

Moving on to the next level, the input to FL_2 is the matrix of memberships $\tilde{\mathbf{o}} = \left[\tilde{\mathbf{o}}_1, \tilde{\mathbf{o}}_2, \tilde{\mathbf{o}}_3, \tilde{\mathbf{o}}_4\right]$, and the output is the vector $\tilde{\mathbf{z}}$ of memberships to the linguistic terms END (termination), NONE (no critical point), JUN (junction):

$$\text{FL}_2\colon\ \tilde{\mathbf{o}} \to \tilde{\mathbf{z}} = \left[z^{\text{END}},\ z^{\text{NONE}},\ z^{\text{JUN}}\right]^T \tag{24}$$

To calculate these memberships we introduce scalar variable z, where $z = 1$ corresponds to END, $z = 2$ to NONE, and $z = 3$ to JUN, and we define corresponding Gaussian membership functions μ^Z_{END}, μ^Z_{NONE}, and μ^Z_{JUN}, centered around 1, 2, and 3, respectively, and with fixed standard deviation 0.4 as before (Fig. 10). The rules used by FL_2 to associate the input terms OFF, NONE, ON to the output terms END, NONE, JUN are given in Table 2. They are based on the heuristic assumption that there is a termination (END) if a single streamline is confirmed to correspond to a line-like image structure (ON) and the other three are confirmed to not correspond to such a structure (OFF). Conversely, if at least three are ON, there must be a junction at that location. Finally, if two are ON and two are OFF, or if at least two streamlines are ambiguous (NONE), we assume there is no critical point. Similar to FL_1, all the rules of FL_2 are evaluated and their results combined as:

$$\Upsilon(z) = \max\{\Upsilon_1(z), \ldots, \Upsilon_{22}(z)\} \tag{25}$$

which, after centroid defuzzification, results in a scalar output value $\hat{z}$, from which the output of FL_2 (24) is calculated using the membership functions:

$$\tilde{\mathbf{z}} = \left[\mu^Z_{\text{END}}(\hat{z}),\ \mu^Z_{\text{NONE}}(\hat{z}),\ \mu^Z_{\text{JUN}}(\hat{z})\right]^T \tag{26}$$

The proposed fuzzy-logic system is applied to each foreground pixel location $(x, y) \in F$ ("Foreground Selection") so that all memberships introduced above may be indexed

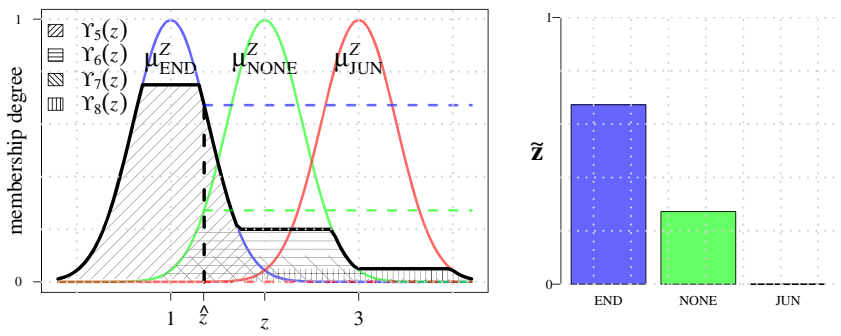

Fig. 10 Output membership functions used in module FL_2. Example output fuzzy sets Υ_i corresponding to rules R_i from Table 2 are shown as the textured areas. Value $\hat{z}$ (*left panel*) represents the centroid of the aggregated output fuzzy sets. The resulting output membership values (*right panel*) indicate the degree to which there may be a termination (END), junction (JUN), or neither of these (NONE) at the image pixel location under consideration

Table 2 The set of rules employed by FL_2. Empty entries indicate "don't care" (could be OFF, NONE, or ON)

R_i	o_1	o_2	o_3	o_4	z
1	OFF	OFF	OFF	OFF	NONE
2	OFF	OFF	OFF	ON	END
3	OFF	OFF	ON	OFF	END
4	OFF	OFF	ON	ON	NONE
5	OFF	ON	OFF	OFF	END
6	OFF	ON	OFF	ON	NONE
7	OFF	ON	ON	OFF	NONE
8	OFF	ON	ON	ON	JUN
9	ON	OFF	OFF	OFF	END
10	ON	OFF	OFF	ON	NONE
11	ON	OFF	ON	OFF	NONE
12	ON	OFF	ON	ON	JUN
13	ON	ON	OFF	OFF	NONE
14	ON	ON	OFF	ON	JUN
15	ON	ON	ON	OFF	JUN
16	ON	ON	ON	ON	JUN
17	NONE	NONE			NONE
18	NONE		NONE		NONE
19	NONE			NONE	NONE
20		NONE	NONE		NONE
21		NONE		NONE	NONE
22			NONE	NONE	NONE

by (x, y). Based on this we calculate the following two maps:

$$I_{\text{END}}(x, y) = \begin{cases} z^{\text{END}}(x, y) & \text{if } (x, y) \in F \\ 0 & \text{otherwise} \end{cases} \quad (27)$$

$$I_{\text{JUN}}(x, y) = \begin{cases} z^{\text{JUN}}(x, y) & \text{if } (x, y) \in F \\ 0 & \text{otherwise} \end{cases} \quad (28)$$

which indicate the degree to which any pixel (x, y) belongs to a termination or a junction, respectively.

Critical-Point Determination

The ultimate aim of our method is to provide a list of critical points in the neuron image, with each point fully characterized in terms of type, location, size, and main branch direction(s). Since each critical point of a neuronal tree typically covers multiple neighboring pixels in the image, giving rise to a high value at the corresponding pixels in the maps I_{END} and I_{JUN}, the final task is to segment the maps and to aggregate the information within each segmented region.

Due to noise, labeling imperfections, and structural ambiguities in the original image, the values of neighboring pixels in the maps may vary considerably, and direct thresholding usually does not give satisfactory results. To improve the robustness we first regularize the real-valued scores in the maps by means of local-average filtering with a radius of 3-5 pixels. Next, max-entropy based automatic thresholding (Kapur et al. 1985) is applied to segment the maps, as in contrast with many other thresholding methods we found it to perform well in separating the large but relatively flat (low information) background regions from the much smaller but more fluctuating (high information) regions of interest. The resulting binary images are further processed using a standard connected components algorithm (Sonka et al. 2007) to identify the critical-point regions.

