Methods for Automated Neuron Image Analysis

Miroslav Radojević

Colophon

This book was typeset by the author using LaTeX2$_\varepsilon$. The main body of the text was set using a 10-points Computer Modern Roman font. All graphics and figures were created using Inkscape (free and open-source) or Autodesk ®Graphic and included in the book as Portable Document Format (PDF).

Cover design by the author. The cover graphics symbolizes Sisyphus building a neuron tree. We all do Sisyphean work now and then.

Nederlandse Organisatie
voor Wetenschappelijk Onderzoek

The research described in this thesis was carried out at the Erasmus MC - University Medical Center Rotterdam, the Netherlands. This work was funded by the Netherlands Organization for Scientific Research (NWO) through grant 612.001.018.

Financial support for the publication of this thesis was provided by the Department of Radiology of Erasmus MC - University Medical Center Rotterdam and the Erasmus University Rotterdam, the Netherlands.

ISBN 978-94-6361-204-3

Printed by Optima Grafische Communicatie B.V. Rotterdam

Methods for Automated Neuron Image Analysis

Methoden voor geautomatiseerde beeldanalyse van neuronen

Proefschrift

ter verkrijging van de graad van doctor aan de
Erasmus Universiteit Rotterdam
op gezag van de
rector magnificus

Prof.dr. R.C.M.E. Engels

en volgens besluit van het College voor Promoties.

De openbare verdediging zal plaatsvinden op
dinsdag 29 januari 2019 om 13:30 uur

door

Miroslav Radojević

geboren te Užice, Servië

Erasmus University Rotterdam

Promotiecommissie

Promotor:	**Prof.dr. W.J. Niessen**
Overige leden:	**Prof.dr.ir. P.H.E. Tiesinga**
	Prof.dr.ir. F.J. Verbeek
	Prof.dr. C.I. de Zeeuw
Copromotor:	**Dr.ir. H.W. Meijering**

Contents

List of Abbreviations

APP	All-path prunning. 8
APP2	All-path prunning, version 2. 8
BoW	Bag-of-words. 95
CNN	Convolutional neural network. 106
GFP	Green fluorescent protein. 91
GLMNET	Generalized linear model via penalized maximum likelihood. 97
HCA	High-content analysis. 90
HOG	Histogram of oriented gradients. 106
KNN	K-nearest neighbor. 97
LASSO	Least absolute shrinkage and selection operator. 97
MST	Minimum spanning tree. 8
NP	Neuron pinpointer. 16
PF	Particle filter. 9
PHD	Probability Hypothesis Density. 12, 45
PSF	Point spread function. 4
RF	Random forest. 90, 97
RFS	Random finite set. 46
SIFT	Scale invariant feature transform. 91
SMC	Sequential Monte Carlo. 9, 12
SNR	Signal to noise ratio. 4
SPM	Spatial pyramid matching. 106
SVM	Support vector machine. 90

SWC	Open-source neuron reconstruction format (Stockley, Wheal, and Cole). 10
ZNCC	Zero-mean normalized cross-correlation. 6

Chapter One

Introduction

1.1 Need for neuron reconstruction

Fascination with neuronal cells dates back to the pioneering investigation over a century ago when a glance at a sample of silver-stained brain tissue disclosed the intricate network that forms the very essence of the nervous system. Remarkable milestone drawings of Santiago Ramón y Cajal [214, 261] – the founding father of neuroscience – remain as vivid as the digital images acquired by the newest fluorescence microscope. Ever since his breakthrough work, numerous studies [16,67,68,100,165,166,230,272] have explored the morphological features of neurons to gain deeper insight into their functionality, and the discipline gradually established as neuroscience [132]. In the meantime, the astounding advancement of electrical engineering and computer science laid the foundation for more specialized disciplines such as neuroinformatics and neural engineering (Fig. 1.1). Recently, attention has been directed towards brain science – the field dedicated to the investigation of the captivating mechanism of, arguably, the most complex and enigmatic organ. The disclosure of Cajal's groundbreaking *neuron doctrine*[1] brought to light the idea that the nervous system is a network composed of building blocks called neuronal cells. Each cell (Fig. 1.2) further represents a sophisticated, interconnected processing component that both transmits and processes the information. Different neuronal cells have different roles, and hence varying properties, including morphology, which offers useful evidences related to their functionality.

In the endeavor to understand neuron behavior and unravel the underlying principles, knowledge of neuronal cell morphology is essential for performing specialized analyses, which typically comprise the examination of changes in neuronal structure caused by external stimuli [106, 138], modeling [16], statistical analysis [199, 220], describing connectivity patterns [130], branching patterns [271], cataloging neuron phenotypes [69], classifying neuron types [11], or simulating electrophysiological behavior. The morphology of a neuron can be captured to a high level of detail using microscopic imaging, but many studies require a more explicit representation than offered by the resulting images, emphasizing the need for digital reconstruction of the morphology from the images into a tree-like graph structure. For example, the widely used Sholl analysis [242], commonly employed for quantification of the morphological characteristics of neurons [99], is based on neuron reconstructions as a blue print of

[1] Every neuron in the brain is a separate unit. Neurons conduct information in a defined direction and communicate across their synapses [105].

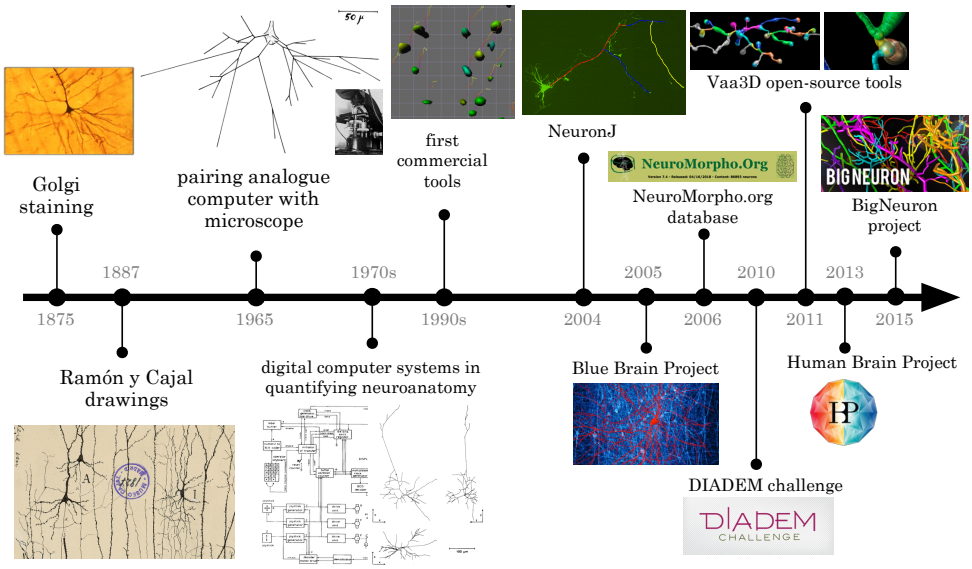


Figure 1.1: Timeline overview of selected accomplishments related to neuron analysis. Advances in electrical engineering and computer science have been crucial for the great achievements in recent decades.

morphology. And numerous other neuroscientific studies directly rely on accurate knowledge of neuronal morphology in the form of digital reconstructions [115, 188]. This makes neuron reconstruction from microscopic images a highly important technical problem in the digital era of neuroscience [194].

Neuron reconstruction methodology has coevolved with technological advances in computing and imaging. With the early analogue computers [103], it became possible to connect a computer with a microscope and use analogue linear motion transducers to significantly speed up the gathering of information about the dendritic and axonal patterns. Subsequent generations of digital computers [45, 46] brought further advances to neuron reconstruction by mixing computer graphics with the neuron image and using advanced operator controls such as a 3D joystick. The introduction of desktop PC hardware and software in the late decades of the 20th century increased the impact of computers [115], which by then were powerful enough to implement algorithms that, although previously developed, could not be directly run and shared between the users on a needed scale. It also marked the period when the first commercial and open-source academic software tools emerged (Fig. 1.2). But with the ever growing amount and complexity of data, the quest for better solutions to neuron reconstruction continued. At present, even though many methods have been published and many tools are available, neurobiologists often still resort to manual or interactive approaches to get satisfactory results, indicating that reliable automated neuron reconstruction is a major challenge.

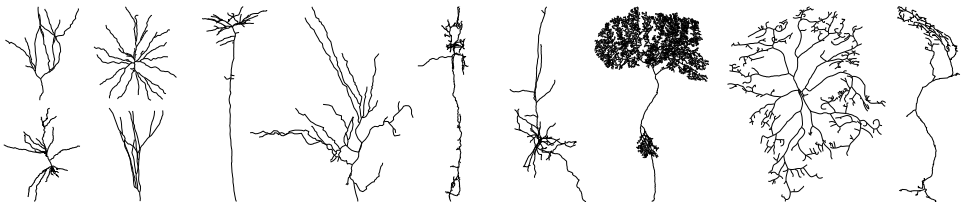


Figure 1.2: Example neuronal arbors from NeuroMorpho.org showcase the morphological diversity across species, brain regions, and laboratories worldwide. The renderings were exported using the web-based neuron morphology viewer [18].

1.2 Challenges in neuron reconstruction

The two crucial engineering tasks in neuron reconstruction are 1) reducing the time needed to obtain the reconstruction compared to manual delineation and 2) emulating or possibly surpassing the accuracy of the reconstruction compared to manual delineation. The astonishing increase in volume of microscopic imaging data rules out manual processing [170] and requires automated solutions that are fast enough to ensure high throughput in neuron analysis [198]. Moreover, the wide variety of neurons and microscopic imaging modalities (dark-field, bright-field, confocal) puts high demands on automated processing [193, 260], and has resulted in a plethora of reconstruction methods [192]. To date, two grand challenges have been organized in the field, namely DIADEM[2] (short for digital reconstruction of axonal and dendritic morphology) and BigNeuron[3] [102, 190, 194], which aimed to stimulate community efforts to improve the state-of-the-art of neuron reconstruction. Despite great advancements, however, both engineering tasks still remain a major challenge. For instance, a 20-fold speedup compared to manual reconstruction has been projected [154], and proof-editing of automated reconstructions is still a bottleneck [193].

In this thesis the focus is on detection and reconstruction of neurons from fluorescence microscopy images. Here, neurons are first labeled using a fluorescent dye in order to expose their structure, and then digital images are acquired using a light microscope, which are subsequently computer processed to reconstruct the morphology of the neurons. In practice, reconstruction algorithms often need to be customized to address a particular biological question or deal with a particular experimental condition, which may prevent them from being applicable across different experiments or laboratories. Key obstacles in achieving accurate and robust automation in neuron reconstruction [2, 76, 169] are the following:

1) The morphology of neurons is remarkably diverse (Fig. 1.2) across brain regions and biological species. Even to expert human observers, the structural complexity of the neuronal arbors poses a significant challenge to visual comprehension, and manual delineation may easily take hours to days per neuron. This not only underscores once again the need for automation but also indicates that automated solutions require

[2] http://diademchallenge.org
[3] http://bigneuron.org

very powerful computer vision algorithms.

