Multiphoton excitation microscopy for reconstruction and analysis of single neuron morphology

Espen Hartveit, Bas-Jan Zandt and Margaret Lin Veruki

University of Bergen, Department of Biomedicine, Bergen, Norway.

Corresponding author: Espen Hartveit, University of Bergen, Department of Biomedicine, Jonas Lies vei 91, N-5009 Bergen, Norway.

espen.hartveit@biomed.uib.no

Number of figures: 4

Number of tables: 0

Running head: MPE microscopy and neuron reconstruction

i. Abstract

Neurons are the main cellular components of the circuits of the central nervous system (CNS). The dendritic and axonal morphology of individual neurons display marked variability between neurons in different regions of the CNS and there is evidence that the morphology of a neuron has a strong impact on its function. For studies of structure-function relationships of specific types of neurons, it is important to visualize and quantify the complete neuronal morphology. In addition, realistic and detailed morphological reconstruction is essential for developing compartmental models that can be used for studying neuronal computation and signal processing. Here we describe in detail how multiphoton excitation (MPE) microscopy of dye-filled neurons can be used for visualization and imaging of neuronal morphology, followed by a workflow with digital deconvolution and manual or semi-automatic morphological reconstruction. The specific advantages of MPE structural imaging are low phototoxicity, the ease with which it can be combined with parallel physiological measurements from the same neurons, and the elimination of tissue post-processing and fixation-related artefacts. Because manual morphological reconstruction can be very time consuming, the chapter also includes a detailed, step-by-step description of a work flow for semi-automatic morphological reconstruction (using freely available software developed in our laboratory), exemplified by reconstruction of a retinal amacrine cell (AII).

ii. Key words

Computational neuroanatomy, Morphology, Neuronal reconstruction, Multiphoton excitation microscopy, 3D microscopy, Dendrites, Morphometry, Retina

1 Introduction

1.1 From Waldeyer and Cajal to MPE Microscopy

In what later became known as the neuron doctrine, Wilhelm Waldeyer proposed in a review published in 1891 that "the nerve cell is the anatomical, physiological,

metabolic, and genetic unit of the nervous system" [1] (quoted and translated in [2]). Despite the wealth of discoveries that have accumulated over the more than 100 years that have since passed, the central tenet of this theory has stood the test of time: individual nerve cells, or neurons, are still considered to be a relevant level for investigating the function of the central nervous system (CNS) [3-5]. Neurons are the main cellular components of the CNS circuits that give rise to a rich variety of neural functions, including sensation and perception, motor action, and cognition. Despite enormous variability in the details of their morphology, neurons are characterized by the presence of multiple branching processes with specialized morphology. With the development of Golgi's technique for silver impregnation of individual neurons [6], and Cajal's technical refinements and subsequent application to essentially all regions of the central nervous system of several different species, this morphological variability was clearly described and documented [7, 8]. Despite delayed recognition of the importance of dendritic morphology for the functional properties of a neuron [9, 10], there is now strong evidence that the specific morphology of a neuron has a dramatic impact on its function [11]. Although the dendritic morphology of a given type of neuron can potentially be constrained by factors not directly related to signal processing and computation, e.g. development and innervation selectivity, it is generally accepted that, at least in principle, the computational and signal processing properties of a neuron are determined by its synaptic inputs, its three-dimensional (3D) dendritic morphology and the properties and location of the different ion channels expressed in the cell membrane.

Because neurons are the fundamental building blocks of the nervous system, it is important to visualize, characterize, and quantitatively describe the morphology of single neurons. This workflow is essential for studies that aim to understand the structure-function relationships of specific types of neurons and how the morphological variability can support a correspondingly rich diversity with respect to differential signal processing [4, 12]. The morphology of single neurons can be

visualized with several different techniques (for an extensive review, including technical descriptions, see [13]). For some purposes, digital images will suffice as documentation, but for quantitative studies, digital reconstruction of the 3D morphology is an essential step [14-16]. A simple variant of digital reconstruction is focused on creating a representation of the skeleton of the tree structure in 3D which can be adequate for studies that primarily aim to characterize e.g. changes in branching pattern during development [17]. In a more advanced variant of digital reconstruction, however, a realistic and detailed morphological reconstruction is generated and used for compartmental modeling of neuronal computation and signal processing (reviewed by [14, 18]).

Classically, reconstructing the morphology of single neurons meant performing a manual reconstruction by drawing, either directly from the light microscope (using special optical arrangements) or from the projected images of photographs. A disadvantage of both approaches is that they essentially lack any ability to represent 3D information in a way that makes the reconstruction useful for quantitative analysis. This situation changed dramatically with the development of increasingly sophisticated methods for reconstruction that combine light microscopic imaging and computer-aided neuronal tracing software [18-21]. Digital reconstruction can be performed simultaneously with light microscopic imaging or subsequent to and independently of imaging, such that the reconstruction is performed from a previously stored digital image stack. Irrespective of the exact method, such manual reconstruction can be extremely time-consuming and it can be challenging to maintain a consistent approach either throughout an extended reconstruction session or from one session to the next.

On this background, two major challenges of light-microscopic reconstruction of single neuron morphology can be summarized as follows: First, which method should be used for visualization of the neuron? Second, which method should be used for

reconstruction of the neuron? As will be reviewed in the remainder of this section, the answers to these questions are interrelated and depend strongly on the purpose of the reconstruction. We will also review important aspects of using MPE microscopy of dye-filled neurons for visualization and describe work flows for both manual and semi-automatic morphological reconstruction.

1.2 Visualization

With light-microscopic imaging, there are several different alternatives available for visualization of single neuron morphology. A major drawback of the classical Golgi method, irrespective of problems related to inconsistency, variable success rate, and reproducibility, is that it cannot easily be extended to a work flow that combines visualization with correlated physiological recordings under conditions where the same cell is used for both physiological and morphological analysis. With the advent of intracellular recording with microelectrodes, it became possible to fill cells with water-soluble substances that could be visualized in the light microscope and used for display and morphological reconstruction. For high-quality visualization, however, there are essentially only two types of substances available; tracers (like biocytin [22] and Neurobiotin [23]) and fluorescent dyes. Fluorescent dyes can also be used for injecting neurons in fixed tissue slices (using sharp microelectrodes) and this can generate excellent morphology (e.g. [24]), but this approach cannot be extended to encompass correlated physiological measurements from the same neurons. Filling neurons with biocytin or Neurobiotin, using either sharp microelectrodes or patch pipettes, requires the use of live tissue (*in vitro* or *in vivo*). Subsequent visualization based on development of an insoluble reaction product has been used in a number of morphological, as well as correlated morphological-physiological investigations (e.g. [25], for review see [14]). Alternatively, biocytin and Neurobiotin can be reacted with avidin linked to a fluorescent dye. However, both visualization methods require tissue fixation which can compromise and distort exact morphological reconstruction [14].

When it is a goal to avoid tissue fixation altogether, a useful alternative is to fill neurons in live tissue with a fluorescent dye, using either sharp microelectrodes or patch pipettes. Such dye-filled neurons can be imaged with wide-field fluorescence microscopy, but this technique lacks optical sectioning and there is a substantial risk of phototoxicity. Images obtained in this way can often be remarkably improved with digital deconvolution (see below), but generally not to the extent that they are adequate for detailed quantitative morphological reconstructions, as needed e.g. for compartmental modeling. In contrast, confocal laser-scanning microscopy provides optical sectioning and high resolution, but is not ideal when imaging fluorescent neurons in live tissue because of the high risk of phototoxicity (e.g. [26]). Confocal microscopy is ideal when imaging fluorescent neurons after tissue fixation, but, as indicated above, fixation introduces a considerable risk of tissue shrinkage and distortion that can lead to artefacts.

Most of the disadvantages discussed above can be strongly reduced (even eliminated) by imaging fluorescently filled neurons with MPE microscopy. This technique provides optical sectioning and the ability to image, with high resolution, deep into intact tissue [27]. MPE microscopy has few disadvantages (perhaps the only major one being the high cost of the required pulsed infrared [IR] laser) and for practical purposes, the resolution is essentially as good as that of confocal microscopy for the same conditions of imaging [28]. Because of the strongly reduced risk of phototoxicity compared to wide-field and confocal fluorescence microscopy, MPE microscopy is uniquely suited for imaging neurons in live tissue in combination with measurements of electrophysiological responses from the same cells [29-31]. This approach also eliminates the need to post-process tissue (which can save many hours of extra work), fixation-related artefacts, and the risk of tissue distortion that often take place when mounting and embedding tissue between microscope slides and cover slips. One limitation that is difficult to avoid results from the compromise between the motivation to image using an objective with the highest possible numerical aperture

(NA) for maximum resolution and the need for a working distance that enables the positioning of micropipettes for recording from and/or filling the neuron. Whereas confocal imaging generally makes use of objectives with NA in the range of 1.3 - 1.4, the short working distance of these objectives prevents their use for imaging neurons with attached micropipettes. Water-immersion objectives that can be used for MPE microscopic imaging of dye-filled neurons have, at most, NA values in the range of 0.95 - 1.05. To partially compensate for this shortcoming, we have implemented the use of post-acquisition deconvolution in our work flow for morphological reconstruction (described in detail below). Although intracellular electrophysiological recordings with sharp microelectrodes have contributed substantially to combined structure-function investigations of single neurons, the current method of choice for such investigations is whole-cell recording with the patch-clamp technique [32, 33].

1.3 Digital Morphological Reconstruction: Manual or Automatic?

Because manual morphological reconstruction can be difficult and time consuming, there has been a strong motivation to develop computerized morphological reconstruction procedures that require minimal user involvement (for recent reviews, see [34, 35]). Ideally, such automatic procedures would, in addition to saving time and increasing output, have the additional benefits of reduced subjectivity (i.e., errors resulting from variations in interpretation between different operators) and reduced susceptibility to operator fatigue. A number of different procedures for automatic reconstruction are available, both as freely available software from academic research laboratories (e.g. ORION [36], TREES [https://www.treestoolbox.org] [37, 38], Neuromantic [https://www.reading.ac.uk/neuromantic/body_index.php] [39], and neuTube [https://www.neutracing.com] [40]) and as commercial software for semi-automated and fully automated single neuron reconstruction (e.g. Neurolucida and Neurolucida 360 from MBF Bioscience [https://www.mbfbioscience.com], Imaris FilamentTracer from Bitplane [http://www.bitplane.com]). The extent to which the algorithms underlying the different procedures are freely available as source code

varies, but for some of the academic software, the full source code is available (e.g. the TREEs toolbox running in the MATLAB environment; [38]). This chapter includes a detailed, step-by-step description of a work flow for semi-automatic reconstruction using software recently developed in our laboratory [41]. The description is focused on reconstruction a retinal amacrine cell (AII), noting that this is just one example among several, but our software has performed reasonably well for this type of neuron. Our motivation for this development resulted from the specific challenge of reconstructing a densely branching neuron, but we expect the approach to be generally useful for reconstruction of neurons with widely varying morphologies.