Each critical region consists of a set of connected pixels $\mathbf{x}_p = (x_p, y_p)$, $p = 1, \ldots, N_p$, where N_p denotes the number of pixels in the region. From these, the representative critical-point location $\mathbf{x}_C = (x_C, y_C)$ is calculated as:

$$\mathbf{x}_C = \frac{1}{N_p} \sum_{p=1}^{N_p} \mathbf{x}_p \quad (29)$$

while the critical-point size is represented by the radius of the minimum circle surrounding the region:

$$r_C = \max_p \left\{ ||\mathbf{w}_p|| \right\} \quad (30)$$

where $\mathbf{w}_p = \mathbf{x}_p - \mathbf{x}_C$ (Fig. 11). To obtain regularized estimates of the main branch directions $\hat{\mathbf{v}}_i$ for the critical point, we aggregate the directions corresponding to the angular profile peaks $\hat{\alpha}_i$ ("Angular Profile Analysis") of all the $\mathbf{x}_p$ in the region as follows. For each $\mathbf{x}_p$ we have $N_{\hat{\alpha}} \leq 4$ angular profile peak direction vectors $\mathbf{a}_{p,i} =$

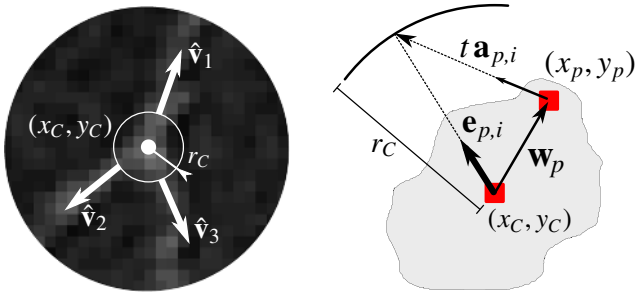

Fig. 11 Critical-point determination. A critical point is characterized by its type, centroid location (x_C, y_C), radius r_C, and its main branch directions $\hat{\mathbf{v}}_i$ (*left panel*, in this case a bifurcation), aggregated from the pixels (x_p, y_p) in the corresponding critical region (*right panel*)

$[\cos\hat{\alpha}_i(\mathbf{x}_p), \sin\hat{\alpha}_i(\mathbf{x}_p)]^T$. Each of these vectors defines a line $\mathbf{a}(t) = \mathbf{x}_p + t\,\mathbf{a}_{p,i}$ parameterized by $t \in \mathbb{R}$. We establish the projection of this line onto the circle $||\mathbf{x} - \mathbf{x}_C||^2 = r_C^2$ by substituting $\mathbf{x} = \mathbf{a}(t)$ and solving for t. From this we calculate the contributing unit vector (Fig. 11):

$$\mathbf{e}_{p,i} = \frac{1}{r_C}(\mathbf{w}_p + t\,\mathbf{a}_{p,i}) \tag{31}$$

which points from $\mathbf{x}_C$ to the intersection point. This is done for all $p = 1, \ldots, N_p$ in the region and $i = 1, \ldots, N_{\hat{\alpha}}$ for each p, resulting in the set of vectors $\{\mathbf{e}_{p,i}\}$. Next, a recursive mean-shift clustering algorithm (Cheng 1995) is applied to $\{\mathbf{e}_{p,i}\}$, which converges to a set $\{\hat{\mathbf{v}}_i\}$, where the cluster vectors $\hat{\mathbf{v}}_i$, $i = 1, \ldots, L$, represent the branches. For a critical region in I_{END}, we need only one main branch direction, which we simply take to be the direction $\hat{\mathbf{v}}_1$ to which the largest number of $\mathbf{e}_{p,i}$ were shifted. For a critical region in I_{JUN}, we take as the main branch directions the $\hat{\mathbf{v}}_i$ (at least three) to which the largest number of $\mathbf{e}_{p,i}$ were shifted. These calculations are performed for all critical regions.

Implementational Details

The method was implemented in the Java programming language as a plugin for the image processing and analysis tool ImageJ (Abràmoff et al. 2004; Schneider et al. 2012). Since the feature extraction step ("Feature Extraction"), in particular the matched filtering for angular profile analysis, is quite computationally demanding, we applied parallelization in multiple ways to reduce the running time to acceptable levels (on the order of minutes on a regular PC). Specifically, the directional filtering was split between CPU cores, each taking care of a subset of the directions (depending on the number of available cores). After this, the angular profile analysis and calculation of the features was also split, with each core processing a subset of the foreground image locations. This was sufficient for our experiments. Further improvement in running time (down to real-time if needed) could be achieved by mass parallelization using GPUs (graphical processing units) instead of CPUs.

Essential parameters that need to be set by the user are the scale parameters k and D ("Angular Profile Analysis") and the inflection points L_{LOW}, L_{HIGH}, U_{LOW}, U_{HIGH}, C_{LOW}, and C_{HIGH} of the input membership functions used by fuzzy-logic module FL_1 ("Termination and Junction Mapping"). In our applications we set D to the expected neuron diameter in a given set of images while $k = 0.7$ was kept fixed. The L inflection points are always in the range [0, 1] since the corresponding feature (likelihood) is normalized. Based on ample experience with many data sets we typically set L_{LOW} close to 0 and L_{HIGH} around 0.5 (Fig. 8). By contrast, the inflection points U correspond to a feature (centerline bending energy) that is not normalized and may vary widely from 0 to any positive value. To obtain sensible values for these we rely on the histogram of all calculated energy values in the image. Parameter U_{LOW} is set to the threshold computed by the well-known triangle algorithm, while typically $U_{\mathrm{HIGH}} \gg U_{\mathrm{LOW}}$. We note that the membership functions defined by these parameters are inverted (Fig. 8) such that the energy becomes a measure of smoothness. Finally, the C inflection points correspond again to a feature (correlation) with a fixed output range $[-1, 1]$. In our applications we usually set them to $C_{\mathrm{LOW}} \in [0.1, 0.5]$ and $C_{\mathrm{HIGH}} = 0.95$ (Fig. 8).

All other aspects of our method that could be considered as user parameters either follow directly from these essential parameters or are fixed to the standard values mentioned in the text. For example, the radius r_d of the circular neighborhood in the foreground selection step ("Foreground Selection") can be set equal to D, and the standard deviation σ_D of the Gaussian profile ("Angular Profile Analysis") can be set to $D/6$ to get a representative shape. Also, the output membership functions of FL_1 (input to FL_2) as well as the output membership functions of FL_2 are Gaussians with fixed levels and standard deviation ("Termination and Junction Mapping"), as they are not essentially influencing the performance of the algorithm.

Experimental Results

To evaluate the performance of our method in correctly detecting and classifying neuronal critical points we performed experiments with simulated images (using the ground truth available from the simulation) as well as with real fluorescence microscopy images (using manual annotation as the gold standard). After describing the performance measures ("Performance Measures"),

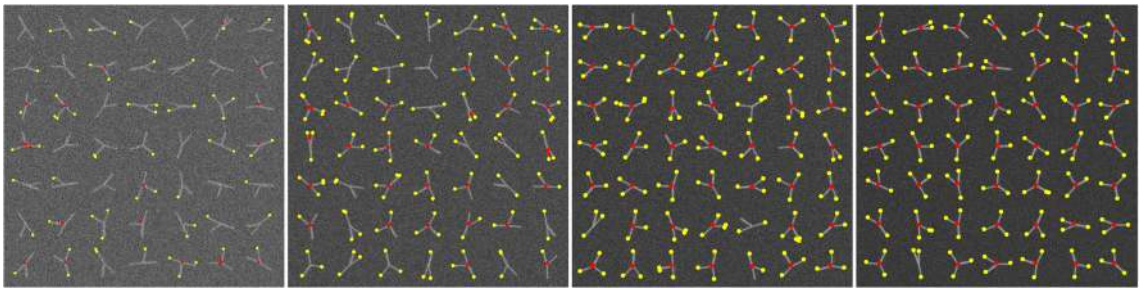

Fig. 12 Examples of simulated triplet images and detection results. Each triplet consists of three branches with different diameters which join at one end to form a bifurcation point and with the other ends being termination points. Images were generated at SNR = 2, 3, 4, 5 (*left* to *right* panel). The detection results with our method are indicated as red circles (bifurcation points) and yellow circles (termination points), where the radius of each circle reflects the size of the critical region found by our method

we present and discuss the results of the evaluation on simulated images, including synthetic triplets ("Evaluation on Simulated Triplet Images") and neurons ("Evaluation on Simulated Neuron Images"), and on real neuron images ("Evaluation on Real Neuron Images"), as well as the results of a comparison of our method with two other methods ("Comparison With Other Methods").