2) Neurons may be imaged using a variety of microscopy imaging modalities [76, 169]. Each modality has its strengths and weaknesses, but all suffer from the diffraction limit, causing a blurring of structures below that limit as characterized by the point-spread function (PSF) of the microscope [60], which under reasonable assumptions may be approximated by the Gaussian function [303]. The diameters of dendrites and axons may vary significantly and be smaller (sub-μm) than the lateral resolution of the microscope. Moreover, the axial resolution is often even multiple times lower, further limiting the level of detail attainable in neuron reconstruction. Also, since neurons may have a relatively huge spatial extent compared to the field of view of the microscope, they are often imaged with lower magnification factors and thus larger voxel sizes, which limits the level of detail even more.

3) Eventually, all types of optical imaging are affected by photon noise [274], being a Poisson process. In accordance with earlier studies, the signal-to-noise ratio (SNR) of a microscope image is expressed as the ratio of the intensity inside a neuron above the background and the standard deviation of the noise inside the neuron [50,250]. Depending on the illumination intensity and the fluorophore labeling density and homogeneity in an experiment, the SNR level may vary significantly between images, and even within one image. In addition, images may contain other types of background "noise", such as debris.

4) The ever increasing data volumes in neuron imaging counterbalance the increasing memory and processing speed of modern computers. Existing methods for neuron reconstruction often do not scale well with image volume size in terms of required processing time and accuracy [198]. Dedicated engineering strategies to allow existing algorithms to reconstruct unlimited data volumes have been reported [198, 311], using data tiling approaches, but the challenge remains to develop algorithms that are not too adversely affected by image volume.

5) Evaluation of neuron reconstruction algorithms requires the availability of a reference or "gold standard". Reference reconstructions are typically obtained by expert manual annotation, which suffers from inter-observer and even intra-observer variability. Careful proof-editing [193] and making consensus reconstructions from multiple observers [190] to improve the gold standard is extremely time consuming and not always feasible. Alternatively, synthetic neuron images may be used [136,210,212], whose true reconstructions are known by definition, but which are inevitably less realistic and less representative of the real problem. Thus, if the ultimate goal is to outperform humans in terms of both time and accuracy, new ways of evaluating algorithms are needed that do not depend as much on humans.

1.3 Common neuron reconstruction strategies

Single-cell neuron reconstruction methods typically employ different algorithmic approaches. The vast majority of works treats reconstruction as a modular task and uses a pipeline of different algorithmic units dedicated to processing particular tree components [169]. Nevertheless, there is inevitably a degree of commonality and interdependence among methods, as a cascade of processing units requires the output of

a) Shortest path tracing b) Minimum spanning tree c) Path-pruning

Figure 1.3: Common neuron reconstruction strategies. a) Finding an optimal path between any two control points. b) Inferring the optimal tree structure from a set of landmark points serving as nodes. c) Pruning an over-completely traced tree.

one computation stage to be used in the next. Reconstruction accuracy also depends on the ability of a method to generalize well, which implies undertaking a fair share of global reasoning. The many methods introduced over the years [2, 76, 169] can be broadly cast into three categories of strategies (Fig. 1.3), each involving various optimization tasks: 1) finding an optimal path between given control points (Fig. 1.3a), 2) inferring an optimal tree from a set of given landmark points that already belong to the tree and that serve as its nodes (Fig. 1.3b), and 3) pruning an over-traced and thus over-represented tree (Fig. 1.3c). Of course, this categorization is not absolutely rigorous, and often the different strategies are combined.

Neuron branch tracing (Fig. 1.3a) can be interpreted as an energy or cost optimization problem [171, 195]. A classic example of this is to compute an optimal ("shortest") path through the voxel grid, where the given geodesic metric or cost function is defined to weigh the transition between grid elements (graph vertices). This strategy has been used in many published methods [155, 171, 196], notably for semi-automated neuron tracing, where the control points are given by the user or are (interactively) computed from the image. Notable tracing approaches are geodesic shortest path [195] or live-wire segmentation [171] solved using Dijsktra's shortest-path algorithm [73]. The downside of tracing defined in this manner is the necessity of having control points, which can be a challenge to obtain automatically and require global image reasoning. Computed traces can further be refined with energy optimization and solved using gradient descent algorithm [191, 195].

The commonly used format to store reconstruction results (Section 1.5) assumes neurons to be tree-like structures consisting of connected nodes. Each two successive nodes constitute a neuronal branch compartment. Thus the essential geometrical quantities computed during tracing are the node center coordinates $\mathrm{p} = (x, y, z)$ and corresponding local (spherical) radii r of the neuronal tree at those points. In addition the local direction $\mathrm{v} = (v_x, v_y, v_z)$ at p may be computed. Taken together, a node is represented by a vector $\mathrm{x} = [\mathrm{p}, r, \mathrm{v}]$, and a branch by a sequence of N nodes $\mathrm{X} = \{\mathrm{x}_1, \ldots, \mathrm{x}_i, \ldots, \mathrm{x}_N\}$. The cumulative cost used by the optimization scheme to steer the tracing toward an optimal solution is computed over all nodes and compartments. Aside from geometrical constraints this typically also includes constraints based on image feature values such as local intensity or tubularity:

$$C(\mathrm{X}) = \sum\nolimits_i c(\mathrm{x}_i, I(\mathrm{x}_i))\Delta\mathrm{x} \tag{1.1}$$

where the cost function c could for example be decompose into:

$$c(\mathrm{x}, I(\mathrm{x})) = \alpha c_\iota(\mathrm{x}, I(\mathrm{x})) + (1-\alpha)c_\gamma(\mathrm{x}) \tag{1.2}$$

with c_ι denoting some intensity-based cost, c_γ a geometry-based cost, and $\alpha \in [0,1]$ a weight factor. The intensity-based cost could directly use intensities:

$$c_\iota(\mathrm{x}, I(\mathrm{x})) = I(\mathrm{x}) \tag{1.3}$$

or for example the exponential of the inverse intensity [195]:

$$c_\iota(\mathrm{x}, I(\mathrm{x})) = \exp\left(\lambda(1-\tilde{I}(\mathrm{x}))^2\right) \tag{1.4}$$

where $\tilde{I}(\mathrm{x}) = \tilde{I}(\mathrm{p}) = I(\mathrm{p})/I_{\max}$, with $I_{\max} \equiv \max_{\mathrm{p}}\{I(\mathrm{p})\}$, denotes normalized intensity. A popular alternative is to use:

$$c_\iota(\mathrm{x}, I(\mathrm{x})) = \rho(\mathrm{p}) \tag{1.5}$$

where ρ denotes some tubularity measure computed from the image intensities. If computed over a set of scales, it is written ρ_Σ, with $\Sigma = \{\sigma_k\}$, $1 \leq k \leq K$, and σ_k denoting the individual scales. In particular the eigenvalues of the Hessian matrix computed at different scales are often used to measure the tubularity of local image structure [92, 171, 224, 254]. Tubularity measures are commonly computed at prefiltering stage and aim to reduce the presence of noise and non-tubular structures. They can be computed in an unsupervised manner, independent of the input data, or can be learned from the data in a supervised way [149, 247], enabling adaption to a specific application based on the training. The downside of the latter approach, aside from having to design a suitable machine learning model, is that it requires dedicated training, which in practice can be time and resource consuming.

Trace refinement can be based on the discrepancy between the node positions and the centers of the local image intensity profiles. For example, to further align a trace to the underlying image intensities, the following cost could be minimized:

$$c_\iota(\mathrm{x}, I(\mathrm{x})) = \exp\left(\lambda \frac{\sum_{\mathrm{y}\in\Theta(\mathrm{x})} \|\mathrm{x}-\mathrm{y}\|^2 I(\mathrm{y})\Delta\mathrm{y}}{\sum_{\mathrm{y}\in\Theta(\mathrm{x})} I(\mathrm{y})\Delta\mathrm{y}}\right) \tag{1.6}$$

where $\Theta(\mathrm{x})$ denotes a local spatial neighborhood of x. Such refinement boils down to image intensity based mean-shifting [52] of the node position, which can also be used to refine a collection of the overlapping tracings [210]. Alternatively, the zero-mean normalized cross-correlation (ZNCC), which is independent of intensity offsets and scalings, can be used to quantify the similarity of the local image intensity profile and a theoretical model profile for trace refinement:

$$c_\iota(\mathrm{x}, I(\mathrm{x})) = \frac{\sum\limits_{\mathrm{p}\in\Theta(\mathrm{x})} (I(\mathrm{p}) - \bar{I}(\mathrm{p}))(G_\sigma(\mathrm{p}) - \bar{G}_\sigma)}{\sqrt{\sum\limits_{\mathrm{p}\in\Theta(\mathrm{x})} (I(\mathrm{p}) - \bar{I}(\mathrm{p}))^2 \sum\limits_{\mathrm{p}\in\Theta(\mathrm{x})} (G_\sigma(\mathrm{p}) - \bar{G}_\sigma)^2}} \tag{1.7}$$

where in this case $\Theta(\mathrm{x})$ typically is a cylindrical neighborhood of node $\mathrm{x} = [\mathrm{p}, \sigma, \mathrm{v}]$, centered at p with radius σ and directed along v, and G_σ denotes the model profile at given scale σ, while $\bar{I}$ and $\bar{G}_\sigma$ are the respective mean intensity values of the image and the model inside region Θ. In microscope images of neurons a reasonable choice for G_σ is the Gaussian profile in the cross-sectional plane and the uniform profile in the axial plane of the cylindrical neighborhood [209].

The geometry-based cost in (1.2) can incorporate the vectorial (directional) alignment of nodes along the estimated tubularity direction [171]:

$$c_\gamma(\mathrm{x}) = c_\gamma(\mathrm{x}_i, \mathrm{x}_{i+1}) = \frac{1}{2}\left(\sqrt{1-\varphi(\mathrm{p}_i, \mathrm{p}_{i+1})} + \sqrt{1-\varphi(\mathrm{p}_{i+1}, \mathrm{p}_i)}\right) \tag{1.8}$$

where $\varphi(\mathrm{p}_i, \mathrm{p}_j) = |\mathrm{v}_i \cdot \mathrm{u}_{ij}|$ quantifies the alignment of the tubularity directions of two successive trace points with the unit link vector $\mathrm{u}_{ij} = (\mathrm{p}_j - \mathrm{p}_i)/\|\mathrm{p}_j - \mathrm{p}_i\|$ between them. Another example of a geometry-based cost is the trace smoothness [195] or bending energy [212] estimated using the second-order derivative:

$$c_\gamma(\mathrm{x}) = c_\gamma(\mathrm{x}_{i-1}, \mathrm{x}_i, \mathrm{x}_{i+1}) = \sum\nolimits_i (\mathrm{p}_{i-1} - 2\mathrm{p}_i + \mathrm{p}_{i+1})^2 \Delta\mathrm{p} \tag{1.9}$$

Besides Dijkstra's shortest path algorithm, numerous other approaches to optimized local branch tracing have been reported. For example, energy optimization techniques such as active contours have been used to determine the optimal parametrized skeleton [228], or to directly find the open-curve representing a branch using gradient-vector flow to optimize the energy [283]. Although path optimization methods require initialization in the form of control points, which can be a critical step, the use of active contours may refine even the initialization and improve the alignment of the control points to the local image context. The general downside of active contours, however, is the possibility of ending up with many discontinued branch segments due to gaps in low quality images. An alternative is to use the fast marching (FM) algorithm [233]. Conceptually similar to Dijkstra's shortest path algorithm, FM appears to be particularly useful for tracing curvilinear structures such as neuronal branches [24, 177, 192, 222, 273, 291], as it is able to bridge gaps. Particular implementations of FM-based neuron reconstruction, such as the APP2 method [291], have been very effective in reducing false-positives occurring in previous approaches [192, 273]. Follow-up FM-based works have reported improvements in the speed of tracing discontinuous image structures by using gradient-descent combined with reinitialization [177] and back-tracking [153].