2 Materials and General Methods

For animal anaesthesia, *in vitro* slice preparation, visualization with IR Dodt gradient contrast videomicroscopy, intra- and extracellular solutions, electrophysiological recording, data acquisition, MPE structural (and functional) imaging, and IR-laser scanning gradient contrast (IR-LSGC) imaging, see the chapter by Hartveit and Veruki (this volume).

When electrophysiological responses are recorded for compartmental modeling, it is useful to correct membrane potentials for liquid junction potentials [42]. Such potentials can be measured experimentally, although it is difficult to do so without introducing considerable error. Instead, it is recommended to estimate the liquid junction potential by theoretical calculation, using e.g. the software program JPCalcW / JPCalcWin. Some data acquisition programs can correct the holding potentials for liquid junction potentials online (e.g. Patchmaster from HEKA Elektronik) which can be very convenient.

2.1 Acquisition of an Image Stack

An image stack is acquired as a series of optical slices (XY) at fixed focal plane intervals from a given region. Depending on the size of the dendritic arborization, and

potentially even the size of the axonal field of the imaged neuron, it might be sufficient to acquire a single stack. Alternatively, tiling acquisition can be used when more than one stack is required to cover the cell and all its processes. When this is necessary, the individual image stacks should be sampled with sufficient overlap such that they can be assembled into a single composite stack before reconstruction (see below).

To obtain image stacks that can be adequately processed with deconvolution (see Section 2.3), images must be sampled at a rate close to the ideal Nyquist rate, both in the X, Y, and Z direction. This sets an upper limit for the voxel ("3D pixel") size along each dimension and constrains the size of the image as a function of the digital zoom, the number of pixels (along X and Y), and number of slices (along Z). The Nyquist sampling distance in the lateral direction is calculated as:

$$\Delta_x = \Delta_y = \frac{\lambda_{ex}}{4kn \times \sin\alpha}$$

and for the axial direction, the Nyquist sampling distance is calculated as:

$$\Delta_z = \frac{\lambda_{ex}}{2kn \times (1 - \cos\alpha)}$$

where n is the lens medium refractive index, k is the number of excitation photons (photon count; set to 2 for MPE microscopy), λ_{ex} is the wavelength of the excitation light, and α is the half-aperture angle of the objective (reviewed by [43]). The web site for Huygens deconvolution software (https://www.svi.nl/NyquistCalculator) has a simple browser-based application to directly perform these calculations. As an example, for λ_{ex} = 810 nm and a water immersion (n = 1.338) objective with NA = 0.95 (NA = n × sin α), we get a Nyquist sampling distance in the lateral direction of ~107 nm and in the vertical direction of ~512 nm. After calibrating the microscope, it is then possible to calculate the voxel size (XYZ) for any given combination of digital zoom and the number of pixels sampled in the X and Y directions for a given image. A useful approach is to adjust the digital zoom until the desired structure (e.g. dendritic arborization) almost fills the imaged region and then increase the number of pixels for ech image (e.g. to 1024[X] × 1024[Y]) until Nyquist rate sampling is satisfied. The

vertical interval between the slices of the image stack should be set directly in the image acquisition software. For acquisition of the complete arborization of an AII amacrine cell [44], we would typically sample image stacks with each slice set to 1024 × 1024 pixels and adjust the digital zoom to obtain an XY pixel size of 80 - 90 nm. Constrained by the available intervals of the vertical focus mechanism of the microscope, we would use a focal plane interval of 400 nm.

At each focal plane, a number of images can be averaged within each channel to improve the signal-to-noise ratio (SNR). It is tempting to increase the number of averages to improve the image quality, but there are trade-offs. Increased averaging does not improve the SNR linearly, it increases the risk of phototoxicity, and it increases the time required to sample a complete stack. When image stacks are subsequently processed by deconvolution (see below), we have found it optimal to average two frames for each slice.

For neurons with larger dendritic trees, the requirement of Nyquist sampling may not be compatible with acquiring the complete morphology in a single stack, and multiple stacks must be acquired in a tiling pattern. Combining the individual stacks to a single composite stack can be done with an "image montage" stitching function (e.g. in Neurolucida) that creates a 3D image montage by manually arranging the set of stacks to optimize X, Y, and Z alignment. An alternative computer program ("Volume integration and Alignment System"; VIAS) is freely available and designed to integrate multiple stacks into a single volumetric dataset (http://research.mssm.edu/cnic/tools-vias.html).

2.2 Image Processing Prior to Deconvolution

ScanImage [45], the MPE microscopy acquisition software used in our laboratory (see chapter by Hartveit and Veruki, this volume), stores the different channels of an image stack in an interleaved format. Thus, it is necessary to process the image stacks before

importing the data to the deconvolution software whenever more than one image channel has been acquired. For this, we use a simple routine developed in the IGOR Pro environment (64-bit) that takes a multi-channel image file, de-interleaves the data based on acquisition channels and saves each channel as an individual file. To compensate for drift and mechanical instabilities that can occur during the image acquisition (due to e.g. perfusion of the recording chamber), we use the Object Stabilizer module of Huygens to align images along the Z axis. The IGOR Pro *ImageRegistration* operation, as implemented in the SARFIA *RegisterStack* routines [46] can also be used to align the slices of an image stack. A specific advantage of the *ImageRegistration* operation is that it can align the image stack in one of the channels (e.g. the IR-LSGC channel) and then use the generated registration parameters to align the image stack of the other channels (e.g. fluorescence) according to the same parameters. This can be a useful feature when aligning image stacks of neurons with few processes.

2.3 Deconvolution of Fluorescence Image Stacks

Post-acquisition digital deconvolution is a very powerful technique for enhancing image quality [26, 47] and can be used with great advantage to increase the SNR and decrease the axial and lateral blurring [48] of image stacks before they are used for digital morphological reconstruction. Essentially, deconvolution removes noise and reassigns out-of-focus light, using a theoretically calculated or experimentally determined point spread function (PSF).

There are several options for deconvolution software, both freely available academic software as well as commercial software. With respect to commercial software, our experience is with the powerful and flexible Huygens package from Scientific Volume Imaging. The Huygens software comes in two different base versions, Essential and Professional (both 64-bit). Whereas they differ in some of the added functionality (and therefore the cost), the core of the deconvolution operation is identical between the two

versions. The deconvolution procedure depends on specifying a PSF for the imaging system, but can use either a PSF that is experimentally determined for a specific setup and imaging condition or a PSF that is calculated theoretically. A number of different algorithms are available for deconvolution within the Huygens software. For deconvolving MPE microscopic images, we have obtained the best results using the Classic Maximum Likelihood Estimation (CMLE) algorithm.

Before starting deconvolution of an image or image stack, the Huygens software requires user input of several parameters related to the microscope and imaging. Some parameters can be left with their default values calculated from data available in the image stack. The most critical user-specified parameter is the SNR. In theory the SNR value can be deduced from the image intensities and photon shot noise. However, we have adopted a more heuristic strategy to obtain an optimal setting, as is advised by the Huygens software. Increasing the SNR value used during deconvolution increases the sharpness of the restoration result, but when set higher than an optimal value, it leads to enhanced noise and spatial fragmentation of the structures in the images.

Fig. 1 near here

Although it can be time-consuming, it is well worth the effort to run a series of repeated deconvolutions with several different values of the SNR on the same image stack in order to estimate an optimal SNR (while keeping all other parameters and settings constant). In the course of a study with morphological reconstruction of dye-filled AII amacrine cells from the rat retina [44], we investigated this process in detail. An example of the results that can be obtained with this procedure is illustrated in Fig. 1, for an arbitrary region with a number of tightly packed dendritic processes imaged in a single focal plane of an AII amacrine image stack. The raw (unprocessed) image data are shown in Fig. 1a and Figs. 1b-g show the results for the same region after deconvolution with different values for the SNR (set to 1, 5, 10, 20, 40, and 80, respectively). Note that with increasing value for SNR, the sharpness of the

deconvolved images increases and noise is removed (corresponding to removing out-of-focus light), but when the SNR for deconvolution is increased beyond an optimal value, the deconvolved images begin to display structural fragmentation of the dendritic processes. This is a sign that the SNR values are too high and the resulting images are over-deconvolved. For a more detailed analysis, we plotted linear intensity profiles for fluorescence at a series of locations across several dendritic processes. An example is illustrated in Fig. 1, where the position of the intensity profile corresponds to the line displayed in Fig. 1a. The population of linear intensity profiles for all the deconvolution results (Figs. 1b-g), together with that for the raw data, is illustrated in Fig. 1h. When the SNR for deconvolution is increased, it progressively increases the peak value of the intensity profile and the profiles remain smooth until the optimum SNR value has been reached. However, when the SNR is increased beyond the optimum (approximately 20 for the example illustrated in Fig. 1), the intensity profile displays increased noise, reflecting the spatial (morphological) fragmentation in the corresponding deconvolved images. This procedure depends on the judgment of and input from the operator. To ensure that the optimally deconvolved image stack is selected for morphological reconstruction, the procedure described here with visual inspection and generation of linear intensity profiles should be applied to several different regions (and processes) of the same stack.

3 Manual 3D Morphological Reconstruction

3.1 General Aspects

In our laboratory, we routinely use the computer programs Neurolucida [19] and Neurolucida 360 for manual, computer-aided, quantitative morphological reconstruction of fluorescently labeled neurons. During reconstruction, the fluorescent processes in the different slices of an image stack are tracked by scrolling up and down through the stack and the computer mouse is used to delineate a given process and visually determine its diameter. Further details can be found in the on-line documentation (https://learn.mbfbioscience.com). 3D reconstruction of the soma can

be done in two different ways. One can identify the image slice corresponding to the largest outline ("maximum projection") and trace the circumference with a single contour. The volume of the soma is then estimated by spatial integration of this contour (through rotation) during the subsequent analysis. Alternatively, 3D reconstruction can be performed by tracing the soma with a series of contours, each at a different focal plane. In this case the volume of the soma is estimated by integrating over the series of individual contours. In general, we choose to use a single contour.

Fig. 2 near here

Figures 2a-c illustrate three different stages of the reconstruction workflow, with maximum intensity projections (MIPs) of the fluorescence image stack before (Fig. 2a) and after (Fig. 2b) deconvolution, and a projection of the final digital reconstruction (Fig. 2c). All projections have been overlaid on a single, representative image slice from the IR-LSGC channel (identical for panels a-c). The details of the dendritic arborization of the reconstructed neuron are more clearly displayed by the two-dimensional (2D) projection (shape plot) in Fig. 2d and the 3D visualization in Fig. 2e.