Performance Measures

Performance was quantified by counting the correct and incorrect hits and the misses of the detection with respect to the reference data. More specifically, we counted the true-positive (TP), false-positive (FP), and the false-negative (FN) critical-point detections, and we used these to calculate the recall $R = TP/(TP + FN)$ and precision $P = TP/(TP + FP)$. Both R and P take on values in the range from 0 (meaning total failure) to 1 (meaning flawless detection). They are commonly combined in the F-measure (Powers 2011), defined as the harmonic mean of the two: $F = 2RP/(R + P)$. The F-measure was computed separately for each type of critical points considered in this paper, yielding F_{END} for terminations and F_{JUN} for junctions. As a measure of overall performance we also computed the harmonic mean of the two F-measures: $F_{BOTH} = 2F_{END}F_{JUN}/(F_{END} + F_{JUN})$.

Evaluation on Simulated Triplet Images

Before considering full neuron images we first evaluated the performance of our method in detecting terminations and junctions in a very basic configuration as a function of image quality. To this end we used a triplet model, consisting of a single junction modeling a bifurcation, having three branches with arbitrary orientations (angular intervals) and diameters (Fig. 12). Orientations were randomly sampled from a uniform distribution in the range $[0, 2\pi]$ while prohibiting branch overlap. Since in principle the directional filtering step ("Angular Profile Analysis") uses a fixed kernel size D, we wanted to investigate the robustness of the detection for varying ratios d_{max}/d_{min} between the maximum and the minimum branch diameter in a triplet. For this experiment we considered ratios 1,0.33,2,2.5,3 by taking normalized diameter configurations (d_1, d_2, d_3) = (0.33,0.33,0.33), (0.3,0.3, 0.4), (0.2, 0.4,0.4), (0.2,0.3,0.5), (0.2,0.2,0.6), where in each case the actual smallest diameter was set to 3 pixels (the resolution limit) and the other diameters were scaled accordingly. For each configuration we simulated images with 1,000 well-separated triplets for signal-to-noise ratio levels SNR = 2, 3, 4, 5 (see cropped examples in Fig. 12). We chose these levels knowing that SNR = 4 is a critical level in other detection problems (Smal et al. 2010; Chenouard et al. 2014). Poisson noise

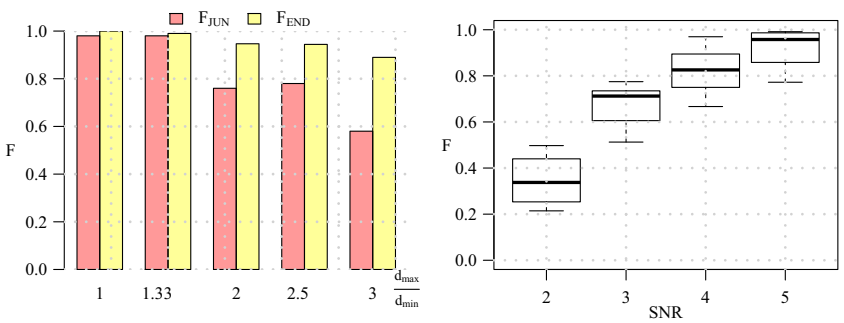

Fig. 13 Performance of our method in detecting terminations and junctions in simulated images of triplets. The values of F_{END} and F_{JUN} are shown (*left panel*) for the various branch diameter ratios d_{max}/d_{min} at SNR = 4. The distribution of F_{BOTH} values is shown as a box plot (*right panel*) for the various SNR levels

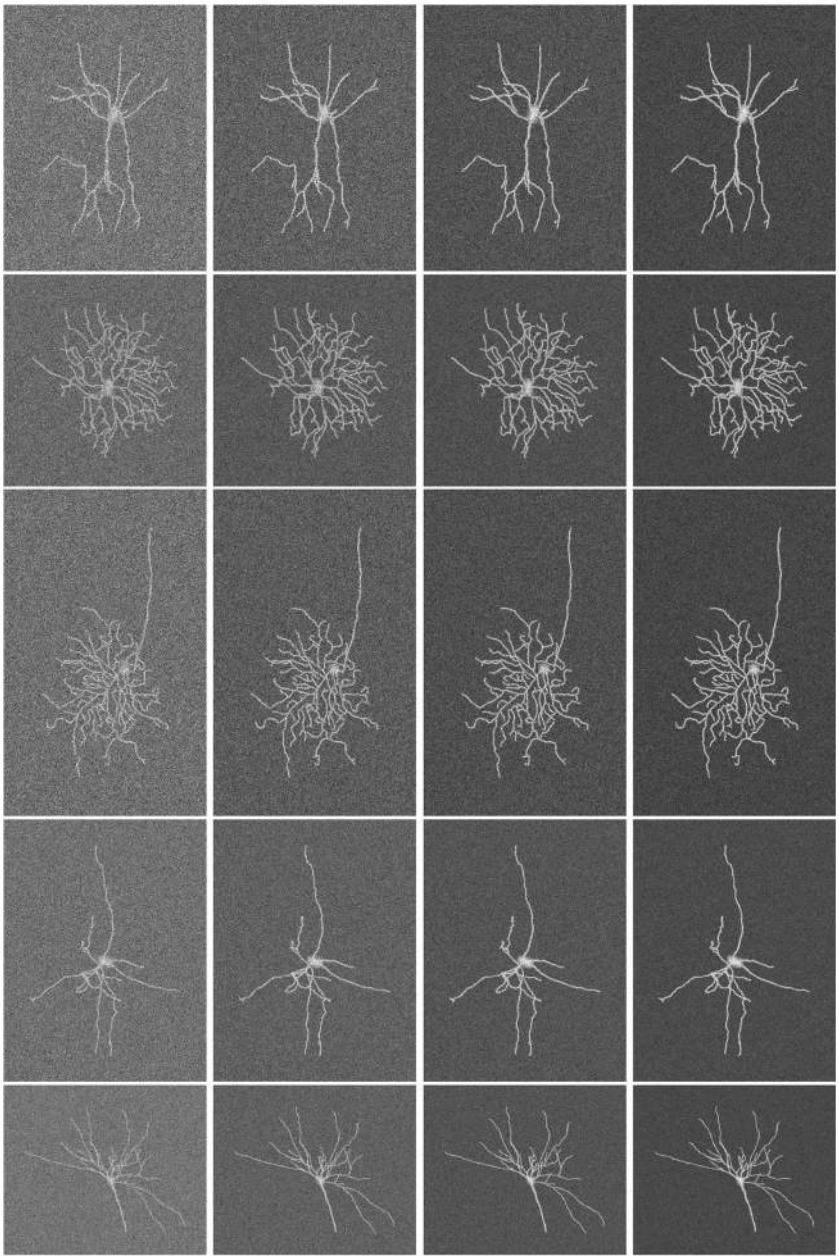

Fig. 14 Examples of simulated neuron images based on expert reconstructions from the NeuroMorpho.Org database. The images show a wide range of morphologies (one type per row) and image qualities of SNR = 2, 3, 4, 5 (from *left* to *right* per row)

was used in simulating fluorescence microscopy imaging of the triplets. From the results of this experiment (Fig. 13) we conclude that our method is very robust for diameter ratios $d_{\max}/d_{\min} \leq 2\frac{1}{2}$ and an image quality of SNR ≥ 4. We also conclude that our method is somewhat better in detecting terminations than detecting junctions. Example detection