Accurately merging control points and any computed traces between them into a tree representation (Fig. 1.3b) is a significant challenge due to the many possible alternatives to explore and evaluate. Since an exhaustive search for the optimal solution may require prohibitive amounts of computation time, the problem of assembling the many pieces of the neuron tree structure is often solved using approximations, but even so it remains an ongoing challenge [190]. A significant share of the methods aims at finding a globally optimal solution, where the set of node candidates is turned into a weighted graph, and the reconstruction problem consists in extracting the optimal tree from this graph. One prominent method is the NP-complete[4] minimum spanning

[4] Non-deterministic polynomial-time.

tree (MST) algorithm [107, 269, 292, 300]. A possible approach is to extract minimum trees spanning a subset of the graph's edges, but the solution to this problem is NP-hard [54] involving plenty of exploration scenarios, and approximative solutions are used in practice [33, 107]. An alternative [107, 269, 292] is to find a k-minimum spanning tree (k-MST) spanning a subset of the graph's nodes (the control points) as opposed to the original MST spanning all nodes. Various criteria have been used with the MST algorithm to weigh and connect graph elements, for example based on distances and intensities [300], or on probabilistic costs computed from the proximity of elements towards the middle of a filament [269].

A particularly prominent family of methods is based on the pruning strategy (Fig. 1.3c) to reconstruct neurons. Such methods start with an over-complete neuron tracing which is then gradually reduced ("pruned") to converge towards an optimal concise representation of the neuronal tree. The underlying idea is to initially capture all voxels belonging to a neuron in a graph and subsequently to iteratively discard particular graph elements using various criteria. For example, in all-path pruning (APP) [192], over-complete traces are obtained using Dijkstra's algorithm, and the iterative pruning starts with the leaf nodes of the resulting graph. It executes in linear time, ensuring maximum coverage and minimum redundancy of the underlying neuron signal. Graph nodes that are already covered by others are removed with higher priority. A similar approach was employed in the follow-up work APP2 [291], which introduced several improvements, such as the usage of a long-segment-first hierarchical pruning and dedicated preprocessing of the input image stack intended to optimize fast-marching based tracing.

1.4 Neuron reconstruction using Bayesian filtering

One of the main novelties proposed in this thesis to improve on commonly used neuron reconstruction strategies is the use of Bayesian filtering. Bayesian reasoning operates with the assumption that the states (quantities of interest) of any given system are not measurable directly but can be estimated (filtered) from observations using a recursive approach involving two essential processing steps: prediction and update [77]. More specifically, the unknown (hidden) states are expressed as a series of state vectors $\{\mathrm{x}_t; t \in \mathbb{N}, t \geq 0\}$ and are estimated in a probabilistic fashion using sequentially arriving independent observations $\{\mathrm{z}_t; t \in \mathbb{N}, t > 0\}$, leading to a posterior distribution $p(\mathrm{x}_t|\mathrm{z}_t)$ of the states. This approach has been used extensively for tracking objects in sequential data (time series) in many applications [164, 215, 223, 257] but may also be applied to the problem of tracing neuronal branches in static data (spatial images). Here, the states could be defined as the node vectors $\mathrm{x} = [\mathrm{p}, r, \mathrm{v}]$ introduced in the previous section, which are predicted and updated from one node to the next using Bayesian filtering. The actual state value $\hat{\mathrm{x}}$ of any given node may afterwards be estimated from the posterior distribution $p(\mathrm{x}|\mathrm{z})$ using for instance the maximum a-posteriori (MAP) approach or computing the centroid.

Formally, the joint posterior distribution of all state vectors up until (recursion) time t, that is $\mathrm{x}_{0:t} \equiv \{\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_t\}$, is obtained from corresponding observations,

$\mathrm{z}_{1:t} \equiv \{\mathrm{z}_1, \mathrm{z}_2, \ldots, \mathrm{z}_t\}$, by applying Bayes' theorem:

$$p(\mathrm{x}_{0:t}|\mathrm{z}_{1:t}) = \frac{p(\mathrm{z}_{1:t}|\mathrm{x}_{0:t})p(\mathrm{x}_{0:t})}{\int p(\mathrm{z}_{1:t}|\mathrm{x}_{0:t})p(\mathrm{x}_{0:t})d\mathrm{x}_{0:t}} \tag{1.10}$$

and the marginal distribution $p(\mathrm{x}_t|\mathrm{z}_{1:t})$ can be computed by recursively applying

$$\text{Prediction:} \quad p(\mathrm{x}_t|\mathrm{z}_{1:t-1}) = \int p(\mathrm{x}_t|\mathrm{x}_{t-1})p(\mathrm{x}_{t-1}|\mathrm{z}_{1:t-1})d\mathrm{x}_{t-1} \tag{1.11}$$

$$\text{Update:} \quad p(\mathrm{x}_t|\mathrm{z}_{1:t}) = \frac{p(\mathrm{z}_t|\mathrm{x}_t)p(\mathrm{x}_t|\mathrm{z}_{1:t-1})}{\int p(\mathrm{z}_t|\mathrm{x}_t)p(\mathrm{x}_t|\mathrm{z}_{1:t-1})d\mathrm{x}_t} \tag{1.12}$$

The recursion mechanism uses a transition prior $p(\mathrm{x}_t|\mathrm{x}_{t-1})$, also called the transition model, to predict the next hidden state distribution from the previous, which allows to incorporate prior knowledge about state dynamics. And the predicted states are subsequently updated using the likelihood $p(\mathrm{z}_t|\mathrm{x}_t)$, also called the observation model, which allows to incorporate prior knowledge about state appearance. Equations (1.11) and (1.12) can be rewritten more concisely as:

$$p(\mathrm{x}_t|\mathrm{z}_{1:t}) \propto p(\mathrm{z}_t|\mathrm{x}_t) \int p(\mathrm{x}_t|\mathrm{x}_{t-1})p(\mathrm{x}_{t-1}|\mathrm{z}_{1:t-1})d\mathrm{x}_{t-1} \tag{1.13}$$

By custom design of the two models involved, the Bayesian filtering framework can be made to work for any given application, making it a fairly universal tool.

Several well-known analytical solutions to Bayesian filtering, such as the famous Kalman filter (KF) or hidden Markov model (HMM) filter, are based on mathematical assumptions that turn out to be very limiting in practice. For instance, the closed-form solution in the case of KF is based on the assumption that the models are linear and the distributions are Gaussian, which may not be accurate. Numerous other approximation schemes (extended KF, grid-based filters, et cetera) have been proposed with varying accuracy, practical feasibility, and computational cost. A popular alternative is sequential Monte Carlo (SMC) filtering [12], in the literature also referred to as particle filtering (PF), which has been used for a wide range of applications and is also the method of choice in this thesis. It approximates the posterior using discrete samples or particles[5] and corresponding weights:

$$\left\{\mathrm{x}_t^{(i)}, w_t^{(i)}; i = 1, \ldots, N\right\} \tag{1.14}$$

so that the possibly intractable calculation of (1.13) for non-linear and/or non-Gaussian cases is made tractable again. Here, N denotes the number of samples, and using more samples generally leads to better approximation:

$$p(\mathrm{x}_t|\mathrm{z}_{1:t}) \approx \sum_{i=1}^{N} w_t^{(i)} \delta\big(\mathrm{x}_t - \mathrm{x}_t^{(i)}\big) \tag{1.15}$$

[5] These terms are used interchangeably in the SMC-related content of this thesis.

During SMC filtering, the weights are updated using a process known as sequential importance sampling (SIS), involving an importance function π:

$$w_t^{(i)} \propto w_{t-1}^{(i)} \frac{p(\mathrm{z}_t|\mathrm{x}_t^{(i)})p(\mathrm{x}_t^{(i)}|\mathrm{x}_{t-1}^{(i)})}{\pi(\mathrm{x}_t^{(i)}|\mathrm{x}_{0:t-1}^{(i)}, \mathrm{z}_{1:t})} \tag{1.16}$$

One problem with SIS is the possibility of weight deterioration over time, leading to particle degeneration and a very sparse particle distribution, as a consequence of which the approximation fails to accurately represent the posterior. This can be avoided by regular resampling of the posterior distribution so that particles having higher importance weights remain more numerous.

1.5 Neuron reconstruction tools and formats

Reconstruction software is nowadays implemented in a wide range of programming languages on different computer platforms and distributed over all regularly used operating systems [2,169]. The work presented in this thesis focuses on Java implementation within the ImageJ platform [1,155,200] and on C++ implementation for the Vaa3D platform [189]. Both platforms are widely used in bioimage analysis and thus facilitate deployment of the proposed methods. The latter is also the base platform for the BigNeuron project and provides various means to evaluate and benchmark neuron reconstruction methods. Evaluation is commonly carried out by comparing the reconstructed neuron tree with a gold standard reconstruction. In other words, measuring the performance of a neuron reconstruction method boils down to quantifying the resemblance of two graphs, one resulting from the method and the other typically from manual annotation of the image data. Resemblance is often computed in terms of internodal spatial distances but can also be based on a comparison of quantitative features computed from the graphs using tools such as L-Measure [231] and BlastNeuron [282].

The most widely accepted format to store, exchange, and compare reconstructed neuron trees is the SWC format [44], named after Stockley, Wheal, and Cole, who first described it [256]. It is an open-source space-delimited text format that stores tree structures as a list of nodes, $\mathcal{N} = \{n_1, \ldots, n_i, \ldots, n_j, \ldots\}$, where each node contains certain properties related to the underlying neuronal geometry and topology. More specifically, each node in the SWC format, being a single line in the SWC file, contains seven attributes: node index identifier i, node type (soma, dendrite, axon, et cetera), three spatial coordinates (x_i, y_i, z_i), radius r_i, and a parent node index (Fig 1.4). By convention the latter is set to -1 for the origin node. To conform with a tree structure, each node n_j may have only one parent n_i, with lower node index $(i < j)$. Loops and disconnected branches should not exist as that would violate the tree-like structure. Even though there exist some variations of the SWC format, especially concerning the definition of the soma node[6,7,8] for which the simple spherical model may not be

[6] http://research.mssm.edu/cnic/swc.html
[7] http://www.neuronland.org/NLMorphologyConverter/MorphologyFormats/SWC/Spec.html
[8] NeuroMorpho.org FAQ: What is SWC format? http://www.neuromorpho.org/myfaq.jsp

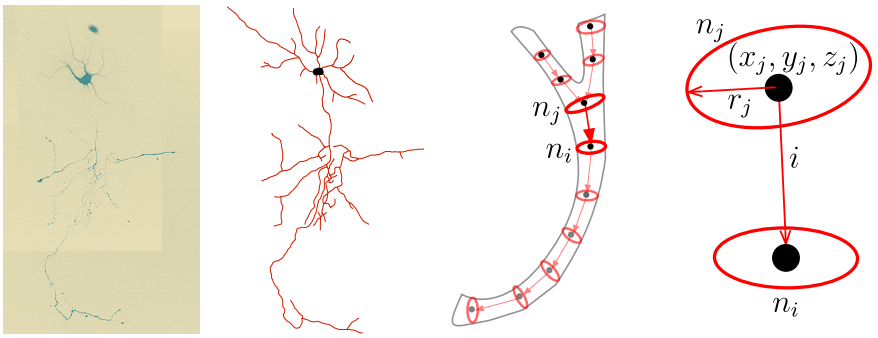


Figure 1.4: Neuron morphology representation from image to digital reconstruction data format. a) Single neuron image. b) Digital reconstruction of the neuron exported using the web-based neuron morphology viewer [18]. c) Illustration of the SWC format as a list of nodes n_i. d) Detailed visualization of one segment (truncated cone) constituted by two connected nodes (n_i, n_j) and basic node attributes.

suitable [18], it is straightforward to implement and use, which explains its widespread adoption in many neuron morphology projects [15,190]. Some reconstruction methods are even based directly on the SWC format [86]. The work described in this thesis uses exclusively the SWC format.