For general morphological analysis and quantification of dendritic branching of a complete reconstruction, there are many different options available in the program Neurolucida Explorer (64-bit, MBF Bioscience; https://www.mbfbioscience.com/neurolucida-explorer; https://www.mbfbioscience.com/help/neurolucida_explorer/Content/NeurolucidaExplorer.html). In addition, the freely available program L-measure (http://cng.gmu.edu:8080/Lm/; [49]) offers similar as well as additional analysis options compared to Neurolucida Explorer, but does not have visualization capabilities.

3.2 Special Considerations When Reconstructing Very Thin Processes

During manual reconstruction, it is essentially up to the operator to determine the thickness of the different fluorescent processes by visual inspection during the reconstruction, irrespective of whether this is done at the microscope or at the computer with an image stack. When morphological reconstruction is based on light microscopic imaging, it is a problem when the diameter of a neuronal process is below the resolution limit of light microscopy [14, 15]. For a self-luminous point object, as is the case when imaging subresolution structures with fluorescence light microscopy, the lateral (XY) Rayleigh two-point resolution (minimum resolved distance) is given by $0.61\lambda/NA$ (e.g. [26, 50]), where λ is the wavelength of the emitted light and NA is the numerical aperture of the microscope objective. With MPE microscopy, only the excitation wavelength is important, and the resolution is improved by $\sqrt{2}$ (in the ideal, diffraction-limited case, assuming that the laser beam completely fills the back-focal plane of the objective) and the equation becomes $0.61\lambda/(NA\sqrt{2})$ [51]. For example, in a situation with an excitation wavelength of 810 nm and an objective NA of 0.95, the resolution limit becomes approximately 0.37 μm in the ideal (diffraction-limited) case. If a neuronal process is thinner than this value, it can be detected if the fluorescence intensity is high enough to generate a signal above the noise level in the microscope hardware, but the diameter cannot be adequately resolved. To our knowledge, the only case where super-resolution (i.e., diffraction-unlimited) light microscopy has been used to image neuronal processes under conditions where the acquired images can be used for morphological reconstruction is the work of Sigal et al. [52]. Here, retinal neurons were imaged with stochastic optical reconstruction microscopy (STORM) applied to fluorescently labeled ultrathin sections, providing a lateral resolution of ~20 nm and an axial resolution of ~70 nm (limited by section thickness), which is sufficient for imaging some of the thinnest neuronal processes. However, sectioning of tissue for serial electron microscopy (EM) makes this approach very labor-intensive.

EM has a limit of resolution which is adequate for imaging the very thinnest neuronal processes, but has not routinely been applied for reconstructing complete arborizations of single neurons. The existence of ultrastructural data for a specific type of neuron, however, can often provide valuable information that can guide light microscopic reconstruction of processes with diameters below the limit of resolution of light microscopy [15]. An example is the detailed study of Tsukamoto & Omi of AII amacrine cells in mouse retina [53], which we used to guide our work with reconstruction of rat AII amacrine cells imaged with MPE microscopy [44]. We used the published 2D projections of complete EM reconstructions to make measurements from the thinnest processes illustrated. The estimated range of diameters were clearly below the expected limit of resolution for MPE microscopic imaging. We then subsequently corrected our morphological reconstructions digitally using a custom algorithm (for details, see [44]).

4 Semi-automatic and Automatic Morphological Reconstruction

There is a strong motivation to develop semi-automatic or even fully automatic methods for reconstructing the morphology of single neurons. The most important reason for this is undoubtedly the amount of time and work required for reconstructing even neurons with simple morphologies. During the course of manually reconstructing a large number of neurons from the mammalian retina, we became increasingly interested in supplementing the manual reconstructions with semi-automatic or fully automatic reconstructions. However, our success after trying out some of the available computer programs for automatic reconstruction was limited, both for freely available and commercial software. To compensate for this, we developed a method that combined elements of existing reconstruction procedures, in particular the algorithm implemented in the TREES toolbox [38]. To enhance the functionality, we extended this with an algorithm for connecting disconnected neuronal segments that typically result from a previous stage of image segmentation.

This was implemented with the Fast Marching method for generating image-extracted paths [54].

Fig. 3 near here

In the following, we present a detailed account and work flow for semi-automatic reconstruction using our approach (Fig. 3), with the retinal AII amacrine cell as an example. This is currently the only type of neuron for which our method has been extensively tested. Although the method was developed for reconstructing neurons with densely branching dendritic trees, we believe that it is likely to work well for other types of neurons as well. The software was developed in MATLAB and is freely available, including the source code (see Section 8; additional details can be found in the original publication [41]). Specific parts of the code use functions in the TREES toolbox [37, 38] and the CellSegm toolbox [55], both of which are freely available.

4.1 Image Preprocessing

The preprocessing consists of two steps:

1. Image restoration by deconvolution. This is described in detail in section 2.3 and is performed as for manual reconstruction. The Object Stabilizer module in Huygens is also used here to align images along the Z axis to compensate for drift and mechanical instabilities. Processed image stacks are saved in 16-bit TIFF file-format, utilizing the whole dynamic range, and converted to the NRRD format in Fiji (http://fiji.sc; [56]). Following the next step, all subsequent processing is performed with MATLAB and the NRRD format makes file handling in MATLAB easier.

Fig. 4 near here

2. Manual removal of the pipette and extraneous fluorescence. Before automatic cell segmentation, it is necessary to remove the image of the recording pipette used to fill the cell with fluorescent dye (Fig. 4a). This step can also be used to remove any extraneous fluorescence that sometimes accumulates around the site of recording, caused by leakage from the pipette when the recording was established (Fig. 4b).

Otherwise, as long as the intensity is higher than background, such false signals will be treated by the segmentation algorithm as being part of the cell. To remove both the pipette and extraneous fluorescence, our procedure depends on manual circumscription of the corresponding area in a frontal (XY) MIP of the image stack (using the *getline* function in MATLAB; Fig. 4c). In cases where the image of the pipette is projected on top of cellular processes in the MIP, this procedure will unfortunately not generate a satisfactory result and an alternative method must be developed.

4.2 Image Segmentation

First, before intensity segmentation with thresholding, the image stack is filtered (smoothed) with coherence-enhancing diffusion [57]. For segmenting the dendritic arborization, the procedure uses adaptive thresholding, with an adaptive filter size r = [2.5, 2.5, 5] (μm). To reduce the likelihood that noise is segmented, we routinely increase the adaptive threshold by a constant value equal to 5% of the maximum image intensity. Sometimes adaptive thresholding generates indentations or holes in thicker structures with size comparable to the size of the adaptive filter. To compensate for this, we perform a second segmentation by applying simple global thresholding (by trial and error, the threshold is set to ~11% of the maximum image intensity), followed by morphological operations that identify the largest connected component. The goal of the simple second segmentation is to obtain the soma and, for the specific example of the AII amacrine cell, the neck of the thick primary (apical) dendrite. The results from the two segmentation operations (adaptive thresholding and global threshold) are joined and followed by removing small components consisting of <20 voxels (considered to be noise).

In the next step, the goal is to connect all disconnected components generated by the initial segmentation by computing the optimal paths by which they can be connected. To accomplish this, our procedure uses the Fast Marching method, generally considered to be a versatile path extraction technique [54]. The procedure uses a point

in the soma as starting point and computes the Fast Marching arrival times for all voxels in the image stack. For the so-called speed function, we use the original image stack (after preprocessing). Then, segments are connected by back tracing the Fast Marching arrival time map from each segment to the soma. To accomplish this, the procedure automatically selects a point approximately in the center of the soma (for details, see [41]). The final (binary) representations of the cells are created by joining the voxels traversed by the connection paths and the segments from the initial intensity segmentation.

4.3 Tree Representation by Skeletonization and Tubularization

In the next step, the goal is to transform the binary cell representation (generated by the automated segmentation) to a tree representation. Our procedure uses a tree representation that follows the definitions of the SWC format. This is a simple tabular text format that lists a set of (X, Y, Z) positions and associated radii [58]. First, all slices within the segmented volume of the image stack are smoothed (3 x 3 pixel mean filter) and binarized by the Smooth and Make Binary functions in Fiji. Then, the segmented volume is skeletonized, i.e., thinned by surface erosion [59], using the Skeletonize3D function in Fiji. Each of the functions in Fiji is called automatically from within MATLAB. The coordinates of the voxels in the resulting skeleton are then used to generate a minimum spanning tree (MST; [60]), using the TREES toolbox [37, 38]. Generating an MST simultaneously optimizes both the total branch length and the total path length to the soma for all points (for details, see [41]). The process requires input from the user of a value for the so-called balancing factor that controls the relative weight between the wiring cost and the path length cost. For the AII amacrine cells imaged in our experiments, a small balancing factor of 0.01 provides a good result and essentially corresponds to drawing the shortest possible connections between points.

In the last step of this section, the goal is to obtain the diameters of the neuronal branches. It is well known that for image stacks generated by light microscopy, the

axial (Z) resolution is inherently lower than the lateral (XY) resolution. In both the raw image stacks and those obtained by segmentation, this is reflected by the elliptical cross-sections of the branches, i.e., they are much thicker along the Z direction. In our procedure, this imaging artefact is corrected by a process termed "tubularization" where the diameter associated with each reconstruction point is calculated as twice the distance from the point to the closest border of the segmented volume. This is estimated by fitting the largest possible sphere, centered at the reconstruction point, into the segmented volume. These results are obtained by calculating the distance transform of the segmented image [61, 62]. As for the manual reconstructions, our procedure enforces a minimum branch diameter (set by the user).

4.4 Postprocessing and Correction of Reconstruction Errors

It is essentially unavoidable that an automatic reconstruction will contain errors. Some of these can be corrected in a series of semi-automatic post-processing steps, with some requiring more explicit user intervention.

The first post-processing step is to clean the reconstructed tree from short and most likely false branches introduced during skeletonization. This is done with the *clean_tree* routine in the TREES toolbox [37, 38]. For this step the user has to input a value for the cleaning factor. Based on empirical testing, we set this value to 0.2, but for other cells different values might generate better results. The skeletonization procedure also usually generates a number of false branches inside and close to the outside of the soma. We have not found a satisfactory way to remove these branches programmatically and instead we have implemented a semi-automated procedure where the user is prompted to manually delineate a region around the soma where such "spurious" branches occur (using MATLAB's *getline* function). This is conveniently done by superimposing the generated skeleton on a MIP of the pre-processed image stack from two different perspectives (front and side views). All points with XY- and YZ-coordinates within the delineated outlines are then removed.

The second post-processing step is to compensate for the introduction of discrete steps along the reconstructed branches. If left uncorrected, this will artificially increase the apparent branch length. It is a common problem that both manual and automatic reconstructions suffer from discrete steps along the Z-dimension, but with the automatic reconstructions we have observed that a similar phenomenon can be observed along the X- and Y-dimensions as well, most likely due to the large number of reconstruction points (per unit branch length). As a correction, our procedure spatially filters the reconstruction using the *smooth_tree* function in the TREES toolbox [38].