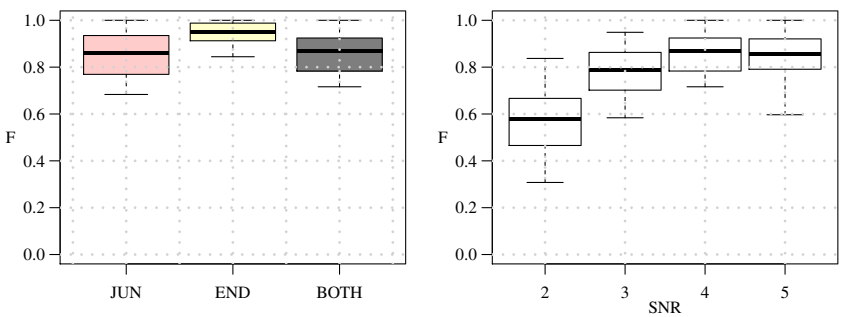

Fig. 15 Performance of our method in detecting terminations and junctions in 30 simulated images of neurons. The distributions of the F_{END}, F_{JUN}, and F_{BOTH} values are shown as box plots for SNR = 4 (*left panel*) and in addition the distribution of F_{BOTH} is shown for SNR = 2, 3, 4, 5 (*right panel*)

results for $d_{max}/d_{min} \leq 2$ for the considered SNR levels are shown in Fig. 12.

Evaluation on Simulated Neuron Images

To evaluate our method on more complex images, but for which we would still know the truth exactly, we simulated the imaging of complete neurons. Although there are various ways this could be done (Koene et al. 2009; Vasilkoski and Stepanyants 2009), we chose to use existing expert reconstructions from the NeuroMorpho.Org database (Ascoli et al. 2007). A total of 30 reconstructions from different neuron types were downloaded as SWC files (Cannon et al. 1998), in which the reconstructions are represented as a sequence of connected center-point locations in 3D with corresponding radii in micrometers. Fluorescence microscopy images were generated from these reconstructions in 2D by using a Gaussian point-spread function model and Poisson noise to emulate diffraction-limited optics and photon statistics. For each reconstruction we generated images of SNR = 2, 3, 4, 5 (Fig. 14). This way we obtained simulated images of neurons for which the termination and junction point locations were known exactly from the SWC files.

From the evaluation results (Fig. 15) we confirm the conclusion from the experiments on triplets that our method performs well for SNR $\geq$ 4 and is somewhat better in detecting terminations than detecting junctions. For SNR = 4 we find that the performance for junction detection is $F_{JUN} \approx 0.85$ while for termination detection $F_{END} \approx 0.95$. The higher performance for termination detection may be explained by the fact that the underlying image structure is usually less ambiguous (a single line-like structure surrounded by darker background) than in the case of junctions (multiple line-like structures that are possibly very close to each other). Example detection results are shown in Fig. 16.

Evaluation on Real Neuron Images

As the ultimate test case we also evaluated our method on real fluorescence microscopy images of neurons from a published study (Steiner et al. 2002). A total of 30 representative images were taken and expert manual annotations of the critical points were obtained to serve as the gold standard in this experiment. Needless to say, real images are generally more challenging than simulated images, as they contain more ambiguities due to labeling and imaging imperfections, and thus we expected our method to show lower performance. Since in this case we have no control over the SNR in the images we report the detection results of all images together. From the evaluation results (Fig. 17) we find that the median performance in detecting critical points is $F_{JUN} = 0.81$ for junctions and $F_{END} = 0.73$ for terminations while $F_{BOTH} = 0.76$. As expected, these numbers are lower than those of the simulated neuron images. Surprisingly, we observe that in the real images our method is better in detecting junctions than detecting terminations. A possible explanation for this could be that in the simulated images we used a constant intensity for the neuron branches, as a result of which terminations are as bright as junctions but

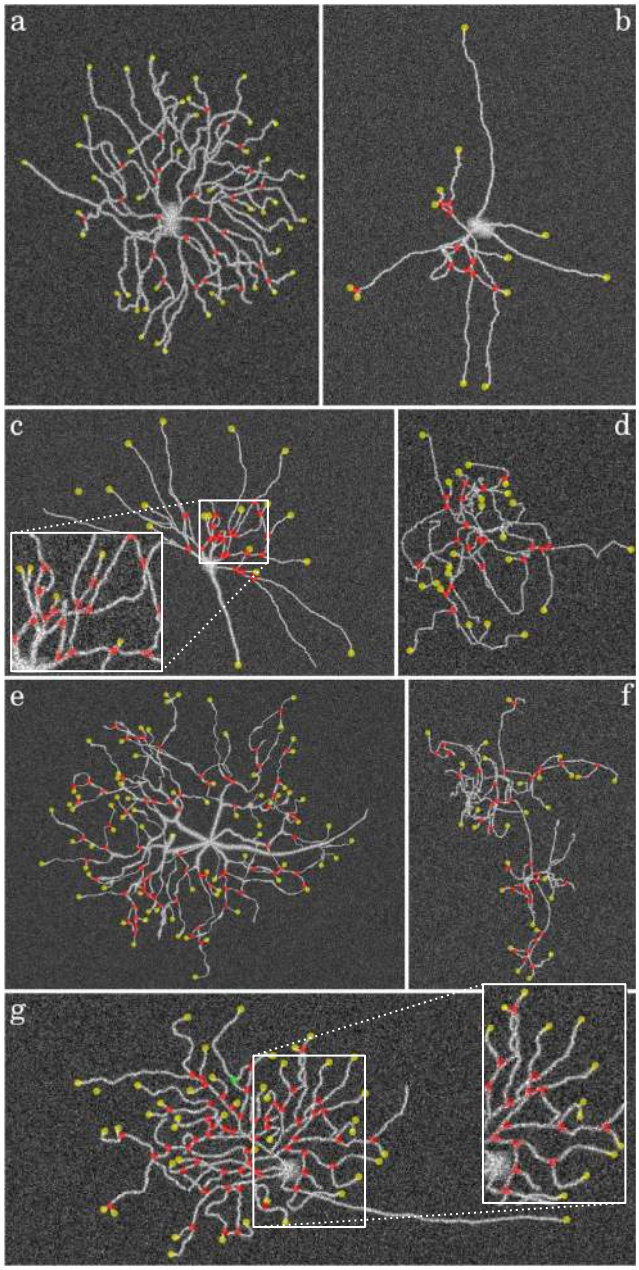

Fig. 16 Example detection results in simulated neuron images at SNR = 4. The images are contrast enhanced and show the detected terminations (*yellow circles*) and junctions (*red circles*) as overlays with fixed radius for better visibility. The value of F_{BOTH} in these examples is **a** 0.69, **b** 0.85, **c** 0.85, **d** 0.77, **e** 0.75, **f** 0.68, **g** 0.86