1.6 Thesis outline and contributions

This thesis addresses the need for further automation of neuronal image analysis. It presents methodological contributions and experimental results obtained while investigating computer vision solutions to various problems in extracting useful information from microscope images of single neuronal cells. The chapters propose original approaches to problems such as the detection of landmarks (critical points) of the neuronal tree, complete tracing and reconstruction of the tree, and the detection of regions containing neurons in high-content screens. Each chapter describes the proposed method, its software implementation, and evaluation. The remainder of this section gives a brief outline of the thesis and its contributions.

In Chapter 2 a new method is presented to detect critical points such as junctions and terminations of the neuronal tree in the images. A junction is loosely defined as a point where three branches merge, and a termination corresponds to a point where a branch ends. The proposed detection method uses directional filtering to extract essential features at every point in an image which are stored in linguistic terms and subsequently processed using fuzzy logic and rule-based reasoning to classify the point as background or a specific type of critical point. A custom-tailored set of IF-THEN rules is proposed for this purpose. The very concept of fuzzy sets allows points to

have partial class memberships and thus helps to cope with uncertainty in the data. Defining a set of rules that can be truthful *to a degree* is a convenient approach to fusing information into an aggregated decision.

In Chapter 3 the Bayesian filtering framework is harnessed in a novel way to perform tracing of neuronal branch centerlines. The proposed method is based on probability hypothesis density (PHD) filtering to allow simultaneous tracing of an a priori unknown number of branches of the neuronal arbor. This specific type of filtering extends probabilistic tracing to the next generalization level and is implemented using SMC estimation. Compared to other works the present application differs fundamentally in the sense that the filtering is applied in space instead of in time. To make this work, new transition and observation models are proposed, incorporating prior knowledge about the shape and appearance of neurons in microscope images. Since the method is probabilistic, multiple runs may yield slightly different traces, providing accumulating evidence about the neuronal structures. This is helpful in the case of structural ambiguities in the images. The experimental results show this strategy indeed leads to more accurate tracings.

In Chapter 4 the probabilistic tracing idea is extended to a new automatic method for full neuron reconstruction. The method, named probabilistic neuron reconstructor (PNR), starts by identifying regions in the image that are highly likely to be neuronal branches according to a given tubularity measure, and seed points are then extracted from these regions to initiate the tracing process. Continuing the line of thought in the previous chapter, the method uses SMC estimation to perform probabilistic over-tracing of the neuronal arbor, resulting in multiple traces per branch to accumulate evidence about their morphology, including local branch diameters. Since the very idea of over-tracing is to collect statistically independent pieces of evidence, the tracing process from each seed point can be run independently, which in principle allows for straightforward parallelization to speed up the computations. To obtain the full reconstruction, the traces are refined and grouped into a single trace per branch, from which a tree representation is obtained using breadth-first search. An early version of the PNR method was a contender in the BigNeuron benchmark, where it already performed quite well. The method presented in this chapter is a substantially redesigned and improved version of it.

Lastly, in Chapter 5, a feasibility study is presented of detecting neurons in high-content images of cell cultures from screening studies. The considered task is typically the first step in a high-throughput analysis pipeline, where regions containing neurons are detected in large image mosaics of low resolution but wide field-of-view, which can subsequently be scanned in high resolution for further morphological analysis using methods such as presented in the previous chapters. The detection is performed using supervised machine-learning methods trained on a very large number of image features. The considered machine-learning methods include support vector machines, random forests, k-nearest neighbors, and generalized linear model classifiers, and the image features are extracted using the compound hierarchy of algorithms representing morphology (CHARM) and the scale-invariant feature transform (SIFT). The experimental results indicate that a random forests classifier using the right feature subset ranks best but is not statistically significantly better than some of the support-vector machine based classifiers. A pilot with deep-learning based artificial neural networks

shows that the traditional classification methods are yet to be preferred given the limited available data and annotations.

Chapter Two

Fuzzy-logic based detection and characterization of junctions and terminations in fluorescence microscopy images of neurons

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. In this chapter, a fully automatic method for the integrated detection and characterization of both types of critical points in fluorescence microscopy images of neurons is presented. In view of the majority of the current studies (currently available to the authors), which are based on cultured neurons, the method was described and evaluated 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 the presented method. A comparison with the output of two existing neuron reconstruction methods reveals that the method described in this chapter achieves substantially higher detection rates and could provide beneficial information to the reconstruction process.

Based upon: M. Radojević, I. Smal, E. Meijering, "Fuzzy-logic based detection and characterization of junctions and terminations in fluorescence microscopy images of neurons", *Neuroinformatics*, vol. 14, no. 2, pp.201-219, 2016.

2.1 Introduction

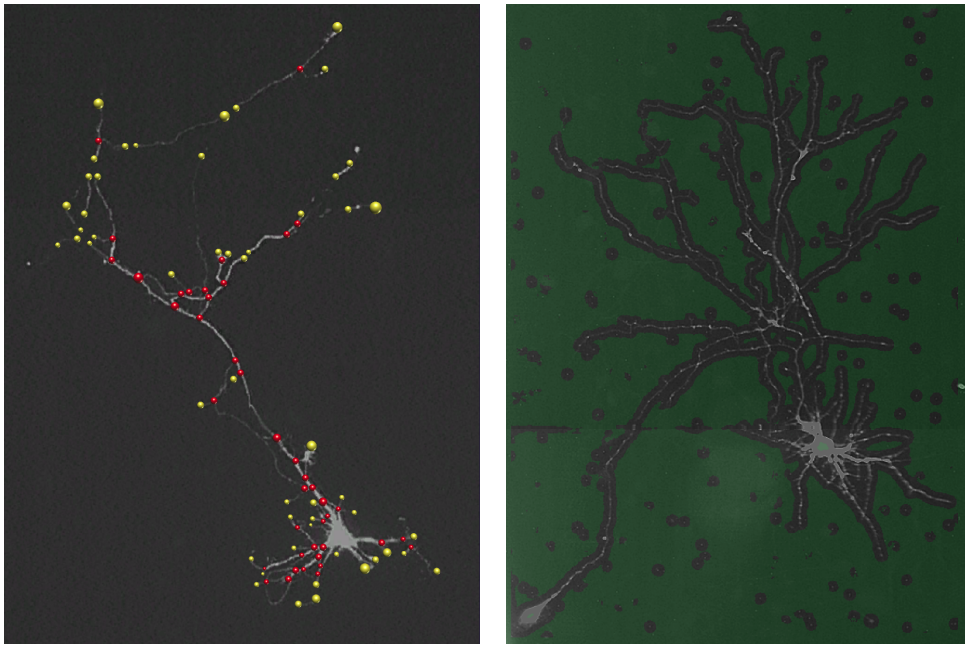
The complexity and functionality of the brain depend critically on the morphology and related interconnectivity of its neuronal cells [14, 75, 132]. To understand how a healthy brain processes information and how this capacity is negatively affected by psychiatric and neurodegenerative diseases [7, 151, 248] 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 [115, 193, 232, 260] 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 [76, 169] but the consensus of recent studies is that there is still much room for improvement in both accuracy and computational efficiency [154, 260].

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" throughout this chapter (Fig. 2.1), 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 [5,22,169]. 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 [21, 55, 56, 70, 74, 122, 182, 277, 291, 293]. 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 [71, 159, 193]. 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 [155, 157, 171, 182, 196, 228], or by fully automatic detection using sophisticated image filtering and pattern recognition methods (discussed in the chapter sequel). 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.

A novel method is presented - coined Neuron Pinpointer (NP) - for fully automatic detection and characterisation of critical points in fluorescence microscopy images of neurons. The method is described and evaluated 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 [311]. 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 chapter is a considerably extended version of the conference report [213]. The filtering algorithms and fuzzy-logic rules had been modified so as to be able to detect both junction and termination points. In addition, the full details of the method are presented here and an extensive evaluation based on both manually annotated real neuron images and computer generated neuron images. To obtain the latter a new computational approach is proposed based on publicly available expert manual tracings. Chapter starts with a brief overview of related work on critical-point detection (Section 2.2) and then the underlying concepts (Section 2.3), implementational details



a) b)

Figure 2.1: a) 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. b) 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^{th} variation percentile as threshold. In this example, 25% of the total number of pixels is selected for further processing.

(Section 2.4), and experimental evaluation (Section 2.5) are presented, followed by a summary of the conclusions that can be derived from the results (Section 2.6).

2.2 Related work

Detecting topologically critical points in images has been a long-standing problem in many areas of computer vision. This section provides with a brief review of the problem and proposed solutions in order to put presented 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") [116, 142, 246] or three or more lines join ("junctions of lines") [72, 299]. 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 [127, 136]. Hence, research in this area has focused on finding image locations where three (or more) elongated structures join [3, 4, 17, 26, 28, 42, 185, 259, 266, 308].

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 [4, 27, 74, 120, 124, 144, 200, 286], but these typically produce very fragmented results. Popular methods to enhance elongated image structures prior to segmentation include Hessian based analysis [5, 22, 92, 180, 222, 269, 293, 300, 305], Laplacian-of-Gaussian filters [56], Gabor filters [17, 28], phase congruency analysis [184], and curvelet based image filtering approaches [181]. 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 [185], steerable wavelet based local symmetry detection [203], or combining eigen analysis of the Hessian and correlation matrix [259]. 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 [174]. Image filtering methods for bifurcation detection have also been combined with supervised machine-learning based approaches such as support vector machines [269], artificial neural networks [27], or with multiple classifiers using AdaBoost [308], 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 author knowledge, no generic methods currently exist for critical-point detection in neuron images. The method proposed in this chapter aims to fill this gap and to complement exploratory neuron reconstruction algorithms that initialize on a set of seed points.

2.3 Proposed method

Proposed method for detection and characterization of critical points consists of three steps: feature extraction (Section 2.3.1), fuzzy-logic based mapping (Section 2.3.2), and, finally, critical-point determination (Section 2.3.3). Each step is described further in detail.

2.3.1 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, unnecessary computations are avoided by first performing a foreground selection step (Sec. 2.3.1.1), 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 profile (Section 2.3.1.2) of each foreground pixel is constructed, from which the quantitative features are computed.