In the last step, the procedure generates a single contour to represent the soma by automatically tracing the outline of the soma in each slice of the image stack and selecting the contour close to the center, typically the contour with the largest area [41]. For calculating the surface area and volume of the soma, analysis programs such as L-measure and NEURON (https://neuron.yale.edu/neuron/) assume that the soma has a primary axis in the XY plane and is rotationally symmetric. In the final step, the complete reconstruction is exported and saved in the SWC format.

4.5 Proofreading and Editing for Semi-Automatic versus Automatic Reconstructions

Inevitably, a procedure for automatic morphological reconstruction will generate errors that a human observer immediately recognizes as such. It is perhaps a paradox that when procedures for automatic reconstruction are designed by giving the user extensive control of the different steps of the reconstruction by providing access to a number of different parameters that control the reconstruction process, the user might end up spending more time tuning and varying the different parameters and re-running the reconstruction, such that in the end, little time is saved relative to full manual reconstruction. Similarly, if automatic reconstruction has to be followed by

difficult and time-consuming post-reconstruction manual editing, the time saved during automatic reconstruction is effectively lost and little real progress has been made. Naturally, this calls for powerful, real-time visualization tools that can be used for efficient evaluation and proofreading and also for correcting the reconstructions when needed (see [63, 64] for detailed discussions). Our procedure does not provide such functionality. Instead, we recommend visualization and proofreading using either commercial software like Neurolucida/Neurolucida 360 and Imaris or the freely available programs Vaa3D (http://www.alleninstitute.org/what-we-do/brain-science/research/products-tools/vaa3d/; [63, 65]), neuTube [40], and CVAPP (https://github.com/pgleeson/Cvapp-NeuroMorpho.org). Hopefully, even with an additional step of manual editing and post-reconstruction error correction, the total reconstruction time will be significantly shorter and less exhausting compared to a fully manual reconstruction.

5 Morphological Reconstruction for Compartmental Modeling

A major motivation for morphological reconstruction of any neuron with an arborizing dendritic tree is to perform computational modeling and simulations of signal propagation using realistic neuronal geometry (e.g. [36, 66]). Neuronal function can depend strongly on the dendritic tree morphology (e.g. [11]) and computer simulations based on accurate reconstruction of neuronal morphology and electrophysiological recording are considered necessary to understand the underlying mechanisms [67, 68]. Importantly, for a neuron with a complex dendritic arborization, the flow of electric current cannot be solved analytically and investigations must resort to compartmental representations where the neuronal structure is discretized. When the goal is to understand signal processing in a specific type of neuron and reproduce its electrical behavor in a series of conditions, the morphological reconstruction must be detailed and accurate. Ideally, such models should be developed from the same neurons from which the electrophysiological measurements are made [32, 33, 69]. Even though morphological reconstructions of the desired type of neuron might be available from a

data base, they will be of limited value for the development of compartmental models unless they are accompanied by high-quality electrophysiological data from the same cells. If the investigator assigns numerical values to the passive electrical parameters (specific membrane capacitance; C_m, specific membrane resistance; R_m, and cytoplasmic resistivity; R_i) of the model by guessing, there is no simple way of knowing how well the result will reproduce the behavior of the neuron under physiological conditions.

When electrophysiological recording is performed in parallel with MPE microscopy to develop compartmental models for investigating single neuron computation and signaling, potential artefacts associated with fixation and processing of the tissue are completely avoided. Tissue shrinkage and distortion during fixation and histological processing is often a major limitation during subsequent morphological reconstruction and can change both branch diameters, individual branch morphology, as well as the overall shape of the dendritic (and axonal) arborization [14, 30]. Live imaging in parallel with the electrophysiological recording also eliminates the need to embed the tissue which can lead to tissue compression and consequent distortions. Live imaging with MPE microscopy has the additional advantage that there is no need to remove the recording pipette before imaging (as would generally be the case for confocal microscopy). With longer-lasting electrophysiological recordings it is often difficult to remove the pipette from the cell without removing the whole cell body. This can be problematic for subsequent morphological reconstruction and compartmental modeling, especially for small neurons where the cell body constitutes a large part of the total surface area of the neuron.

6 Limitations and Comparison with other Methods

Despite the amount of work required, light microscopic reconstruction is currently unrivaled when it comes to generating databases with quantitative information for a large population of cells of a given type. Such databases can be very useful for understanding the degree of variability of different morphological properties within a

given type of neuron [12, 70]. Even a purely morphological database can be useful for judging the extent to which any individual neuron of the same kind is representative of the "typical" morphology.

Fully automatic reconstruction is not the only alternative to manual reconstruction. Semi-automatic procedures, e.g methods that involve variants of "finger pointing" with user-guided input to determine the basic skeleton and branching pattern of a neuronal tree (dendritic, axonal) have been developed. This includes both commercial solutions (e.g. Neurolucida 360) and freely available, academic software such as neuTube [40] and hxskeletonize (developed for the Amira environment; [16, 71]).

It is currently not feasible to use EM for a larger population analysis with complete volumetric reconstruction of individual neurons. So far, complete EM reconstructions of single neurons have typically been limited to center-line skeletons, e.g. for large-scale reconstructions of amacrine and bipolar cells in mouse retina [72, 73].

7 Notes

1. The workflow for both manual and (semi-)automatic reconstruction can also be applied for reconstruction of neuron morphology from confocal image stacks (with only a few minor modifications). When cells are injected with fluorescent dye using sharp (high-resistance) electrodes, either in live or fixed tissue, we generally recommend imaging with confocal microscopy as this allows the use of microscope objectives with higher NA and consequent higher resolution (both lateral and axial).
2. For structural imaging with subsequent morphological reconstruction and compartmental modeling, we have obtained good results with a pipette solution with either Alexa Fluor 488 hydrazide (50 or 100 μM) or Alexa Fluor 594 hydrazide (40 or 60 μM) as sodium salts (Thermo Fisher / Invitrogen). Additional details can be found in the chapter by Hartveit and Veruki (this volume).

8 Step-by-Step Procedure for Running the Automated Reconstruction in MATLAB

This section describes running the automated algorithm described above (for details, see [41]). All computer code required is freely available and can be downloaded from https://www.researchgate.net/profile/Bas-Jan_Zandt. Running the code requires installations of MATLAB (commercial; https://www.mathworks.com) and Fiji (freely available; https://imagej.net/Fiji). The description below assumes that the software runs under Mac OS X with Apple's version 6 (legacy) of Java.

8.1 Files Provided, Description of Folders and Subfolders

Code_reconstruction (folder with MATLAB files for the reconstruction algorithm, containing a number of subfolders).

Code (folder with files containing reconstruction code, including functions from the MATLAB toolboxes CellSegm [55] and Fast Marching [74]).

Miji (folder with files to run Fiji from MATLAB).

Xtra (folder with files from the TREES toolbox, files from the cbiNiftiRead toolbox to read/write NIFTI images and to scroll through image stacks).

Data (folder with an image stack for testing the algorithm, downsampled to 256 × 256 × 100 voxels to speed up execution and avoid long processing times).

NLMorphologyConverter (folder with the program NLMorphologyConverter for converting between different file formats for representing the morphology of reconstructed neurons; freely available at http://www.neuronland.org/NL.html).

8.2 Outline of Workflow

The overall workflow consists of three parts:

1. Installing and configuring the necessary software.

2. Running the automated reconstruction on an image stack of an AII amacrine cell acquired with MPE microscopy and converting the resulting files such that they can be

opened in Neurolucida and Neurolucida Explorer (for editing or analysis) or imported to NEURON (for compartmental modeling and simulations).

3. Optimizing reconstruction parameters and inspecting the results of the individual steps of the algorithm.

8.3 Installing and Configuring the Software

1. Running the program requires an installed version of MATLAB. The workflow described here has been tested on two different versions of MATLAB, version 2014b (running under Mac OS X 10.7.5) and version 2015b (running under Mac OS X 10.9.5). The Image Processing Toolbox and the Curve Fitting Toolbox are both required.

2. In MATLAB, go to Preferences - MATLAB - General - Java Heap Memory and set "Java Heap Size" to its maximum (4096 MB) to prevent memory issues.

3. Download and install Fiji, a distribution of ImageJ used for converting the format of the image stack and for the skeletonizatin step during reconstruction. Note: If too many plugins are installed for Fiji, this might cause MATLAB to crash. If this a problem, it can be solved by moving a number of the plugins from "/Applications/Fiji.app/plugins" to a new folder in the same path, e.g. named "plugins2".

8.4 Running the Automated Reconstruction

It is assumed that the image data are available in the form of a single image stack in the TIFF format. For the example described here, we will use the file named m999999_9_002_ch0_DS.tif (available in the *Data* folder). Note that the algorithm assumes a specific folder structure and that the subfolders containing MATLAB code and image data reside in the same parent folder. The output data will be written to a new subfolder inside this parent folder (as detailed below).

1. Start by generating a folder for the image data and a scaling file:

a. Make a new folder inside the *Code_reconstruction* folder and provide it with a name identical to that of the file containing the image stack, e.g. *m999999_9* for the example used here.

b. Copy the image stack file to the folder generated in step a.

c. For easier file handling, convert the image stack file to the NRRD format by opening it in Fiji and saving it as an *.nrrd file (menu: File / Save As / Nrrd ...). Save the file used here as *m999999_9_002_ch0_DS.nrrd* in the folder generated in step a.

d. Create a **.mat* file in MATLAB with the appropriate voxel (X × Y × Z) dimensions (same as in the image stack), i.e., 250 × 250 × 800 nm for the file used here. Open the Command Window in MATLAB and type on the command line:

```
voxel_size = [250, 250, 800]*1e-6
```

(for historical reasons, the dimension of voxel_size is in mm). In MATLAB's Workspace window, right-click on *voxel_size* and select "Save As..." to save as *voxel_size.mat* in the folder generated in step a above (*m999999_9*).

2. Generate an empty folder under *Code_reconstruction* and name it *results_m999999_9* for storing the results generated by the reconstruction algorithm. Now the reconstruction script can be run.

3. In MATLAB, open the file *Main.m* (*Code_reconstruction* / *Code* / *Main.m*).

4. Right-click the tab of *Main.m* and select Change Current Folder To / ... / Code

5. Set up the parameters for the reconstruction algorithm as follows:

a. Select the Editor tab in MATLAB's ribbon and edit *Main.m* in MATLAB's editor window. Set all parameters to the values recommended in Appendix A (tuning the parameters is described in section 8.5).

b. In *Main.m*, set the FIJIpath and FIJIscriptpath variables to the file paths where the Fiji plugins and scripts, respectively, are stored (see Appendix A for an example).

c. Optionally save *Main.m* (this will be done automatically in the next step).