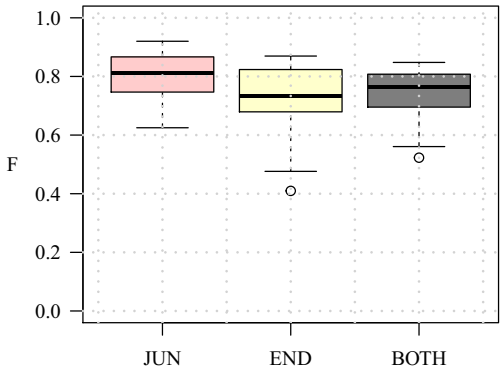

Fig. 17 Performance of our method in detecting terminations and junctions in 30 real fluorescence microscopy images of neurons. The distributions of the F_{END}, F_{JUN}, and F_{BOTH} values for all images together are shown as box plots

much less ambiguous due to a clear background, while in the real images the terminations are usually much less clear due to labeling imperfections and the fact that the branch tips tend to be thinner and thus less bright than the junctions. This illustrates the limitations of the simulations. Example detection results are shown in Fig. 18.

Comparison With Other Methods

Finally we sought to compare the performance of our Neuron Pinpointer (NP) method with other methods. Since we were not aware of other methods explicitly designed to detect and classify critical points in neuron images before reconstruction, we considered two existing software tools relevant in this context and we compared their implicit detection capabilities with our explicit method. If our method performs better, this would indicate that the existing methods may be improved by exploiting the output of our method.

The first tool, AnalyzeSkeleton (AS) (Arganda-Carreras et al. 2010), available from http://fiji.sc/AnalyzeSkeleton, is an ImageJ plugin for finding and counting all end-points and junctions in a skeleton image. To obtain skeleton images of our neuron images, we used the related skeletonization plugin available from the same developers, http://fiji.sc/Skeletonize3D, which is inspired by an advanced thinning algorithm (Lee et al. 1994). The input for the latter is a binary image obtained by segmentation based on smoothing (to reduce noise) and thresholding. For our experiments we considered a range of smoothing scales and manually selected thresholds as well as automatically determined thresholds using the following algorithms from ImageJ: Intermodes, Li, MaxEntropy, RenyiEntropy, Moments, Otsu, Triangle, and Yen. All of these were tried in combination with the AS method and the highest F-scores were used.

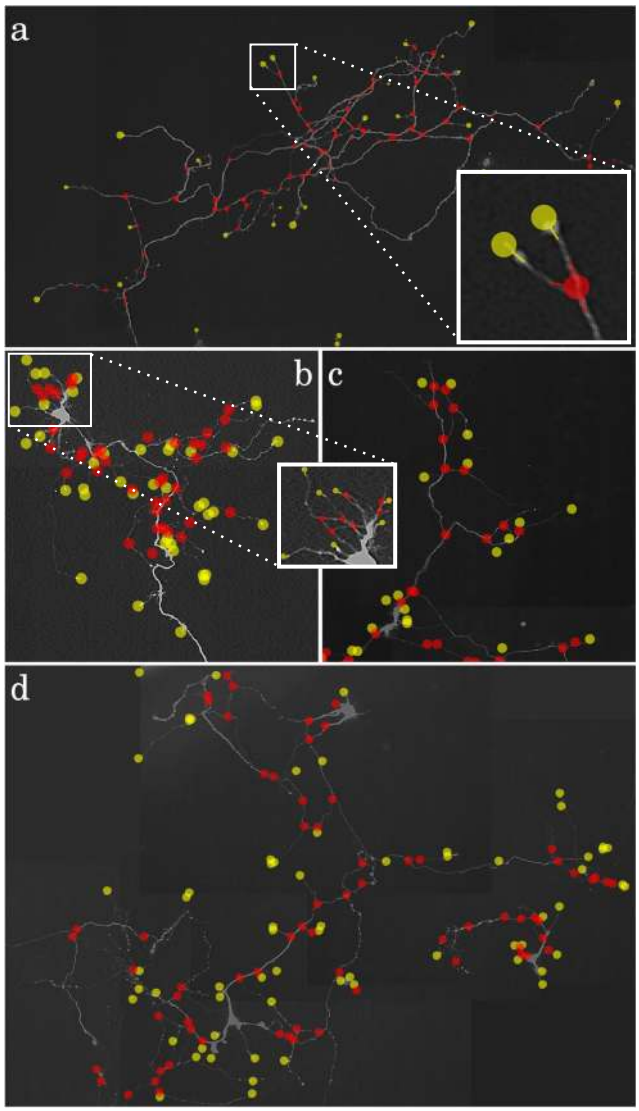

Fig. 18 Example detection results for four real neuron images. The images show the detected terminations (*yellow circles*) and junctions (*red circles*) as overlays with fixed radius for better visibility. The value of F_{BOTH} in these examples is **a** 0.82, **b** 0.78, **c** 0.68, **d** 0.65

The second tool, All-Path-Prunning (APP2) (Xiao and Peng 2013), is a plugin for Vaa3D (Peng et al. 2010; Peng et al. 2014), available from http://www.vaa3d.org/. It was not designed specifically for a priori critical-point detection but for fully automatic neuron reconstruction. Nevertheless, in producing a tree representation of a neuron, the reconstruction algorithm must somehow identify the branch end-points and junctions, and for our experiments we can easily retrieve them from the SWC output files. In principle, any neuron reconstruction method is also implicitly a critical-point detection method, and we can quantify its performance by comparing the output tree nodes with the reference data. The interesting question is whether an explicit

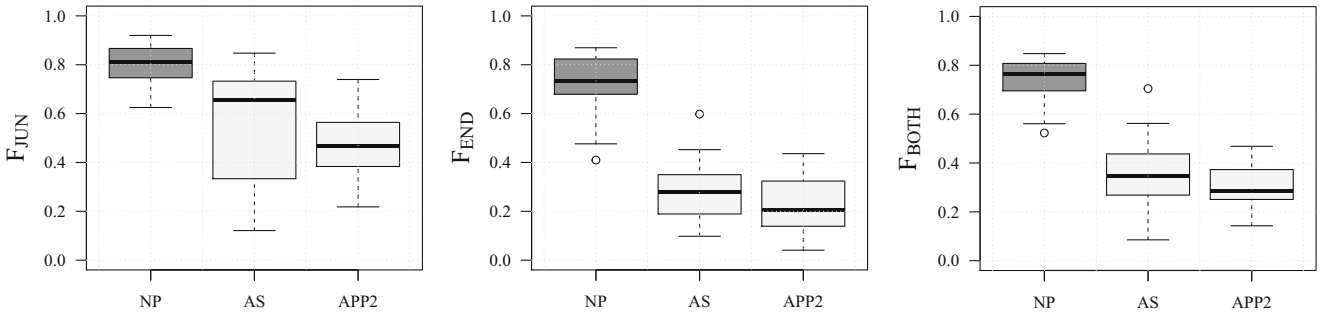

Fig. 19 Critical-point detection performance of our method (NP) compared to two other methods (AS and APP2). The median values of F_{JUN} (*left plot*) are 0.81 (NP), 0.65 (AS), and 0.47 (APP2). The median values of F_{END} (*middle plot*) are 0.73 (NP), 0.28 (AS), and 0.21 (APP2). Finally, the median values of F_{BOTH} (*right plot*) are 0.76 (NP), 0.35 (AS), and 0.29 (APP2)

detector such as NP outperforms the implicit detection carried out in a tool such as APP2. We manually adjusted the user parameters of the tool to get optimal performance in our experiments.