2.3.1.1 Foreground selection

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

$$\rho(x, y) = \mathcal{P}_{95}(\mathcal{I}_{xy}) - \mathcal{P}_{5}(\mathcal{I}_{xy}) \tag{2.1}$$

$$\mathcal{I}_{xy} = \left\{ I(m, n) \mid (m - x)^2 + (n - y)^2 \leq r_d^2 \right\} \tag{2.2}$$

$$x, m \in [0, W - 1] \text{ and } y, n \in [0, H - 1] \tag{2.3}$$

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 [79]. 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). The practical experimentation proved that in this particular application a percentile of around 75 is a safe threshold (Fig. 2.1b). Small

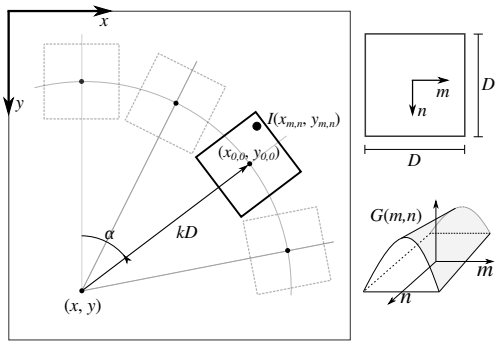


Figure 2.2: Geometry involved in the computation of the angular profile. In effect, the value of $p(x, y, \alpha, k, D)$ is the correlation of the image $I(x, y)$ with the kernel $G(m, n)$ of size $D \times D$ pixels, after rotating the kernel patch over angle α and shifting it over kD with respect to (x, y).

gaps in the foreground region are closed by morphological dilation. The resulting set of foreground pixel locations is denoted by F. Similarly, such application requires the parameter r_d to be typically set to the diameter of the axonal and dendritic structures observed in the image.

2.3.1.2 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 the image is correlated with a set of oriented kernels distributed evenly over a range of angles around that location [213]. 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. 2.2):

$$G(x, y) = \mathrm{e}^{-x^2/2\sigma_D^2}/S \tag{2.4}$$

where S is a normalization factor such that the sum of $G(x, y)$ over all kernel pixels is unity. The Gaussian is selected both because the cross-sectional profile of axons and dendrites in such applications is observed to be 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 kernel profile and should correspond to the expected branch diameter.

The local angular profile at any pixel location (x, y) in the given image I is computed using the kernel as:

$$p(x, y, \alpha, k, D) = \sum_m \sum_n I(x_{m,n}, y_{m,n})\, G(m, n) \tag{2.5}$$

where the transformed image coordinates are obtained as:

$$\begin{bmatrix} x_{m,n} \\ y_{m,n} \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + kD \begin{bmatrix} \sin\alpha \\ -\cos\alpha \end{bmatrix} + \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} m \\ n \end{bmatrix} \tag{2.6}$$

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. 2.2). 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 [43,299,305], this approach employs the matched kernel (Eq. 2.4), 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 presented framework, the computational burden is drastically reduced by the foreground selection step, and further reduction is possible since the filtering process is highly parallelizable.

The computed angular profile is further processed in order to extract several features (Fig. 2.3) relevant for critical-point detection and characterization:

Peaks. At each foreground pixel location, method determines the number and direction of the line-like image structures which 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 [90]. 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).

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

$$l_i = \frac{p(x, y, \hat{\alpha}_i, k, D) - p_{\min}}{p_{\max} - p_{\min}} \tag{2.7}$$

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

Energy. Next, the local grid $\pi_i(x, y, \hat{\alpha}_i, k, D)$ is considered for each $\hat{\alpha}_i$ (Fig. 2.3), consisting of the transformed coordinates $(x_{m,n}, y_{m,n})$ corresponding to $\alpha = \hat{\alpha}_i$ (Eq. 2.6), and a refined centerline point set λ_i (or "streamline") is extracted over this grid by finding for each n the local maximum over m:

$$\lambda_i = \left\{(x_{\hat{m}_n,n}, y_{\hat{m}_n,n})\right\}_{n \in [-D/2, D/2]} \tag{2.8}$$

$$\hat{m}_n = \underset{m \in [-D/2, D/2]}{\arg\max} \; I(x_{m,n}, y_{m,n}) \tag{2.9}$$

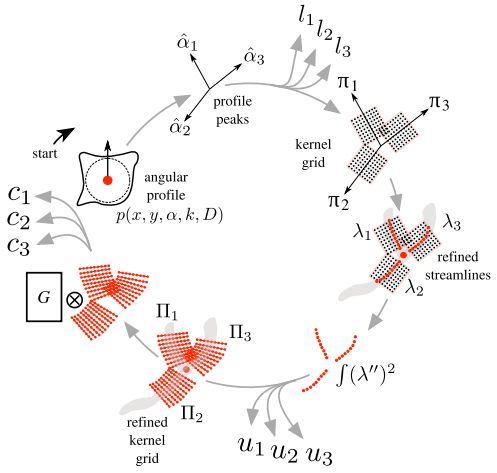


Figure 2.3: Flowchart of the feature extraction scheme. The example showcases a bifurcation but the same scheme is used also for terminations. The scheme, which starts with the angular profile $p(x, y, \alpha, k, D)$ and is executed clockwise, is applied to each pixel in the selected foreground regions and results in the set of features l_i, u_i, and c_i, where i indexes the streamlines. See main text for details.

The degree of the streamline deviation from a straight line is quantified by estimating its bending energy $u_i \geq 0$ as:

$$u_i = \frac{1}{\Delta m} \sum_n \left(\hat{m}_{n-1} - 2\hat{m}_n + \hat{m}_{n+1}\right)^2 \tag{2.10}$$

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, the orthogonal direction is estimated 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 a refined local grid $\Pi_i(x, y, \hat{\alpha}_i, k, D)$ is sampled, consisting of image coordinates $(\tilde{x}_{m,n}, \tilde{y}_{m,n})$ relative to the streamline (Fig. 2.3), and compute a normalized cross-correlation [147,297] score $c_i \in [-1, 1]$ as:

$$c_i = \frac{\sum_m \sum_n \left[I(\tilde{x}_{m,n}, \tilde{y}_{m,n}) - \bar{I}\right]\left[G(m,n) - \bar{G}\right]}{\sqrt{\sum_m \sum_n \left[I(\tilde{x}_{m,n}, \tilde{y}_{m,n}) - \bar{I}\right]^2 \sum_m \sum_n \left[G(m,n) - \bar{G}\right]^2}} \tag{2.11}$$

where, similar to the angular profile calculation (Eq. 2.5), 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

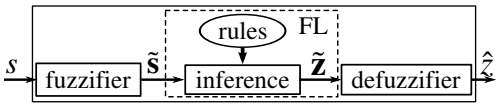


Figure 2.4: Scheme of a single input/output fuzzy-logic (FL) system. A scalar input value s is converted to a vector $\tilde{\mathbf{s}}$ containing the memberships of s for each of the input fuzzy sets, resulting in a vector $\tilde{\mathbf{z}}$ containing the memberships of z for each of the output fuzzy sets.

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), the largest score of a set of templates with standard deviations of the Gaussian profile model [259] is taken, covering $\left\{1, \ldots, \lfloor \frac{D}{2} \rfloor \right\}$ set of values measured in pixels.

2.3.2 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" [301], and allowing for nonlinear input-output mappings, the method is tailored to use fuzzy logic for this purpose. Fuzzy logic has been successfully used in many areas of engineering [172] but, based on a thorough search of the relevant literature - has not been explored for neuron critical-point analysis. In this chapter, the basics of fuzzy logic (Section 2.3.2.1) are briefly described followed by the description of the fuzzy-logic system tailored for calculating critical-point maps of neuron images (Section 2.3.2.2).

2.3.2.1 Basics of fuzzy logic

In a fuzzy-logic system (Fig. 2.4), 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 [172]. As an example, the linguistic terms LOW and HIGH may be used to express the used quantities, representing the subjective notions "low" and "high", respectively. The degrees to which "s is low" (which in this chapter will be denoted as $s = \text{LOW}$) and "s is high" ($s = \text{HIGH}$) are given by membership values $\mu_{\text{LOW}}(s)$ and $\mu_{\text{HIGH}}(s)$, respectively. The output of the fuzzification step thus becomes $\tilde{\mathbf{s}} = [\mu_{\text{LOW}}(s), \mu_{\text{HIGH}}(s)]^T$.

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:

$$R_i\colon \quad \text{IF } (s_1 = \text{HIGH}) \ \wedge \ (s_2 = \text{LOW}) \ \text{ THEN } \ (z = \text{OFF}) \tag{2.12}$$

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 [172]. In the example, the IF-part of R_i (Eq. 2.12) would result in the following intermediate value (degree of verity):

$$\upsilon_i = \min\left\{\mu_{\text{HIGH}}(s_1), \mu_{\text{LOW}}(s_2)\right\} \tag{2.13}$$

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:

$$\Upsilon_i(z) = \min\left\{\mu_{\text{OFF}}(z), \upsilon_i\right\} \tag{2.14}$$

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 [172] is by means of a weighted fuzzy union:

$$\Upsilon(z) = \max\left\{w_1\Upsilon_1(z), \ldots, w_{N_R}\Upsilon_{N_R}(z)\right\} \tag{2.15}$$

Although it is possible to assign a different weight to each rule by setting $w_i \in [0, 1]$, in the introduced applications this is not critical, and therefore $w_i = 1$ is used 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 [172]:

$$\hat{z} = \frac{\int z\Upsilon(z)dz}{\int \Upsilon(z)dz} \tag{2.16}$$

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

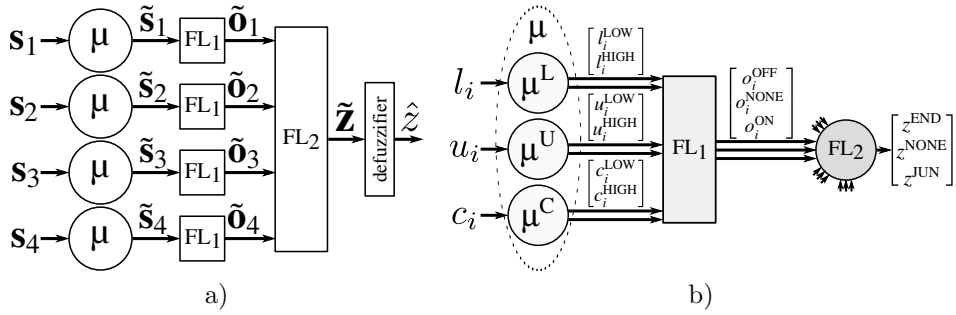


Figure 2.5: Architecture of the proposed fuzzy-logic system: a) critical-point detection: a cascade of two fuzzy-logic modules (FL_1 and FL_2), where the first determines the degree to which streamlines (up to four) are present at the image location under consideration, and based on this information the second determines the degree to which that location corresponds to the possible types of critical points, b) 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.

2.3.2.2 Termination and junction mapping

To determine the presence and type of critical point at any foreground image location, a cascade of two fuzzy-logic systems is used, representing two decision levels (Fig. 2.5a). 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 (Section 2.3.1.2). For each streamline (Fig. 2.5b), 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, 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 the application introduced in this chapter trapezoidal membership functions are used, each having two inflection points, such that μ_{LOW} and μ_{HIGH} are each other's complement (Fig. 2.6). For example, the degrees to which $l_i = \mathrm{LOW}$ and $l_i = \mathrm{HIGH}$, are given by $l_i^{\mathrm{LOW}} = \mu^L_{\mathrm{LOW}}(l_i)$ and $l_i^{\mathrm{HIGH}} = \mu^L_{\mathrm{HIGH}}(l_i) = 1 - l_i^{\mathrm{LOW}}$, respectively, and because of this complementarity it is sufficient to use μ^L to refer to both membership functions (Fig. 2.5) throughout the manuscript. Similarly, the membership degrees of u_i and c_i are given by μ^U and μ^C, respectively. In summary, the following notations

and definitions for the fuzzification step are used:

$$\begin{aligned} \mu^L\colon \quad l_i \quad &\rightarrow \quad \tilde{\mathbf{l}}_i \quad = \left[l_i^{\text{LOW}},\, l_i^{\text{HIGH}}\right]^T \\ \mu^U\colon \quad u_i \quad &\rightarrow \quad \tilde{\mathbf{u}}_i \quad = \left[u_i^{\text{LOW}},\, u_i^{\text{HIGH}}\right]^T \\ \mu^C\colon \quad c_i \quad &\rightarrow \quad \tilde{\mathbf{c}}_i \quad = \left[c_i^{\text{LOW}},\, c_i^{\text{HIGH}}\right]^T \end{aligned} \tag{2.17}$$

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. 2.6).