6. Run *Main.m* as follows: Select the Editor tab in MATLAB's ribbon and click *Run* (green arrow head). This also automatically saves the *Main.m* file. If Step 4 above was

not executed, MATLAB will open a dialog stating that the file is not in the current directory. Respond by selecting "Change Folder".

7. MATLAB displays a dialog with a request to select the image stack (*.nrrd) file:
... / Code_reconstruction / m999999_9 / m999999_9_002_ch0_DS.nrrd
Note: this automatically selects and configures the output folders and files based on the name of the folder in which the file is contained (folder *m999999_9* for the example used here).

8. MATLAB generates a window displaying a MIP of the cell in the frontal (XY) image plane, requesting the user to draw a region of interest (ROI) containing the cell, but excluding the pipette and any spurious fluorescence. Use the computer mouse to draw a ROI loosely around the cell by clicking multiple times and try to keep spurious fluorescence "blobs" (if present) outside the ROI. Draw the ROI such that it separates the image of the recording pipette from the image of the cell (presumably the soma, as in the example used here; Fig. 4).

9. During execution, MATLAB will print information about the progress to the Command Window. Wait for the segmentation and skeletonization to finish. Note: the time required depends heavily on the size of the stack. For a stack with 1024 × 1024 × 100 voxels, the process can take several hours and the time depends both on the number of voxels and the ratio between the parameters *voxel_size* and *adaptivefiltersize* (contained in the file *Main.m*). If the algorithm takes too long, it is a good idea to downsample the image stack (e.g. in Fiji or ImageJ) and/or change the size of *adaptivefiltersize* to a smaller value.

10. The preliminary skeleton that is generated and displayed in a separate window will typically contain several spurious branches inside the soma. MATLAB prompts the user to indicate the region in which these branches should be removed. The region has to be delineated both in front (XY) and side (YZ) view. The reconstruction points that are inside the drawn ROIs will be removed. For the example cell used here, the selected ROIs should correspond to the soma in both projection planes such that only branches inside the soma will be removed.

11. After the ROIs with spurious branches have been indicated, program execution will continue and the algorithm will generate the tree, clean it, denote the soma, and export the results to SWC files. These steps take place automatically and do not require user intervention.

12. Potential warnings (printed to the Command Window) about trifurcations can be ignored.

13. When program execution has finished, MATLAB will have written four reconstruction files to the results folder (*results_m999999_9*) in addition to several *.mat files that contain the results of various intermediate steps:

m999999_9_auto.swc (contains the reconstructed branches)

m999999_9_auto_sm.swc (contains a smoothed version of the reconstructed branches)

m999999_9_auto_soma.swc (contains the reconstructed branches and a soma contour)

m999999_9_auto_sm_soma.swc (contains a smoothed version of the reconstructed branches and a soma contour)

14. The *.swc output files can be directly visualized in e.g. Vaa3D. If desired, the user can convert the SWC format files to the Neurolucida ASC format with the NLMorphologyConverter program as follows:

a. Copy the three *.swc files in the *results_m999999_9* folder to the *NLMorphologyConverter* folder.

b. Open a Terminal window (Mac OS X program).

c. Change directory (with the *cd* command) to the *NLMorphologyConverter* folder.

d. Execute the following command in the Terminal window (the character ">" is the prompt and should not be typed):

> ./NLMorphologyConverter m999999_9_auto.swc m999999_9_auto.ASC NeurolucidaASC

(repeat this step for the two other SWC format files)

15. The ASC format reconstruction files can be opened in Neurolucida, Neurolucida Explorer and NEURON. Note that in Neurolucida, morphology reconstruction files contain a single or multiple contours to represent the soma and a single or multiple

unconnected trees that represent the branching tree(s) emanating from the soma (irrespective of whether the files are saved in the DAT or ASC formats). In SWC files, branch roots (corresponding to origin nodes in Neurolucida) are denoted by setting their parent identifier to -1 (at the last entry of a given line). In principle, an SWC file can therefore contain multiple unconnected trees. However, several applications, including the TREES toolbox, that are used to generate and export the reconstruction do not allow this. Our workaround for this problem is twofold. First, for the intermediate reconstruction files that do not contain a soma contour (see step 13 above), the code connects all trees to a single point inside the soma (with the thickness of this point set to zero). Second, for the intermediate reconstruction files that contain a soma contour, the code connects all trees to a common point on the soma contour. As such, this might appear a bit odd in some of the figures generated to display the reconstruction. A more fundamental problem is that this property also prevents correctly importing the reconstruction to NEURON. Accordingly, we recommend using the ASC format reconstruction files generated with NLMorphologyConverter for this purpose.

8.5 Optimizing Reconstruction Parameters

When an image stack differs with respect to resolution, cell type, etc., from the one used in the procedure described above, the reconstruction parameters should be tuned to generate optimal results. Optimizing parameter values can be an art. The following section will be a guide to some of the tricks, with the goal of selecting optimal parameter values and settings in an informed way. The algorithm will be run step-by-step to optimize the corresponding parameters. For this, we will make use of the convenient option in MATLAB that permits the user to run individual sections of code without running the whole script. To enable this functionality, the relevant sections in the *Main.m* file are delimited by a double percentage sign (%%) and can be run by using the computer mouse to click anywhere within a section (this will highlight the

code in the corresponding section) and clicking on the icon "Run and Advance" in the MATLAB ribbon displayed for the Editor tab (see https://www.mathworks.com/help/matlab/matlab_prog/run-sections-of-programs.html).

Detailed procedure:

1. Open *Main.m* in the MATLAB editor window (as described above).

2. Use the recommended parameter values, but for the purpose of the example work flow described here, edit *Main.m* to change two parameter values (from their default values) in the section "2. Parameters" (in this way, the script will run with parameters not optimized for the example image stack):

add2threshold = 0.2;

simpleThreshold = 0.2;

3. Select the section "1. Clear previous session" by positioning the text cursor within it (left mouse click). In the MATLAB ribbon (displayed after selecting the Editor tab), click "Run and Advance" to execute the section and advance to the next section. Run the sections up to and including "4. Segmentation (thresholding)".

4. Three graphs will appear with the results from the thresholding procedures. Inspect the graphs carefully. Compared to the result obtained with the default parameter values, it can be seen that many branches have been missed and the threshold values should be lowered.

5. Optimize the threshold parameters as follows:

a. In MATLAB, open the file *optimize_threshold.m* (located in the *Code* folder) that will be used to optimize the threshold.

b. Set two parameter values in the section "2. Calculate and plot result of thresholding" of this optimization script as follows:

add2threshold = 0.2;

simpleThreshold = 0.2;

c. Note that the optimization script loads results from the adaptive threshold calculations. Therefore, for a new image stack the script in *Main.m* needs to have been run, up to and including the section "4. Segmentation (thresholding)", at least once

with any set of parameters before running the optimization script (see above). In *optimize_threshold.m*, run the section "1. Initiation" followed by the section "2. Calculate and plot result of thresholding".

d. A graph displaying the results appears. The *add2threshold* parameter increases the adaptive threshold. Setting it to a high value, where 1 corresponds to the maximum image intensity, results in the weaker branches not being segmented, while setting it to a low value can result in segmentation of background / noise fluorescence.

e. Note that the simple thresholding mainly contributes to the total result by closing the gaps that appear in the segmentation of the soma.

f. Lower the value of *add2threshold* as follows:

add2threshold = 0.1;

and run the section "2. Calculate and plot result of thresholding" again. The result already looks better.

g. Gradually reduce the value of *add2threshold* until reaching 0.005 (e.g. along the series 0.2, 0.1, 0.02, 0.005) and check the consequence for the segmentation for each value by re-executing the section code. Note that changing this parameter affects not only which branches are segmented, but also their thickness.

h. Try to find a value for *add2threshold* such that the resulting segmentation represents a reasonable compromise between segmenting too much noise and missing too many branches.

i. The image stack used in this example can be properly segmented using adaptive thresholding alone. However, we found that especially in stacks with larger cell bodies, adaptive thresholding can lead to gaps in the segmentation inside the soma and primary dendrite. In such cases, the value for the parameter *simpleThreshold* can be tuned (in a similar way as done for *add2threshold* above) to find a value for which the gaps are removed, without affecting the rest of the segmentation too much. For the tuning, gradually reduce the value of *simpleThreshold* from 1 to 0 (e.g. along the series 1, 0.7, 0.4, 0.2, 0.07, 0).

j. The last segmentation parameter that can be tuned is *Nvoxels_segment_min*. It is used to remove small and isolated segmented clusters of voxels ("blobs"). Segmented clusters containing fewer voxels than the parameter value are removed. To clearly demonstrate the effect of this parameter (do not do this for a real optimization), lower the threshold value to deliberately segment blobs of low-intensity background fluorescence, e.g. by setting

add2threshold = 0.005;

Now gradually change the parameter *Nvoxels_segment_min* (located at the beginning of the section "2. Calculate and plot result of thresholding" in *optimize_threshold.m*) from 1 to 500 (e.g. along the series 1, 10, 50, 200, 500; the default value is 20). Observe that for the smallest values of *Nvoxels_segment_min* only the smallest loose segments will be removed, whereas the larger segments will be removed when larger values are selected. Again, a compromise has to be found between removing all unwanted background and not removing any smaller fragmented neuronal segments at all.

k. Note the optimal values found for *add2threshold*, *simpleThreshold* and *Nvoxels_segment_min*. Close the file *optimize_threshold.m* and its accompanying figure window**.** Enter the optimized values for the parameters in the file *Main.m*.

l. In *Main.m*, verify that the parameter *redo_segmentation* is set to "true" and rerun *Main.m* from the top, section by section, up to and including the section "4. Segmentation (thresholding)". Verify that the resulting segmentation is reasonable.

6. Run the section "5. Segmentation (fast marching)". The result of this section depends on the parameter *NbrightPixelsSoma* which sets how many brightest pixels are initially used to determine the soma center (set to 100 by default; in section "2. Parameters"), but there is not much point in tuning this parameter, as it will not significantly influence the outcome. Inspect the results of the Fast Marching procedure after executing the code in the section "5. Segmentation (fast marching)". The various segments are displayed with different colors and are connected by the Fast Marching paths contoured with lines.

7. Run the section "6. Skeletonization". There is nothing to optimize here, as the skeletonization algorithm does not have any parameters. The result will be shown after executing the next section of code.