A comparison of the F-scores of NP, AS, and APP2 for the 30 real neuron images used in our experiments is presented in Fig. 19. From the plots we see that the detection rates of our NP method are substantially higher than those of AS and APP2. The difference is especially noticeable for the termination points. More specifically, the difference between F_{END} and F_{JUN} is relatively small for NP, but much larger for both AS and APP2. This indicates a clear advantage of using our explicit and integrated approach for detecting critical points, as accurate neuron reconstruction requires accurate detection of both junctions and terminations. However, with the current implementation, this advantage does come at a cost: timing of the three methods on a standard PC (with Intel Core i7-2630QM 2GHz CPU and 6 GB total RAM) revealed that with our images of 10^5 to 10^6 pixels in size, NP took about 40 seconds per image on average, while both AS and APP2 took only about 1.5 seconds per image. Fortunately, since virtually all the computation time of our method is spent in the directional filtering step, which is highly parallelizable, this cost can be reduced to any desired level by employing many-core hardware (such as GPUs).

Conclusions

We have presented a novel method for solving the important problem of detecting and characterizing critical points in the tree-like structures in neuron microscopy images. Based on a directional filtering and feature extraction algorithm in combination with a two-stage fuzzy-logic based reasoning system, it provides an integrated framework for the simultaneous identification of both terminations and junctions. From the results of experiments on simulated as well as real fluorescence microscopy images of neurons, we conclude that our method achieves substantially higher detection rates than the rates that can be inferred from existing neuron reconstruction methods. This is true for both junction points and termination points, but especially for the latter, which are of key importance in obtaining faithful reconstructions. Altogether, the results suggest that our method may provide important clues to improve the performance of reconstruction methods.

Actual integration of our detection method with existing tracing methods was outside the scope of the present study, but we are currently in the process of developing a new neuron tracing method and, in that context, we aim to perform an extensive evaluation of the beneficial effects of the presented method also on existing tracing methods. For this purpose we also aim to extend our method to 3D, where the exact same workflow could be used, except that the angular profile analysis and the final critical-point determination step would involve two angles (azimuth and elevation) instead of one. Also, it would require mass parallelization of the image filtering step to keep the running times of the method acceptable, but this should be straightforward in view of the highly parallel nature of this step.

Although we focused on neuron analysis in this work, our method may also be potentially useful for other applications involving tree-like image structures, such as blood vessel or bronchial tree analysis, but this requires further exploration. For this purpose it may be helpful to increase the robustness of the detection method to larger branch diameter ratios than tested in this paper. This could be done, for example, by using multiscale filtering approaches, or by selective morphological thinning (or thickening).

Information Sharing Statement

The software implementation of the presented method is available as an ImageJ (RRID:nif-0000-30467) plugin from https://bitbucket.org/miroslavradojevic/npinpoint.

Acknowledgments This work was funded by the Netherlands Organization for Scientific Reseach (NWO grant 612.001.018 awarded to Erik Meijering).

References

Abràmoff, M.D., Magalhães, P.J., & Ram, S.J. (2004). Image processing with ImageJ. *Biophotonics International*, *11*(7), 36–43.

Agam, G., Armato, I.S.G., & Wu, C. (2005). Vessel tree reconstruction in thoracic CT scans with application to nodule detection. *IEEE Transactions on Medical Imaging*, *24*(4), 486–499.

Aibinu, A.M., Iqbal, M.I., Shafie, A.A., Salami, M.JE., & Nilsson, M. (2010). Vascular intersection detection in retina fundus images using a new hybrid approach. *Computers in Biology and Medicine*, *40*(1), 81–89.

Al-Kofahi, Y., Dowell-Mesfin, N., Pace, C., Shain, W., Turner, J.N., & Roysam, B. (2008). Improved detection of branching points in algorithms for automated neuron tracing from 3D confocal images. *Cytometry Part A*, *73*(1), 36–43.

Anderton, B.H., Callahan, L., Coleman, P., Davies, P., Flood, D., Jicha, G.A., Ohm, T., & Weaver, C. (1998). Dendritic changes in Alzheimer's disease and factors that may underlie these changes. *Progress in Neurobiology*, *55*(6), 595–609.

Arganda-Carreras, I., Fernández-González, R., Muñoz-Barrutia, A., & Ortiz-De-Solorzano, C. (2010). 3D reconstruction of histological sections: application to mammary gland tissue. *Microscopy Research and Technique*, *73*(11), 1019–1029.

Ascoli, G.A. (2002). *Computational neuroanatomy: Principles and methods*. Totowa: Humana Press.

Ascoli, G.A., Donohue, D.E., & Halavi, M. (2007). NeuroMorpho.Org: a central resource for neuronal morphologies. *Journal of Neuroscience*, *27*(35), 9247–9251.

Azzopardi, G., & Petkov, N. (2013). Automatic detection of vascular bifurcations in segmented retinal images using trainable COSFIRE filters. *Pattern Recognition Letters*, *34*(8), 922–933.

Bas, E., & Erdogmus, D. (2011). Principal curves as skeletons of tubular objects: locally characterizing the structures of axons. *Neuroinformatics*, *9*(2-3), 181–191.

Basu, S., Condron, B., Aksel, A., & Acton, S. (2013). Segmentation and tracing of single neurons from 3D confocal microscope images. *IEEE Journal of Biomedical and Health Informatics*, *17*(2), 319–335.

Bevilacqua, V., Cambò, S., Cariello, L., & Mastronardi, G. (2005). A combined method to detect retinal fundus features. In *Proceedings of the IEEE European conference on emergent aspects in clinical data analysis* (pp. 1–6).

Bevilacqua, V., Cariello, L., Giannini, M., Mastronardi, G., Santarcangelo, V., Scaramuzzi, R., & Troccoli, A. (2009). A comparison between a geometrical and an ANN based method for retinal bifurcation points extraction. *Journal of Universal Computer Science*, *15*(13), 2608–2621.

Bhuiyan, A., Nath, B., Chua, J., & Ramamohanarao, K. (2007). Automatic detection of vascular bifurcations and crossovers from color retinal fundus images. In *Proceedings of the international ieee conference on signal-image technologies and internet-based system* (pp. 711–718).

Calvo, D., Ortega, M., Penedo, M.G., & Rouco, J. (2011). Automatic detection and characterisation of retinal vessel tree bifurcations and crossovers in eye fundus images. *Computer Methods and Programs in Biomedicine*, *103*(1), 28–38.

Can, A., Shen, H., Turner, J.N., Tanenbaum, H.L., & Roysam, B. (1999). Rapid automated tracing and feature extraction from retinal fundus images using direct exploratory algorithms. *IEEE Transactions on Information Technology in Biomedicine*, *3*(2), 125–138.

Cannon, R., Turner, D., Pyapali, G., & Wheal, H. (1998). An online archive of reconstructed hippocampal neurons. *Journal of Neuroscience Methods*, *84*(1), 49–54.

Cheng, Y. (1995). Mean shift, mode seeking, and clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *17*(8), 790–799.