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:

$$\text{FL}_1\colon\ \tilde{\mathbf{s}}_i \rightarrow \tilde{\mathbf{o}}_i = \left[o_i^{\text{OFF}},\, o_i^{\text{NONE}},\, o_i^{\text{ON}}\right]^T \tag{2.18}$$

To calculate these memberships, scalar variable o is introduced, where $o = 0$ corresponds to OFF, $o = 1$ to NONE, and $o = 2$ to ON. Also, Gaussian membership functions μ^O_{OFF}, μ^O_{NONE} and μ^O_{ON} are defined, centered around 0, 1, and 2, respectively (Fig. 2.7), 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 2.1. 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 2.1) is given by:

$$R_8\colon\ \text{IF } (l = \text{HIGH}) \ \wedge\ (u = \text{HIGH}) \ \wedge\ (c = \text{HIGH}) \text{THEN } (o = \text{ON}) \tag{2.19}$$

which results in the verity value:

$$\upsilon_8 = \min\left\{\mu^L_{\text{HIGH}}(l), \mu^U_{\text{HIGH}}(u), \mu^C_{\text{HIGH}}(c)\right\} \tag{2.20}$$

and the output fuzzy set:

$$\Upsilon_8(o) = \min\left\{\mu^O_{\text{ON}}(o), \upsilon_8\right\} \tag{2.21}$$

All the rules are resolved and combined as:

$$\Upsilon(o) = \max\left\{\Upsilon_1(o), \ldots, \Upsilon_8(o)\right\} \tag{2.22}$$

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 (Eq. 2.18) is calculated using the membership functions:

$$\tilde{\mathbf{o}}_i = \left[\mu^O_{\text{OFF}}(\hat{o}_i),\, \mu^O_{\text{NONE}}(\hat{o}_i),\, \mu^O_{\text{ON}}(\hat{o}_i)\right]^T \tag{2.23}$$

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

$$\text{FL}_2\colon\ \tilde{\mathbf{o}} \rightarrow \tilde{\mathbf{z}} = \left[z^{\text{END}},\, z^{\text{NONE}},\, z^{\text{JUN}}\right]^T \tag{2.24}$$

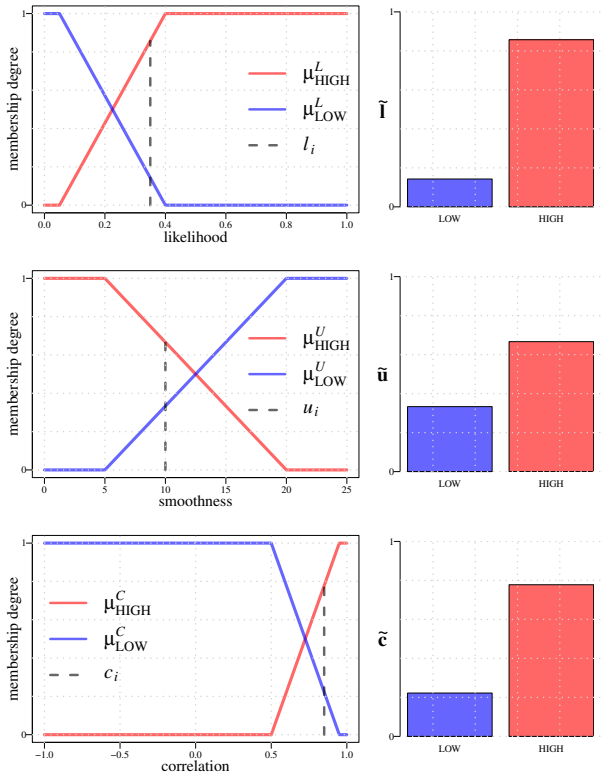


Figure 2.6: Input membership functions used in the fuzzification step for FL_1. Example LOW and HIGH membership values are shown (right column) for input values (dashed vertical lines in the plots on the left) $l_i = 0.35$ (top row), $u_i = 10$ (middle row), and $c_i = 0.85$ (bottom row). The inflection 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).

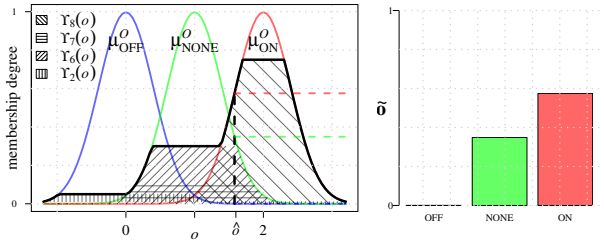


Figure 2.7: Output membership functions used in module FL_1. Example output fuzzy sets Υ_i corresponding to rules R_i from Table 2.1 are shown as the textured areas. Value $\hat{o}$ (left panel) represents the centroid of the aggregated output fuzzy sets. The resulting output membership values (right panel) serve as input for module FL_2.

Table 2.1: 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

To calculate these memberships, scalar variable z is introduced, where $z = 1$ corresponds to END, $z = 2$ to NONE, and $z = 3$ to JUN. Corresponding Gaussian membership functions μ^Z_{END}, μ^Z_{NONE}, and μ^Z_{JUN} are defined. They are centered around 1, 2, and 3, respectively, and with fixed standard deviation 0.4 as before (Fig. 2.8). 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 2.2. 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), it is assumed that there is no critical point. Similar to FL_1, all the rules of FL_2 are evaluated and their results combined as:

$$\Upsilon(z) = \max\{\Upsilon_1(z), \ldots, \Upsilon_{22}(z)\} \tag{2.25}$$

which, after centroid defuzzification, results in a scalar output value $\hat{z}$, from which the output of FL_2 (Eq. 2.24) is calculated using the membership functions:

$$\tilde{\mathbf{z}} = \left[\mu^Z_{\text{END}}(\hat{z}),\, \mu^Z_{\text{NONE}}(\hat{z}),\, \mu^Z_{\text{JUN}}(\hat{z})\right]^T \tag{2.26}$$

Table 2.2: 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

The proposed fuzzy-logic system is applied to each foreground pixel location $(x, y) \in F$ (Section 2.3.1.1) so that all memberships introduced above may be indexed by (x, y). Based on this, the following two maps are calculated:

$$I_{\text{END}}(x,y) = \begin{cases} z^{\text{END}}(x,y) & \text{if } (x,y) \in F \\ 0 & \text{otherwise} \end{cases} \tag{2.27}$$

$$I_{\text{JUN}}(x,y) = \begin{cases} z^{\text{JUN}}(x,y) & \text{if } (x,y) \in F \\ 0 & \text{otherwise} \end{cases} \tag{2.28}$$

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

2.3.3 Critical-point determination

The ultimate aim of the introduced 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

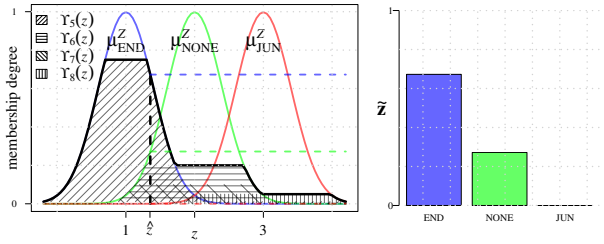


Figure 2.8: Output membership functions used in module FL_2. Example output fuzzy sets Υ_i corresponding to rules R_i from Table 2.2 are shown as the textured areas. Value $\hat{z}$ (left panel) represents the centroid of the aggregated output fuzzy sets. The resulting output membership values (right 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.

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 the real-valued scores in the maps are first regularized by means of local-average filtering with a radius of 3-5 pixels. Next, max-entropy based automatic thresholding [133] is applied to segment the maps, as in contrast with many other thresholding methods it was found 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 [252] 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

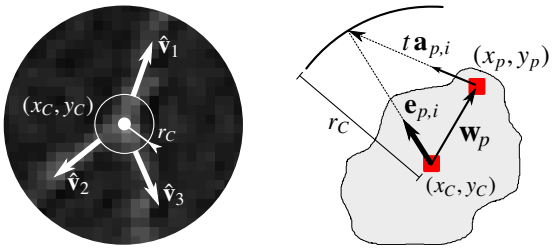


Figure 2.9: Critical-point determination. A critical point is characterized by its type, centroid location (x_C, y_C), radius r_C, and its main branch directions $\hat{\mathbf{v}}_i$ (left panel, in this case a bifurcation), aggregated from the pixels (x_p, y_p) in the corresponding critical region (right panel).

representative critical-point location $\mathbf{x}_C = (x_C, y_C)$ is calculated as:

$$\mathbf{x}_C = \frac{1}{N_p} \sum_{p=1}^{N_p} \mathbf{x}_p \tag{2.29}$$

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

$$r_C = \max_p \{||\mathbf{w}_p||\} \tag{2.30}$$

where $\mathbf{w}_p = \mathbf{x}_p - \mathbf{x}_C$ (Fig. 2.9). To obtain regularized estimates of the main branch directions $\hat{\mathbf{v}}_i$ for the critical point, the directions are aggregated corresponding to the angular profile peaks $\hat{\alpha}_i$ (Section 2.3.1.2) of all the $\mathbf{x}_p$ in the region as follows. For each $\mathbf{x}_p$, $N_{\hat{\alpha}} \leq 4$ angular profile peak direction vectors $\mathbf{a}_{p,i} = [\cos \hat{\alpha}_i(\mathbf{x}_p), \sin \hat{\alpha}_i(\mathbf{x}_p)]^T$ are found. Each of these vectors defines a line $\mathbf{a}(t) = \mathbf{x}_p + t\,\mathbf{a}_{p,i}$ parameterized by $t \in \mathbb{R}$. The projection of this line onto the circle $||\mathbf{x} - \mathbf{x}_C||^2 = r_C^2$ is established by substituting $\mathbf{x} = \mathbf{a}(t)$ and solving for t. From this, the contributing unit vector is calculated (Fig. 2.9):

$$\mathbf{e}_{p,i} = \frac{1}{r_C}(\mathbf{w}_p + t\,\mathbf{a}_{p,i}) \tag{2.31}$$

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 [52] 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}, only one main branch direction is needed, simply taken 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}, the $\hat{\mathbf{v}}_i$ (at least three) are taken as the main branch directions to which the largest number of $\mathbf{e}_{p,i}$ were shifted. These calculations are performed for all critical regions.