8. Run the section "7. Tree generation and conversion to SWC file". Here, the values for the tree parameters can be optimized:

a. Inspect the results from the skeletonization, displayed in the graph labeled "Skeletonization results" (generated after executing the code in the section "6. Skeletonization"), especially with respect to whether the mask for the soma was drawn correctly. The graph shows the skeleton with and without the voxels removed by the soma mask. Also inspect the graph labeled "Tree after cleaning and spurious branch removal" that can be rotated in 3D to see if any spurious branches remain. Note that for this example, relatively few branches have been generated inside the soma.

b. If required, redraw the mask demarcating the branches inside the soma as follows: Delete the file *somamask_manual.mat* (written to the folder *results_m999999_9*) and re-execute the section "7. Tree generation and conversion to SWC file". Note that the algorithm will always load the soma mask from this file if it exists. If an error is reported concerning the size of the mask, delete *somamask_manual.mat*.

c. In MATLAB's Command Window, type

clean_tree_scaling = 2;

and press Return and run the section "2. Parameters". Alternatively, change the value for *clean_tree_scaling* (the cleaning factor) directly in the file *Main.m*. and run the section "2. Parameters". Note that the majority of the small branches are removed, but so are many branches that seem perfectly valid. When *clean_tree_scaling* is set too low (e.g. at zero), small spurious tips that should be removed are kept. This can sometimes be hard to see in the graphs. In the MATLAB Command Window, type

sum(T_tree(tree))

to calculate and display the number of branch tips. Increase the value of the cleaning factor (e.g. along the series 0.01, 0.1, 0.5, 2) to reduce the number of tips. Decrease the

value of the cleaning factor when the cleaning process removes (too many) valid branches.

d. The balancing factor (*MST.bf*) can be varied to change the pattern of connecting branches that are close to each other. A small value for the balancing factor (e.g. 1e-3) favors short connections, whereas a larger value (e.g. 0.1) favors shorter routes to the soma. Try values ranging from 1e-4 to 1 and check the results when running the section "7. Tree generation and conversion to SWC file".

9. Run the section "8. Soma contour generation". Verify that the graph of the area of the individual soma contours, as a function of (axial) distance into the stack, looks approximately circular and that the peak is correctly detected (it is marked by a red diamond symbol in the displayed graph). Verify that the corresponding contour indeed denotes the soma in the figure where the contour is displayed on top of an intensity projection. Enlarge the value of the parameter *Dth* (set in section "2. Parameters"; the default value is 36 degrees) to make the soma rounder or reduce it to include more of the extrusions into the soma contour.

10. Finally, run the section "9. Plotting" to render a series of cylinders representing the tree displayed on top of a MIP of the cell. The figure (labeled "3D Tree above MIP"**)** can be rotated in 3D.

8.6 Additional Tips and Tricks for Running the Semi-Automatic Reconstruction

1. When it is desirable to re-execute parts of the reconstruction algorithm with different parameter values, the redo switches at the top of the section "2. Parameters" (lines 24-27) in *Main.m* can be set to "false" (the default setting is "true") to load the results of previous processing written to files on the hard drive. For example, to generate the trees with a different cleaning factor, set the redo switches for segmentation (line 24), fastmarching (line 25) and skeletonization (line 26) to "false", change the cleaning factor to the desired value and re-execute *Main.m*. This skips the long-lasting calculations corresponding to segmentation and skeletonization and can save much time during parameter tuning.

2. Note that if one section is re-executed, all the sections following it will have to be re-executed as well.

3. The files with the smoothed image (*im_cohendiff.mat*), adaptive threshold (*th.mat* and *imth.mat*) and manual masks (*neuronmask.mat, somamask3d.mat*) are always loaded from the hard drive if they exist. Delete these files from the results folder if their corresponding results need to be recalculated and/or redrawn.

8.7 Appendix: Recommended Parameter Values for Automated Reconstruction

```
%% parameters

% % If the reconstruction is rerun, redo any previous steps?
redo_segmentation = true;
redo_fastmarching = true;
redo_skeletonization = true;
redo_treegeneration = true;
plot_results = true;

% % Segmentation
Nvoxels_segment_min = 20; % Minimum nr of voxels in a segment (smaller
segments are discarded)
add2threshold = 0.05;     % Value added to adaptive threshold
adaptivefiltersize = [2.5 2.5 5]; % [um]
%adaptivefiltersize = [10 10 20];   % used for the Diadem Olfactory
Projection fibers
simpleThreshold = 0.11; % Threshold for soma and apical denrite
(fraction of maximum image intensity)
NbrightPixelsSoma = 100;

% coherence enhancing diffusion filtering parameters
cohenh.DT = 0.2;
cohenh.NITER = 15;
cohenh.KAPPA = 0.001;

% % Skeletonization
Nsample = 1;                % every Nth point in tree is sampled
clean_tree_scaling = 0.2;   % larger values denote more aggressive
cleaning
Nminpts_trees = 4;          % remove trees (branches extending from
the soma) with less then N points
MST.bf = 0.01;              % balancing factor for MST (should be
small if skeleton is accurate)
MST.maxdist = 10;           % [um], maximum distance between connected
points in MST

% % Soma contour generation
Dth = 36;          %degrees, a larger angle cuts off more branches and
makes the soma rounder
maxSomaSizeZ = 20; %[um], length of stack in z-direction to analyse:
Set to double the typical soma size.
SmoothLength = 8;  %[um], should be smaller than soma diameter: Set to
~half of typical soma diameter
```

```
% % FIJI path
FIJIpath = '/Applications/Fiji.app/plugins';
FIJIscriptpath = '/Applications/Fiji.app/scripts';
```

Acknowledgments

This research was supported by The Research Council of Norway (NFR 182743, 189662, 214216 to EH; NFR 213776, 261914 to MLV).

References

1. Waldeyer W (1891) Ueber einige neuere Forschungen im Gebiete der Anatomie des Centralnervensystems. Sonderabdruck aus der "Deutschen Medicinischen Wochenschrift, 1891, No. 44 u. ff. Georg Thieme, Leipzig
2. Shepherd GM (2016) Foundations of the Neuron Doctrine. 25th Anniversary Edition. Oxford University Press, New York
3. McKenna T, Davis J, Zornetzer SF, eds (1992) Single Neuron Computation. Academic Press, Boston
4. Cuntz H, Remme MWH, Torben-Nielsen B, eds (2014) The Computing Dendrite. From Structure to Function. Springer Series in Computational Neuroscience, vol. 11. Series eds, Destexhe A, Brette R. Springer, New York
5. Shepherd GM, Grillner S, eds (2018) Handbook of Brain Microcircuits, 2nd edn. Oxford University Press, New York
6. Golgi C (1873) Sulla struttura della sostanza grigia del cervello (Communicazione preventiva). Gazzetta Medica Italiana 33:244-246. Reprinted as: Sulla sostanza grigia del cervello, Opera Omnia, 1903, Vol. 1, Istologia Normale, pp. 91-98. Ulrico Hoepli, Milan
7. Cajal SRy (1909) Histologie du Système Nerveux de l'Homme et des Vertébrés. Vol. I. Maloine, Paris
8. Cajal SRy (1911) Histologie du Système Nerveux de l'Homme et des Vertébrés. Vol. II. Maloine, Paris
9. Segev I, Rinzel J, Shepherd GM, eds (1995) The Theoretical Foundation of Dendritic Function. MIT Press, Cambridge
10. Rall W (2016) Modeling dendrites: a personal perspective. In: Stuart G, Spruston N, Häusser M (eds) Dendrites, 3rd edn. Oxford University Press, New York, pp. 429-438
11. Mainen ZF, Sejnowski TJ (1996) Influence of dendritic structure on firing pattern in model neocortical neurons. Nature 382:363-366

12. Soltesz I (2006) Diversity in the Neuronal Machine. Order and Variability in Interneuronal Microcircuits. Oxford University Press, New York
13. Meredith GE, Arbuthnott GW, eds (1993) Morphological Investigations of Single Neurons in Vitro. IBRO Handbook Series: Methods in the Neurosciences. General ed.: Smith AD. Wiley, Chichester
14. Jaeger D (2001) Accurate reconstruction of neuronal morphology. In: De Schutter E (ed) Computational Neuroscience. Realistic Modeling for Experimentalists. CRC Press, Boca Raton, pp. 159-178
15. Jacobs G, Claiborne B, Harris K (2010) Reconstruction of neuronal morphology. In: De Schutter E (ed) Computational Modeling Methods for Neuroscientists. MIT Press, Cambridge, pp. 187-210
16. Evers JF, Duch C (2014) Quantitative geometric three-dimensional reconstruction of neuronal architecture and mapping of labeled proteins from confocal image stacks. In: Bakota L, Brandt R (eds) Laser Scanning Microscopy and Quantitative Image Analysis of Neuronal Tissue. Series: Neuromethods. Springer, New York, Vol. 87:219-237
17. Cline HT (2016) Dendrite development. In: Stuart G, Spruston N, Häusser M (eds) Dendrites, 3rd edn. Oxford University Press, New York, pp. 77-94
18. Parekh R, Ascoli GA (2013) Neuronal morphology goes digital: a research hub for cellular and system neuroscience. Neuron 77:1017-1038
19. Glaser JR, Glaser EM (1990) Neuron imaging with Neurolucida -- a PC-based system for image combining microscopy. Comput Med Imaging Graph 14:307-317
20. Turner DA, Wheal HV, Stockley E, Cole H (1991) Three-dimensional reconstructions and analysis of the cable properties of neurons. In: Chad J, Wheal H (eds) Cellular Neurobiology. A Practical Approach. IRL Press at Oxford University Press, Oxford, pp. 225-246
21. Meijering E (2010) Neuron tracing in perspective. Cytometry A 77:693-704

22. Horikawa K, Armstrong WE (1988) A versatile means of intracellular labeling: injection of biocytin and its detection with avidin conjugates. J Neurosci Meth 25:1-11

23. Kita H, Armstrong W (1991) A biotin-containing compound *N*-(2-aminoethyl)biotinamide for intracellular labeling and neuronal tracing studies: comparison with biocytin. J Neurosci Meth 37:141-150

24. Dumitriu D, Rodriguez A, Morrison JH (2011) High-throughput, detailed, cell-specific neuroanatomy of dendritic spines using microinjection and confocal microscopy. Nat Prot 6:1391-1411

25. Blackman A, Grabuschnig S, Legenstein R, Sjöström PJ (2014) A comparison of manual reconstruction from biocytin histology or 2-photon imaging: morphometry and computer modeling. Front Neuroanat 8:65

26. Murphy DB, Davidson MW (2013) Fundamentals of Light Microscopy and Electronic Imaging, 2nd edn. Wiley-Blackwell, Hoboken

27. Denk W, Strickler JH, Webb WW (1990) Two-photon laser scanning fluorescence microscopy. Science 248:73-76

28. Denk W (2011) Introduction to multiphoton-excitation fluorescence microscopy. In: Yuste R (ed and series ed) Imaging. A Laboratory Manual. Cold Spring Harbor Laboratory Press, Cold Spring Harbor, pp. 105-110

29. Tashiro A, Aaron G, Aronov D, Cossart R, Dumitriu D, Fenstermaker V, Goldberg J, Hamzei-Sichani F, Ikegaya Y, Konur S, MacLean J, Nemet B, Nikolenko V, Portera-Cailliau C, Yuste R (2006) Imaging brain slices. In: Pawley JB (ed) Handbook of Biological Confocal Microscopy, 3rd edn. Springer, New York, pp. 722-735