Chenouard, N., Smal, I., de Chaumont, F., Maška, M., Sbalzarini, I.F., Gong, Y., Cardinale, J., Carthel, C., Coraluppi, S., Winter, M., Cohen, A.R., Godinez, W.J., Rohr, K., Kalaidzidis, Y., Liang, L., Duncan, J., Shen, H., Xu, Y., Magnusson, K.EG., Jaldén, J., Blau, H.M., Paul-Gilloteaux, P., Roudot, P., Kervrann, C., Waharte, F., Tinevez, J.Y., Shorte, S.L., Willemse, J., Celler, K., van Wezel, G.P., Dan, H.W., Tsai, Y.S., Ortiz, d.e.S.olrzano.C., Olivo-Marin, J.C., & Meijering, E. (2014). Objective comparison of particle tracking methods. *Nature Methods*, *11*(3), 281–289.

Choromanska, A., Chang, S.F., & Yuste, R. (2012). Automatic reconstruction of neural morphologies with multi-scale tracking. *Frontiers in Neural Circuits*, *6*(25), 1–13.

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

Dehmelt, L., Poplawski, G., Hwang, E., & Halpain, S. (2011). NeuriteQuant: an open source toolkit for high content screens of neuronal morphogenesis. *BMC Neuroscience*, *12*, 100.

Dercksen, V.J., Hege, H.C., & Oberlaender, M. (2014). The Filament Editor: an interactive software environment for visualization, proof-editing and analysis of 3D neuron morphology. *Neuroinformatics*, *12*(2), 325–339.

Deschênes, F., & Ziou D (2000). Detection of line junctions and line terminations using curvilinear features. *Pattern Recognition Letters*, *21*(6), 637–649.

Dima, A., Scholz, M., & Obermayer, K. (2002). Automatic segmentation and skeletonization of neurons from confocal microscopy images based on the 3-D wavelet transform. *IEEE Transactions on Image Processing*, *11*(7), 790–801.

Donohue, D.E., & Ascoli, G.A. (2008). A comparative computer simulation of dendritic morphology. *PLoS Computational Biology*, *4*(6), 1–15.

Donohue, D.E., & Ascoli, G.A. (2011). Automated reconstruction of neuronal morphology: An overview. *Brain Research Reviews*, *67*(1), 94–102.

Doyle, W. (1962). Operations useful for similarity-invariant pattern recognition. *Journal of the ACM*, *9*(2), 259–267.

Flannery, B.P., Press, W.H., Teukolsky, S.A., & Vetterling, W. (1992). *Numerical Recipes in C*. New York: Cambridge University Press.

Frangi, A.F., Niessen, W.J., Vincken, K.L., & Viergever, M.A. (1998). Multiscale vessel enhancement filtering. In *Proceedings of the international conference on medical image computing and computer-assisted interventation* (pp. 130–137).

Halavi, M., Hamilton, K.A., Parekh, R., & Ascoli, G.A. (2012). Digital reconstructions of neuronal morphology: three decades of research trends. *Frontiers in Neuroscience*, *6*(49), 1–11.

Hansen, T., & Neumann, H. (2004). Neural mechanisms for the robust representation of junctions. *Neural Computation*, *16*(5), 1013–1037.

He, W., Hamilton, T.A., Cohen, A.R., Holmes, T.J., Pace, C., Szarowski, D.H., Turner, J.N., & Roysam, B. (2003). Automated three-dimensional tracing of neurons in confocal and brightfield images. *Microscopy and Microanalysis*, *9*(4), 296–310.

Ho, S.Y., Chao, C.Y., Huang, H.L., Chiu, T.W., Charoenkwan, P., & Hwang, E. (2011). NeurphologyJ: an automatic neuronal morphology quantification method and its application in pharmacological discovery. *BMC Bioinformatics*, *12*(1), 230.

Hoover, A., Kouznetsova, V., & Goldbaum, M. (2000). Locating blood vessels in retinal images by piecewise threshold probing of a matched filter response. *IEEE Transactions on Medical Imaging*, *19*(3), 203–210.

Iber, D., & Menshykau, D. (2013). The control of branching morphogenesis. *Open Biology*, *3*(9), 1–16.

Kandel, E.R., Schwartz, J.H., & Jessell, T.M. (2000). *Principles of neural science*. New York: McGraw-Hill.

Kapur, J., Sahoo, P.K., & Wong, A.K. (1985). A new method for gray-level picture thresholding using the entropy of the histogram. *Computer Vision, Graphics, and Image Processing*, *29*(3), 273–285.

Koene, R.A., Tijms, B., van Hees, P., Postma, F., de Ridder, A., Ramakers, G.JA., van Pelt, J., & van Ooyen, A. (2009). NETMORPH: a framework for the stochastic generation of large scale neuronal networks with realistic neuron morphologies. *Neuroinformatics*, *7*(3), 195–210.

Laganiere, R., & Elias, R. (2004). The detection of junction features in images. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing* (pp. 573–576).

Leandro, J.JG., Cesar-Jr, R.M., & da F Costa, L. (2009). Automatic contour extraction from 2D neuron images. *Journal of Neuroscience Methods*, *177*(2), 497–509.

Lee, T.C., Kashyap, R.L., & Chu, C.N. (1994). Building skeleton models via 3-D medial surface axis thinning algorithms. *CVGIP: Graphical Models and Image Processing*, *56*(6), 462–478.

Lewis, J. (1995). Fast normalized cross-correlation. *Vision Interface*, *10*(1), 120–123.

Lin, Y.C., & Koleske, A.J. (2010). Mechanisms of synapse and dendrite maintenance and their disruption in psychiatric and neurodegenerative disorders. *Annual Review of Neuroscience*, *33*, 349.

Liu, Y. (2011). The DIADEM and beyond. *Neuroinformatics*, *9*(2–3), 99–102.

Longair, M.H., Baker, D.A., & Armstrong, J.D. (2011). Simple Neurite Tracer: open source software for reconstruction, visualization and analysis of neuronal processes. *Bioinformatics*, *27*(17), 2453–2454.

Lu, J., Fiala, J.C., & Lichtman, J.W. (2009). Semi-automated reconstruction of neural processes from large numbers of fluorescence images. *PLoS One*, *4*(5), e5655.

Luisi, J., Narayanaswamy, A., Galbreath, Z., & Roysam, B. (2011). The FARSIGHT trace editor: an open source tool for 3-D inspection and efficient pattern analysis aided editing of automated neuronal reconstructions. *Neuroinformatics*, *9*(2–3), 305–315.

Meijering, E. (2010). Neuron tracing in perspective. *Cytometry Part A*, *77*(7), 693–704.

Meijering, E., Jacob, M., Sarria, J.CF., Steiner, P., Hirling, H., & Unser, M. (2004). Design and validation of a tool for neurite tracing and analysis in fluorescence microscopy images. *Cytometry Part A*, *58*(2), 167–176.

Mendel, J.M. (1995). Fuzzy logic systems for engineering: a tutorial. *Proceedings of the IEEE*, *83*(3), 345–377.

Michaelis, M., & Sommer, G. (1994). Junction classification by multiple orientation detection. In *Proceedings of the European conference on computer vision* (pp. 101–108).

Myatt, D.R., Hadlington, T., Ascoli, G.A., & Nasuto, S.J. (2012). Neuromantic – from semi-manual to semi-automatic reconstruction of neuron morphology. *Frontiers in Neuroinformatics*, *6*(4), 1–14.