2.4 Implementational details

The method was implemented in the Java programming language as a plugin for the image processing and analysis tool ImageJ [1, 229]. Since the feature extraction step (Section 2.3.1), in particular the matched filtering for angular profile analysis, is quite computationally demanding, parallelization is applied 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 to run the 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 (Section 2.3.1.2) and the inflection points L_{LOW}, L_{HIGH}, U_{LOW}, U_{HIGH}, C_{LOW},

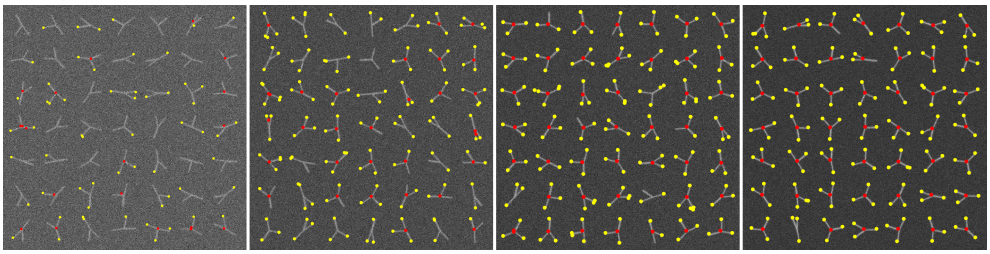


Figure 2.10: Examples of simulated triplet images and detection results. Each triplet consists of three branches with different diameters which join at one end to form a bifurcation point and with the other ends being termination points. Images were generated at SNR $= 2, 3, 4, 5$ (left to right panel). The detection results with presented 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 the method.

and C_{HIGH} of the input membership functions used by fuzzy-logic module FL_1 (Section 2.3.2.2). In the showcased application, parameter D is set 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, L_{LOW} is typically set close to 0 and L_{HIGH} around 0.5 (Fig. 2.6). 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, the histogram of all calculated energy values in the image is used. Parameter U_{LOW} is set to the threshold computed by the well-known triangle algorithm, while typically $U_{\mathrm{HIGH}} \gg U_{\mathrm{LOW}}$. It is useful to note that the membership functions defined by these parameters are inverted (Fig. 2.6) 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 practical applications they are usually set to $C_{\mathrm{LOW}} \in [0.1, 0.5]$ and $C_{\mathrm{HIGH}} = 0.95$ (Fig. 2.6).

All other aspects of the 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 (Section 2.3.1.1) can be set equal to D, and the standard deviation σ_D of the Gaussian profile (Section 2.3.1.2) 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 (Section 2.3.2.2), as they are not essentially influencing the performance of the algorithm.

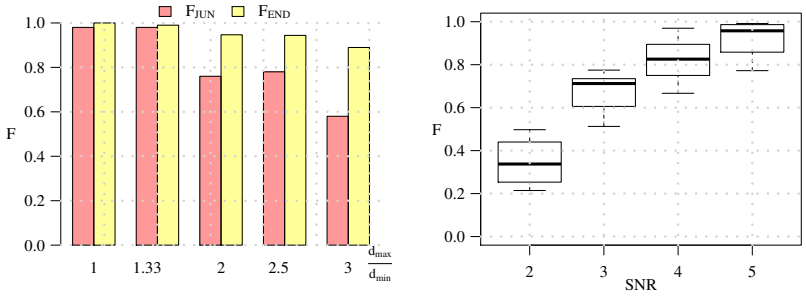


Figure 2.11: Performance of the method in detecting terminations and junctions in simulated images of triplets. The values of F_{END} and F_{JUN} are shown (left panel) for the various branch diameter ratios d_{max}/d_{min} at SNR = 4. The distribution of F_{BOTH} values is shown as a box plot (right panel) for the various SNR levels.

2.5 Experimental results

To evaluate the performance of Neuron Pinpointer method in correctly detecting and classifying neuronal critical points, 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) have been performed. After outlining the performance measures (Section 2.5.1), the results of the evaluation on simulated images are presented, including the synthetic triplets (Section 2.5.2) and synthetic neurons (Section 2.5.3), and on real neuron images (Section 2.5.4). Finally, the results of a comparison of the method with two other methods (Section 2.5.5) is showcased.

2.5.1 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, the true-positive (TP), false-positive (FP), and the false-negative (FN) critical-point detections had been counted, and used 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 [201], 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, the harmonic mean of the two F-measures: $F_{BOTH} = 2F_{END}F_{JUN}/(F_{END} + F_{JUN})$ is also computed.

2.5.2 Evaluation on simulated triplet images

Before considering full neuron imagery the performance of the method was first evaluated at detecting terminations and junctions in a very basic configuration as a function

of image quality. To this end, a triplet model was used, consisting of a single junction modeling a bifurcation, having three branches with arbitrary orientations (angular intervals) and diameters (Fig. 2.10). 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 (Section 2.3.1.2) uses a fixed kernel size D, the intent was to investigate the detection robustness for varying ratios $d_{\max}/d_{\min}$ between the maximum and the minimum branch diameter in a triplet. For such experiment, $1, 0.33, 2, 2.5, 3$ ratios were considered 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. A set of images was simulated for each configuration containing $1,000$ well-separated triplets for signal-to-noise ratio levels SNR = 2, 3, 4, 5 (see cropped examples in Fig. 2.10). These levels are chosen knowing that SNR = 4 is a critical level in other detection problems [53, 250]. Poisson noise was used in simulating fluorescence microscopy imaging of the triplets. The obtained results of this experiment (Fig. 2.11) lead to the conclusion that the method is fairly robust for diameter ratios $d_{\max}/d_{\min} \leq 2\frac{1}{2}$ and an image quality of SNR ≥ 4. Based on the detection rates, it is also possible to draw a conclusuion that the presented method is somewhat better in detecting terminations than detecting junctions in synthetic images. Example detection results for $d_{\max}/d_{\min} \leq 2$ for the considered SNR levels are shown in Fig. 2.10.

2.5.3 Evaluation on simulated neuron images

To evaluate the method on more complex images, with the known ground truth, the imaging of complete neurons was simulated. Although there are various ways this could be done [136, 277], the existing expert reconstructions from the NeuroMorpho.Org database [15] were used. A total of 30 reconstructions from different neuron types were downloaded as SWC files [44], 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, the images of SNR = 2, 3, 4, 5 (Fig. 2.12) were generated. The simulated images of neurons are obtained this way, each with the exactly known location of the termination and junction point, extrapolated from the source SWC file. The evaluation results (Fig. 2.13) confirm the conclusion from the experiments on triplets that the method performs well for SNR ≥ 4 and is somewhat better in detecting terminations than detecting junctions. For SNR = 4 the performance for junction detection is $\mathrm{F}_{\mathrm{JUN}} \approx 0.85$ while for termination detection $\mathrm{F}_{\mathrm{END}} \approx 0.95$. 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. 2.14.

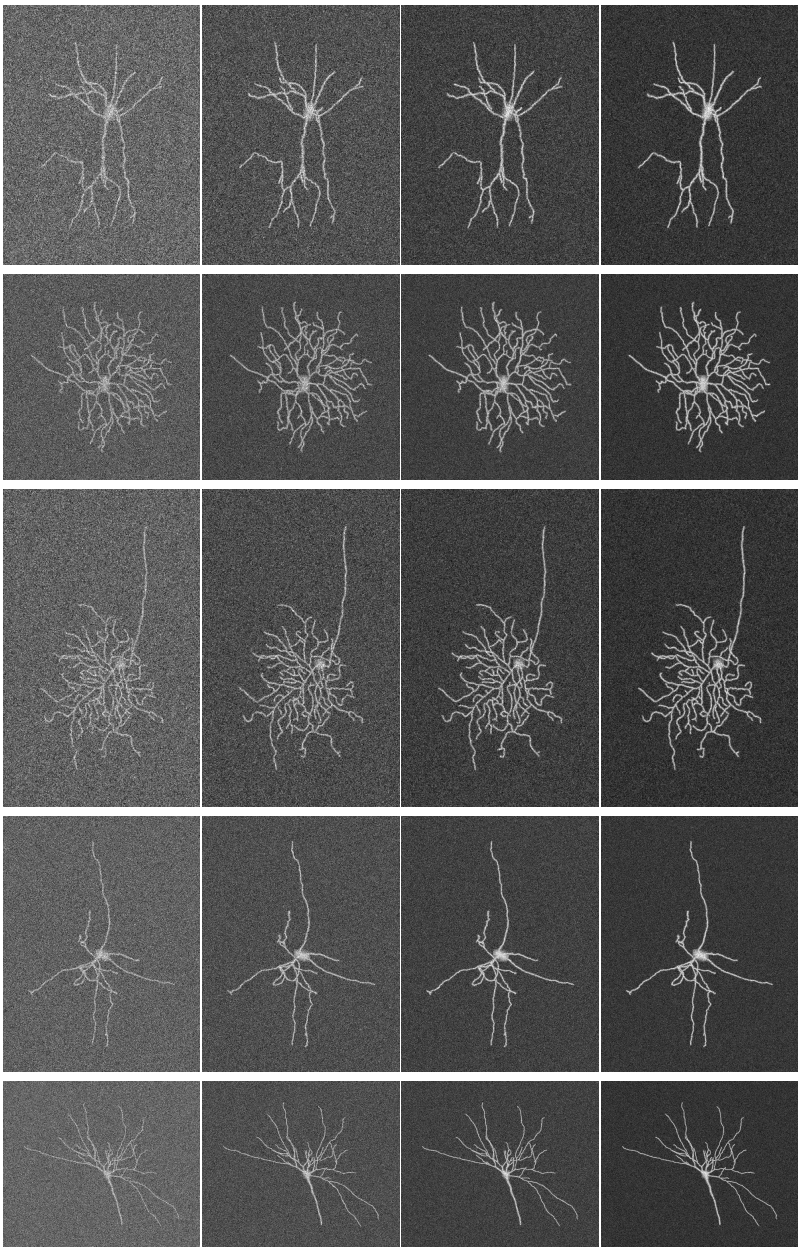


Figure 2.12: Examples of simulated neuron images based on expert reconstructions from the NeuroMorpho.Org database. The images show a wide range of morphologies (one type per row) and image qualities of SNR = 2, 3, 4, 5 (from left to right per row).

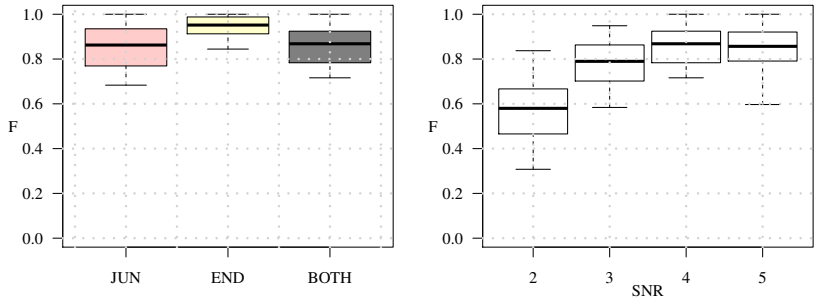


Figure 2.13: Performance of the method in detecting terminations and junctions in 30 simulated images of neurons. The distributions of the F_{END}, F_{JUN}, and F_{BOTH} values are shown as box plots for SNR = 4 (left panel) and in addition the distribution of F_{BOTH} is shown for SNR = 2, 3, 4, 5 (right panel).

2.5.4 Evaluation on real neuron images

As the ultimate test case, the method was also evaluated on real fluorescence microscopy images of neurons from a published study [255]. 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. Naturally, real images are generally more challenging than simulated images, as they contain more ambiguities due to labeling and imaging imperfections, and thus the lower performance was expected. Since in this case there is no control over the SNR in the images, the detection results are reported for all images together. The evaluation results (Fig. 2.15) indicate 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, the junction detection performance with the real images is better than the termination detection which had not been the case with the synthetic neurons. A possible explanation for this could be that in the simulated images a constant intensity was used for the neuron branches, as a result of which terminations are as bright as junctions but 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 the Fig. 2.16.