30. Groh A, Krieger P (2011) Structure-function analysis of genetically defined neuronal populations. In: Helmchen F, Konnerth A (eds), Yuste R (series ed) Imaging in Neuroscience. A Laboratory Manual. Cold Spring Harbor Laboratory Press, Cold Spring Harbor, pp. 377-386

31. Zandt B-J, Veruki ML, Hartveit E (2018) Electrotonic signal processing in AII amacrine cells: compartmental models and passive membrane properties for a gap junction-coupled retinal neuron. Brain Struct Funct 223:3383-3410

32. Major G (2001) Passive cable modeling -- a practical introduction. In: De Schutter E (ed) Computational Neuroscience. Realistic Modeling for Experimentalists. CRC Press, Boca Raton, pp. 209-232

33. Holmes WR (2010) Passive cable modeling. In: De Schutter E (ed) Computational Modeling Methods for Neuroscientists. MIT Press, Cambridge, pp. 233–258

34. Donohue DE, Ascoli GA (2011) Automated reconstruction of neuronal morphology: an overview. Brain Res Rev 67:94-102

35. Acciai L, Soda P, Iannello G (2016) Automated neuron tracing methods: an updated account. Neuroinform 14:353-367

36. Losavio BE, Liang Y, Santamaría-Pang A, Kakadiaris IA, Colbert CM, Saggau P (2008) Live neuron morphology automatically reconstructed from multiphoton and confocal imaging data. J Neurophysiol 100:2422-2429

37. Cuntz H, Forstner F, Borst A, Häusser M (2010) One rule to grow them all: a general theory of neuronal branching and its practical application. PLoS Comput Biol 6:1-14

38. Cuntz H, Forstner F, Borst A, Häusser M (2011) The TREES toolbox -- probing the basis of axonal and dendritic branching. Neuroinform 9:91-96

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

40. Feng L, Zhao T, Kim J (2014) neuTube 1.0: A new design for efficient neuron reconstruction software based on the SWC format. eNeuro DOI: 10.1523/ENEURO.0049-14

41. Zandt B-J, Losnegård A, Hodneland E, Veruki ML, Lundervold A, Hartveit E (2017) Semi-automatic 3D morphological reconstruction of neurons with

densely branching morphology: Application to retinal AII amacrine cells imaged with multi-photon excitation microscopy. J Neurosci Meth 279:101-118

42. Neher E (1992) Correction for liquid junction potentials in patch clamp experiments. In: Rudy B, Iverson LE (eds) Ion Channels. Series: Methods in Enzymology. Academic Press, San Diego, Vol. 207:123-131
43. Heintzmann R (2006) Band limit and appropriate sampling in microscopy. In: Celis JE (ed) Cell Biology. A Laboratory Handbook. Elsevier, Amsterdam, Vol. 3:29-36
44. Zandt B-J, Liu JH, Veruki ML, Hartveit E (2017) AII amacrine cells: quantitative reconstruction and morphometric analysis of electrophysiologically identified cells in live rat retinal slices imaged with multi-photon excitation microscopy. Brain Struct Funct 222:151-182
45. Pologruto TA, Sabatini BL, Svoboda K (2003) ScanImage: flexible software for operating laser scanning microscopes. Biomed Eng Online 2:13
46. Dorostkar MM, Dreosti E, Odermatt B, Lagnado L (2010) Computational processing of optical measurements of neuronal and synaptic activity in networks. J Neurosci Meth 188:141-150
47. Cannell MB, McMorland A, Soeller C (2006) Image enhacement by deconvolution. In: Pawley JB (ed) Handbook of Biological Confocal Microscopy, 3rd edn. Springer, New York, pp. 488-500
48. van der Voort HTM, Strasters KC (1995) Restoration of confocal images for quantitative image analysis. J Microscopy 178:165-181
49. Scorcioni R, Polavaram S, Ascoli GA (2008) L-measure: a web-accessible tool for the analysis, comparison and search of digital reconstructions of neuronal morphologies. Nat Prot 3:866-876
50. Wouterlood FG, Beliën JAM (2014) Translation, touch, and overlap in multi-fluorescence confocal laser scanning microscopy to quantitate synaptic connectivity. In: Bakota L, Brandt R (eds) Laser Scanning Microscopy and

Quantitative Image Analysis of Neuronal Tissue. Series: Neuromethods. Springer, New York, Vol. 87:1-36

51. Cox G, Sheppard CJR (2004) Practical limits of resolution in confocal and non-linear microscopy. Microsc Res Tech 63:18-22

52. Sigal YM, Speer CM, Babcock HP, Zhuang X (2015) Mapping synaptic input fields of neurons with super-resolution imaging. Cell 163:493-505

53. Tsukamoto Y, Omi N (2013) Functional allocation of synaptic contacts in microcircuits from rods via rod bipolar to AII amacrine cells in the mouse retina. J Comp Neurol 521:3541-3555

54. Sethian JA (1996) A fast marching level set method for monotonically advancing fronts. Proc Natl Acad Sci USA 93:1591-1595

55. Hodneland E, Kögel T, Frei DM, Gerdes HH, Lundervold A (2013) CellSegm -- a MATLAB toolbox for high-throughput 3D cell segmentation. Source Code Biol Med 8:16

56. Schindelin J, Arganda-Carreras I, Frise E, Kaynig V, Longair M, Pietzsch T, Preibisch S, Rueden C, Saalfeld S, Schmid B, Tinevez JY, White DJ, Hartenstein V, Eliceiri K, Tomancak P, Cardona A (2012) Fiji: an open-source platform for biological-image analysis. Nat Methods 9:676-682

57. Weickert J (1997) A review of nonlinear diffusion filtering. In: ter Haar Romeny B, Florack L, Koenderink J, Viergever M (eds) Scale-Space Theory in Computer Vision. Series: Lecture Notes in Computer Science. Springer, Berlin, Vol. 1252: 3-28

58. Cannon RC, Turner DA, Pyapali GK, Wheal HV (1998) An on-line archive of reconstructed hippocampal neurons. J Neurosci Meth 84:49-54

59. Lee T-C, Kashyap RL, Chu C-N (1994) Building skeleton models via 3-D medial surface/axis thinning algorithms. CVGIP: Graph Models Image Process 56:462-478

60. Prim RC (1957) Shortest connection networks and some generalizations. Bell Syst Technol J 36:1389-1401

61. Paglieroni DW (1992) Distance transforms: properties and machine vision applications. CVGIP: Graph Models Image Process 54:56-74

62. Maurer CR, Qi R, Raghavan V (2003) A linear time algorithm for computing exact Euclidean distance transforms of binary images in arbitrary dimensions. IEEE Trans Pattern Analysis and Machine Intelligence 25:265-270

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

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

65. Peng H, Bria A, Zhou Z, Iannello G, Long F (2014) Extensible visualization and analysis for multidimensional images using Vaa3D. Nature Prot 9:193-208

66. Koch, C., Segev, I (2000) The role of single neurons in information processing. Nat Neurosci 3 Suppl:1171-1177

67. De Schutter E, Steuber V (2001) Modeling simplex and complex active neurons. In: De Schutter E (ed) Computational Neuroscience: Realistic Modeling for Experimentalists. CRC Press, Boca Raton, pp. 233-257

68. De Schutter E, van Geit W (2010) Modeling complex neurons. In: De Schutter E (ed) Computational Modeling Methods for Neuroscientists. MIT Press, Cambridge, pp. 259-283

69. Carnevale NT, Hines ML (2006) The NEURON Book. Cambridge University Press, Cambridge

70. Schneider CJ, Cuntz H, Soltesz I (2014) Linking macroscopic with microscopic neuroanatomy using synthetic neuronal populations. PLoS Comput Biol 10:e1003921

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

72. Helmstaedter M, Briggman KL, Denk W (2011) High-accuracy neurite reconstruction for high-throughput neuroanatomy. Nat Neurosci 14:1081-1088

73. Helmstaedter M, Briggman KL, Turaga SC, Jain V, Seung HS, Denk W (2013) Connectomic reconstruction of the inner plexiform layer in the mouse retina. Nature 500:168-174

74. Kroon DJ (2009) Accurate Fast Marching toolbox for MATLAB. www.mathworks.com/matlabcentral/fileexchange/ 24531-accurate-fast-marching

Figure Captions

Fig. 1 Digital deconvolution of multiphoton excitation (MPE) microscopic images of dye-filled AII amacrine cells. (**a**) Subregion of individual image slice (average of two individual frames) with details of arboreal dendrites of an AII amacrine cell. Straight line (length 4.4 μm) across process used to create intensity profiles displayed in (**h**). Note how image is affected by noise and blurring. (**b**-**g**) Same image as in (**a**) after deconvolution with different settings for the signal-to-noise ratio (SNR) in the deconvolution software, as indicated in (**h**). Note how deconvolution reduces noise and blurring and how increasing the SNR progressively improves the images, but eventually leads to spatial fragmentation, most pronounced in (**f**) and (**g**). (**h**) Spatial intensity profiles of raw image (**a**) and deconvolved images (**b**-**g**) for different values of SNR during deconvolution. Note noisy profile from raw image, reaching a peak intensity of approximately 200 (thick black line) and how increasing the SNR (**b**-**d**; colored lines) increases the peak intensity from approximately 250 to 350-400. For SNR of 20 (**e**), the intensity profile reaches an overall maximum while still remaining relatively smooth (thick red line). For SNRs of 40 (**f**) and 80 (**g**), the profiles become noisy, corresponding to the spatial fragmentation seen in the images (**f** and **g**). Brightness, contrast and gamma settings were identical for all panels (**a**-**g**). Bar, 2 μm (**a**-**g**). Adapted from Zandt et al. 2017 [44]

Fig. 2 Workflow for MPE microscopic imaging and quantitative morphological reconstruction of dye-filled AII amacrine cells. (**a**) Maximum intensity projection of raw image stack of AII amacrine cell filled with Alexa Fluor 594 during whole-cell recording (dye-filled pipette attached to the cell body) overlaid on image of retinal slice acquired with IR-laser scanning gradient contrast microscopy. (**b**) Same as in (**a**), but after deconvolution. (**c**) Shape plot generated from computerized morphological reconstruction of cell in (**a**, **b**). Brightness and contrast of background image of retina had to be re-adjusted for composite images in (**a**-**c**). (**d**) Shape plot of reconstructed cell

showing details of dendritic arborization. (**e**) Three-dimensional view of morphological reconstruction. Bar, 10 μm (**a**-**d**). Adapted from Zandt et al. 2017 [44]

Fig. 3 Overview of the algorithm and work flow for automatic reconstruction of neuronal morphology. Each box summarizes a series of steps that together constitute a processing stage, the name of which is indicated at the upper left corner of each box. The user is prompted for input twice during the procedure (indicated by "user input" in the work flow). Abbreviations: maximum intensity projection (MIP); three dimensional (3D); minimum spanning tree (MST). Reproduced from Zandt et al. 2017 [41] with permission from Elsevier