Narayanaswamy, A., Wang, Y., & Roysam, B. (2011). 3-D image preprocessing algorithms for improved automated tracing of neuronal arbors. *Neuroinformatics*, *9*(2–3), 219–231.

Narro, M.L., Yang, F., Kraft, R., Wenk, C., Efrat, A., & Restifo, L.L. (2007). NeuronMetrics: software for semi-automated processing of cultured neuron images. *Brain Research*, *1138*, 57–75.

Obara, B., Fricker, M., Gavaghan, D., & Grau, V. (2012a). Contrast-independent curvilinear structure detection in biomedical images. *IEEE Transactions on Image Processing*, *21*(5), 2572–2581.

Obara, B., Fricker, M., & Grau, V. (2012b). Contrast independent detection of branching points in network-like structures. In *Proceedings of SPIE medical imaging: image processing* (p. 83141L).

Peng, H., Ruan, Z., Long, F., Simpson, J.H., & Myers, E.W. (2010). V3D enables real-time 3D visualization and quantitative analysis of large-scale biological image data sets. *Nature Biotechnology*, *28*(4), 348–353.

Peng, H., Long, F., Zhao, T., & Myers, E. (2011). Proof-editing is the bottleneck of 3D neuron reconstruction: the problem and solutions. *Neuroinformatics*, *9*(2–3), 103–105.

Peng, H., Bria, A., Zhou, Z., Iannello, G., & Long, F. (2014). Extensible visualization and analysis for multidimensional images using Vaa3D. *Nature Protocols*, *9*(1), 193–208.

Pool, M., Thiemann, J., Bar-Or, A., & Fournier, A.E. (2008). NeuriteTracer: a novel ImageJ plugin for automated quantification of neurite outgrowth. *Journal of Neuroscience Methods*, *168*(1), 134–139.

Powers, D.MW. (2011). Evaluation: from precision, recall, and F-measure to ROC, informedness, markedness and correlation. *Journal of Machine Learning Technologies*, *2*(1), 37–63.

Püspöki, Z., Vonesch, C., & Unser, M. (2013). Detection of symmetric junctions in biological images using 2-D steerable wavelet transforms. In *Proceedings of the IEEE international symposium on biomedical imaging* (pp. 1496–1499).

Radojević, M., Smal, I., Niessen, W., & Meijering, E. (2014). Fuzzy logic based detection of neuron bifurcations in microscopy images. In *Proceedings of the IEEE international symposium on biomedical imaging* (pp. 1307–1310).

Santamaría-Pang, A., Hernandez-Herrera, P., Papadakis, M., Saggau, P., & Kakadiaris, I.A. (2015). Automatic morphological reconstruction of neurons from multiphoton and confocal microscopy images using 3D tubular models. *Neuroinformatics*, *13*(3), 297–320.

Schmitt, S., Evers, J.F., Duch, C., Scholz, M., & Obermayer, K. (2004). New methods for the computer-assisted 3-D reconstruction of neurons from confocal image stacks. *NeuroImage*, *23*(4), 1283–1298.

Schneider, C.A., Rasband, W.S., & Eliceiri, K.W. (2012). NIH Image to ImageJ: 25 years of image analysis. *Nature Methods*, *9*(7), 671–675.

Senft, S.L. (2011). A brief history of neuronal reconstruction. *Neuroinformatics*, *9*(2–3), 119–128.

Sinzinger, E.D. (2008). A model-based approach to junction detection using radial energy. *Pattern Recognition*, *41*(2), 494–505.

Šišková, Z., Justus, D., Kaneko, H., Friedrichs, D., Henneberg, N., Beutel, T., Pitsch, J., Schoch, S., Becker, A., von der Kammer, H., & Remy S (2014). Dendritic structural degeneration is functionally linked to cellular hyperexcitability in a mouse model of Alzheimer's disease. *Neuron*, *84*(5), 1023–1033.

Smal, I., Loog, M., Niessen, W., & Meijering, E. (2010). Quantitative comparison of spot detection methods in fluorescence microscopy. *IEEE Transactions on Medical Imaging*, *29*(2), 282–301.

Sonka, M., Hlavac, V., & Boyle, R. (2007). *Image processing, analysis, and machine vision*. Florence: Cengage Learning.

Steiner, P., Sarria, J.CF., Huni, B., Marsault, R., Catsicas, S., & Hirling, H. (2002). Overexpression of neuronal Sec1 enhances axonal branching in hippocampal neurons. *Neuroscience*, *113*(4), 893–905.

Su, R., Sun, C., & Pham, T.D. (2012). Junction detection for linear structures based on Hessian, correlation and shape information. *Pattern Recognition*, *45*(10), 3695–3706.

Svoboda, K. (2011). The past, present, and future of single neuron reconstruction. *Neuroinformatics*, *9*(2), 97–98.

Tsai, C.L., Stewart, C.V., Tanenbaum, H.L., & Roysam, B. (2004). Model-based method for improving the accuracy and repeatability of estimating vascular bifurcations and crossovers from retinal fundus images. *IEEE Transactions on Information Technology in Biomedicine*, *8*(2), 122–130.

Türetken, E., González, G., Blum, C., & Fua, P. (2011). Automated reconstruction of dendritic and axonal trees by global optimization with geometric priors. *Neuroinformatics*, *9* (2–3), 279–302.

Vasilkoski, Z., & Stepanyants, A. (2009). Detection of the optimal neuron traces in confocal microscopy images. *Journal of Neuroscience Methods*, *178*(1), 197–204.

Weaver, C.M., Hof, P.R., Wearne, S.L., & Lindquist, W.B. (2004). Automated algorithms for multiscale morphometry of neuronal dendrites. *Neural Computation*, *16*(7), 1353–1383.

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

Xiong, G., Zhou, X., Degterev, A., Ji, L., & Wong, S.TC. (2006). Automated neurite labeling and analysis in fluorescence microscopy images. *Cytometry Part A*, *69*(6), 494–505.

Yu, W., Daniilidia, K., & Sommer, G. (1998). Rotated wedge averaging method for junction characterization. In *Proceedings of the IEEE computer society conference on computer vision and pattern recognition* (pp. 390–395).

Yuan, X., Trachtenberg, J.T., Potter, S.M., & Roysam, B. (2009). MDL constrained 3-D grayscale skeletonization algorithm for automated extraction of dendrites and spines from fluorescence confocal images. *Neuroinformatics*, *7*(4), 213–232.

Zadeh, L.A. (1975). Fuzzy logic and approximate reasoning. *Synthese*, *30*(3–4), 407–428.

Zhang, Y., Zhou, X., Degterev, A., Lipinski, M., Adjeroh, D., Yuan, J., & Wong, S.TC. (2007). Automated neurite extraction using dynamic programming for high-throughput screening of neuron-based assays. *NeuroImage*, *35*(4), 1502–1515.

Zhou, J., Chang, S., Metaxas, D., & Axel, L. (2007). Vascular structure segmentation and bifurcation detection. In *Proceedings of the IEEE international symposium on biomedical imaging* (pp. 872–875).

Zhou, Z., Liu, X., Long, B., & Peng, H. (2015). TReMAP: automatic 3D neuron reconstruction based on tracing, reverse mapping and assembling of 2D projections. *Neuroinformatics*. in press.