Fig. 4 Workflow for MPE microscopic imaging and preprocessing during morphological reconstruction of dye-filled AII amacrine cells. (**a**) Maximum intensity projection of raw image stack of cell filled with Alexa 594 during whole-cell recording (dye-filled pipette attached to the cell body). (**b**) Same as in (**a**), but after deconvolution and alignment. Areas with increased background fluorescence caused by leakage of fluorescent dye are circumscribed by dashed lines. (**c**) Same as in (**b**), but after removal of fluorescence corresponding to pipette and contaminating areas. Continuous line corresponds to manually delineated region containing the cell, outside of which fluorescence was removed. To enhance visibility, contrast in all panels was increased by 50%, leading to saturation of areas with higher intensity. Bar, 10 μm (**a**-**c**). Reproduced from Zandt et al. 2017 [41] with permission from Elsevier

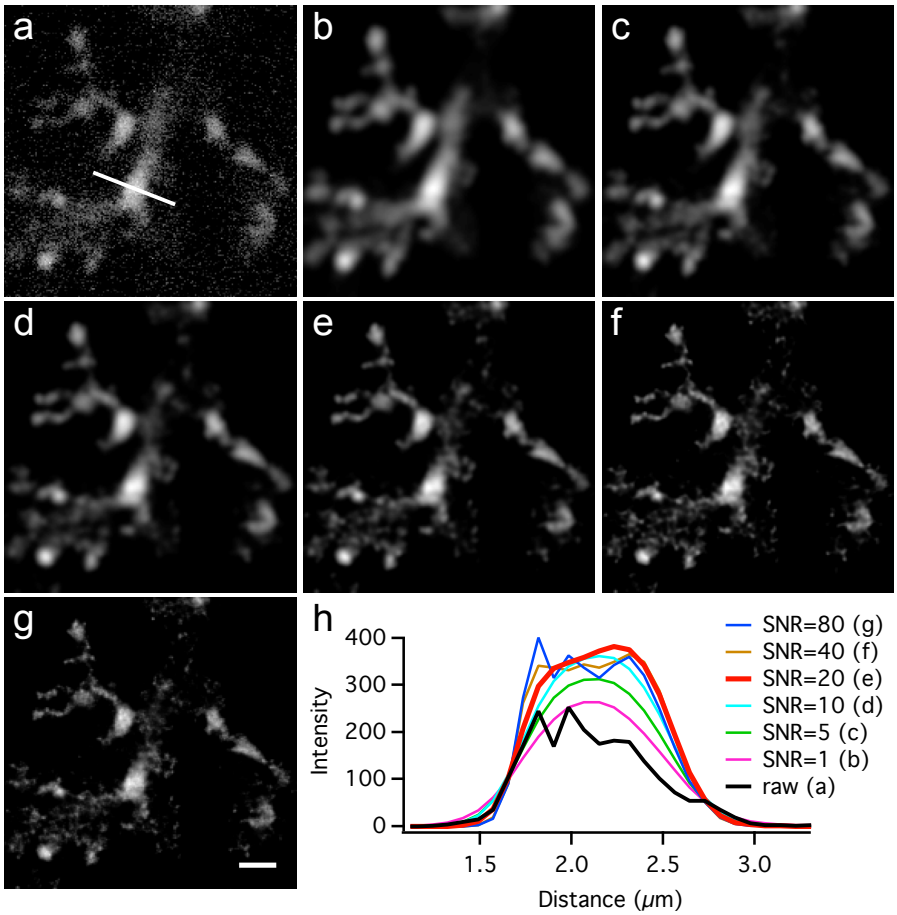

Fig. 1 (Hartveit et al.)

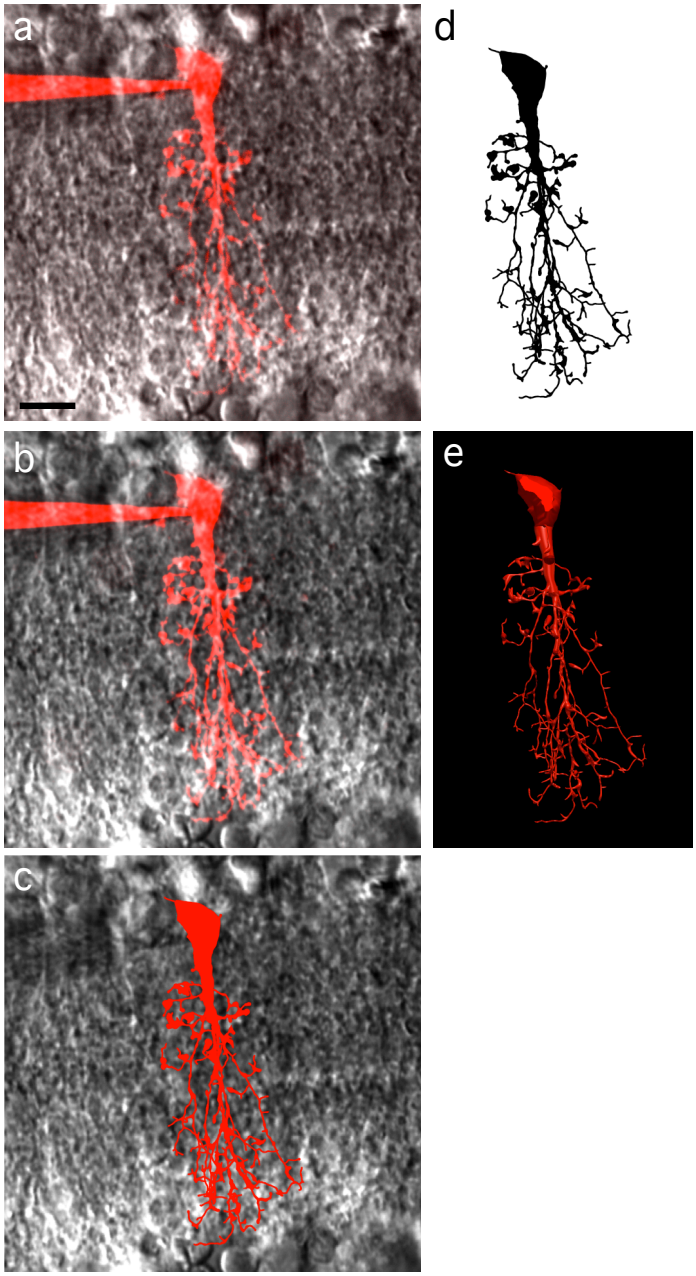

Fig. 2 (Hartveit et al.)

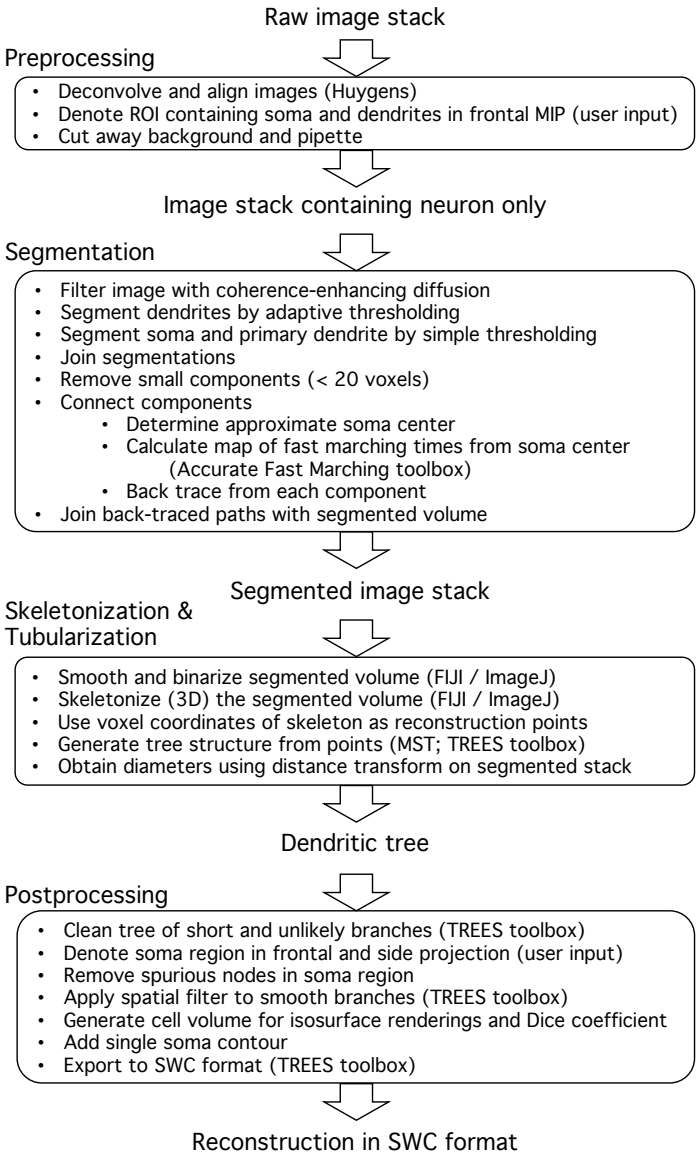

Fig. 3 (Hartveit et al.)

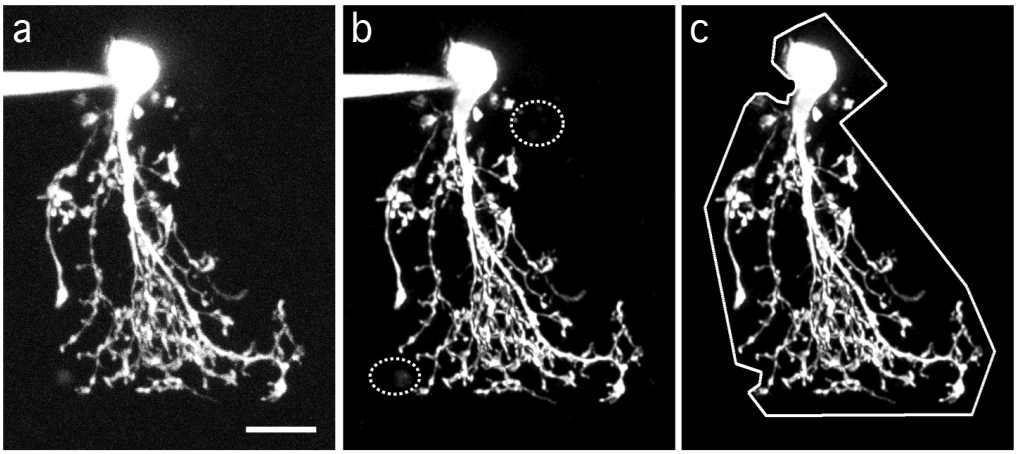

Fig. 4 (Hartveit et al.)