2.5.5 Comparison with other methods

Finally the performance comparison of the Neuron Pinpointer method against other methods is conduct. In lack of availability of other methods explicitly designed to detect and classify critical points in neuron images before reconstruction, two existing software tools relevant in this context were considered and their implicit detection capabilities compared with the presented explicit method. If NP indicates better performance, this would indicate that the existing methods may be improved by ex-

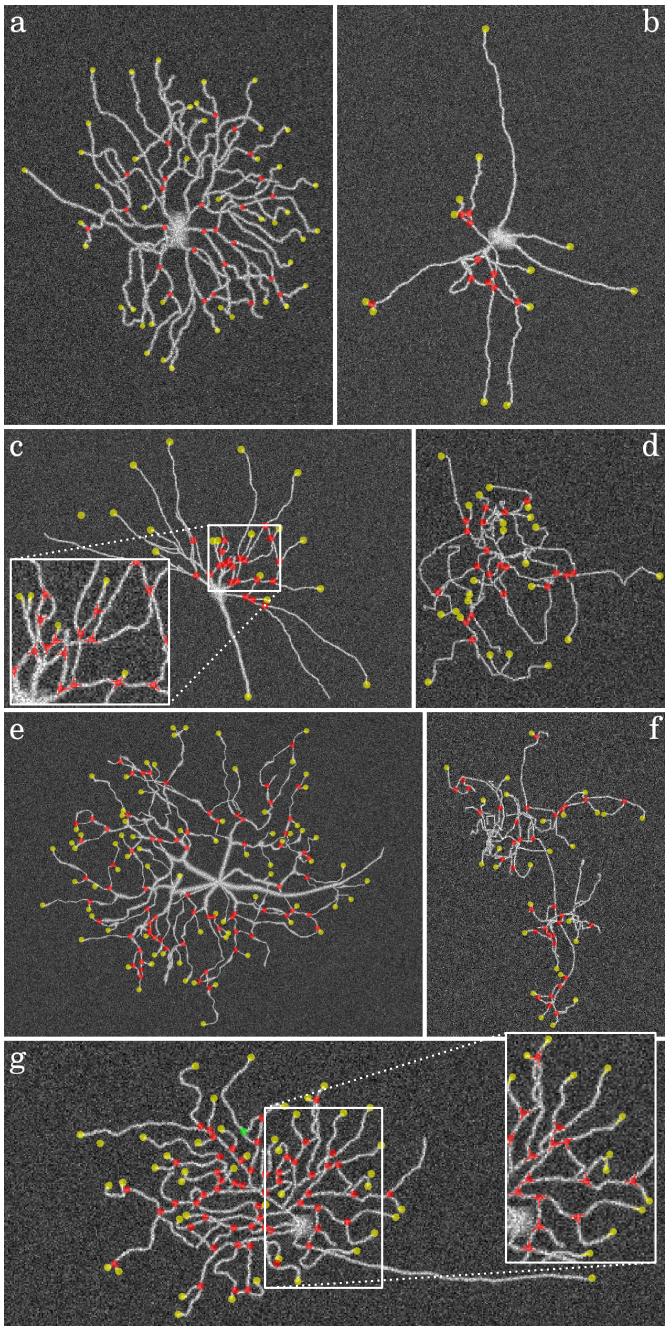


Figure 2.14: Example detections for simulated neuron images at SNR = 4. The showcased images are contrast enhanced and show the detected terminations (yellow circles) and junctions (red circles) as overlays with fixed radius for better visibility. The value of F_{BOTH} in these examples is (a) 0.69, (b) 0.85, (c) 0.85, (d) 0.77, (e) 0.75, (f) 0.68, (g) 0.86.

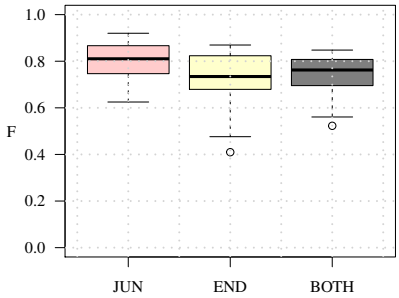


Figure 2.15: Performance of the method in detecting terminations and junctions in 30 real fluorescence microscopy images of neurons. The distributions of the F_{END}, F_{JUN}, and F_{BOTH} values for all images together are shown as box plots.

ploiting the output of the method. The first tool, AnalyzeSkeleton (AS)[1] [9], is an ImageJ plugin for finding and counting all end-points and junctions in a skeleton image. To obtain skeleton images of the neuron images used for this study, the related skeletonization method[2] inspired by an advanced thinning algorithm [146] was used. The input for the latter is a binary image obtained by segmentation based on smoothing (to reduce noise) and thresholding. A range of smoothing scales is considered in the experiments 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.

The second tool, All-Path-Prunning (APP2) [291], is a plugin for Vaa3D[3] [189, 196]. 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 aforementioned experiments it is straightforward to retrieve them from the SWC output files. In principle, any neuron reconstruction method is also implicitly a critical-point detection method, and its performance could be quantified by comparing the output tree nodes with the reference data. The interesting question is whether an explicit detector such as NP outperforms the implicit detection carried out in a tool such as APP2. The user parameters of the tool were manually adjusted to get optimal performance in presented experiments.

A comparison of the F-scores of NP, AS, and APP2 for the 30 real neuron images used throughout the experiments is presented in Fig. 2.17. The plots indicate that the detection rates of the 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 presented explicit and integrated approach for detecting critical points, as accurate neuron reconstruction

[1] available from http://fiji.sc/AnalyzeSkeleton
[2] http://fiji.sc/Skeletonize3D
[3] available from http://www.vaa3d.org/

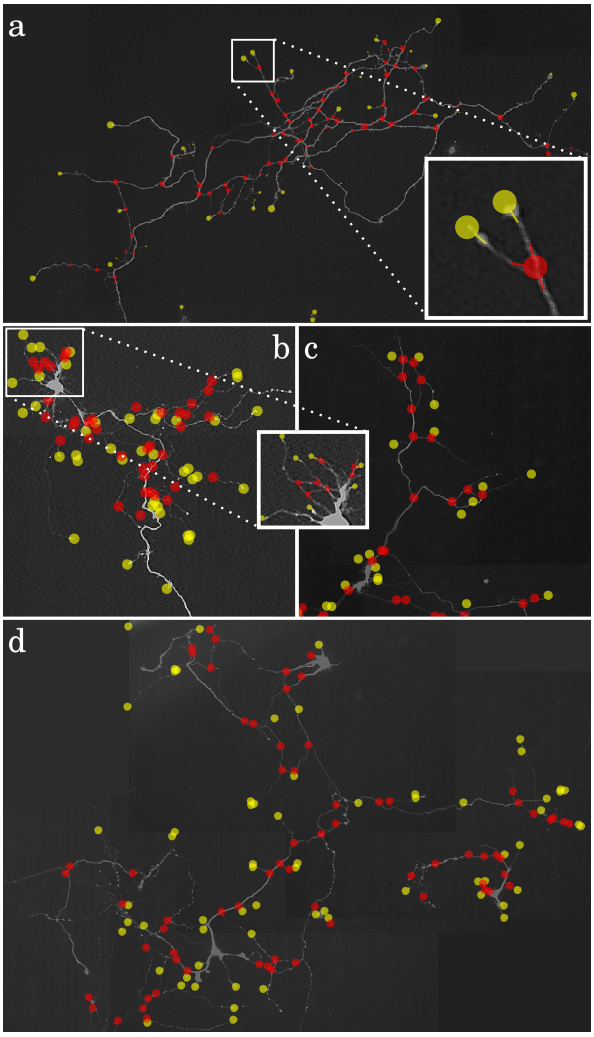


Figure 2.16: Example detections for four real neuron images. The detected terminations (yellow circles) and junctions (red circles) are shown as overlays with fixed radius for better visibility. The value of F_{BOTH} in these examples is (a) 0.82, (b) 0.78, (c) 0.68, (d) 0.65.

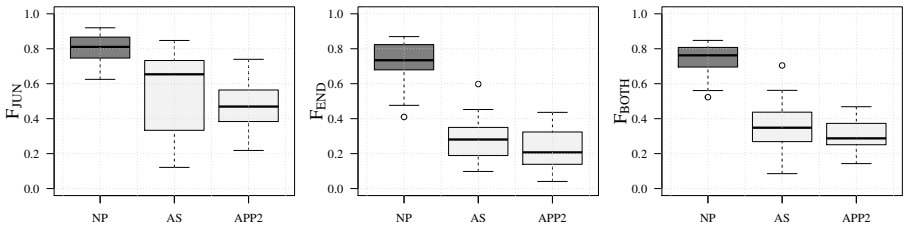


Figure 2.17: Critical-point detection performance of the introduced method (NP) compared to two other methods (AS and APP2). The median values of F_{JUN} (left plot) are 0.81 (NP), 0.65 (AS), and 0.47 (APP2). The median values of F_{END} (middle plot) are 0.73 (NP), 0.28 (AS), and 0.21 (APP2). Finally, the median values of F_{BOTH} (right plot) are 0.76 (NP), 0.35 (AS), and 0.29 (APP2).

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 the used 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 the presented 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).

2.6 Conclusions

A novel method for solving the important problem of detecting and characterizing critical points in the tree-like structures in neuron microscopy images is presented. Based on directional filtering and feature extraction 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 experimentation on simulated as well as real fluorescence microscopy images, it is possible to conclude that the method achieves substantially higher detection rates than the ones 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 obtained results suggest that NP may provide important clues to improve the performance of reconstruction methods. Actual integration of the detection method with existing tracing methods is a potential future direction, as the ultimate aim is further utilization, especially from the context of the neuron tracing, as well as adjusting the processing to the 3D imagery. Although the main focus in this work has been the neuron analysis, introduced method may be potentially useful for other applications involving tree-like image structures, such as blood vessel or bronchial tree analysis. Such applications, however, would require further research. For this purpose it may

be helpful to increase the robustness of the detection method to larger branch diameter ratios than the ones tested in this paper. This could be done, for example, by using multiscale filtering approaches, or by selective morphological thinning (or thickening). The software implementation of the presented method is available as an ImageJ plugin [4].

[4] available from https://bitbucket.org/miroslavradojevic/npinpoint

Chapter Three

Automated neuron tracing using probability hypothesis density filtering

The functionality of neurons and their role in neuronal networks is tightly connected to the cell morphology. A fundamental problem in many neurobiological studies aiming to unravel this connection is the digital reconstruction of neuronal cell morphology from microscopic image data. Many methods have been developed for this, but they are far from perfect, and better methods are needed. In this chapter, a new method for tracing neuron centerlines needed for full reconstruction is presented. The method uses a fundamentally different approach than previous methods by considering neuron tracing as a Bayesian multi-object tracking problem. The problem is solved using probability hypothesis density filtering. Results of experiments on 2D and 3D fluorescence microscopy image datasets of real neurons indicate the proposed method performs comparably or even better than the state of the art.

Based upon: M. Radojević, E. Meijering, "Automated neuron tracing using probability hypothesis density filtering", *Bioinformatics*, vol. 33, no. 7, pp.1073-1080, 2017.

3.1 Introduction

Accurate reconstruction of the tree-like structure of neuronal cells from optical microscopy images is a crucial step in automating the analysis of single neuron morphology or the connectivity of neuronal networks [76, 169, 190]. Microscopic images provide detailed information about the geometrical and topological properties of the neuronal arbors. Extracting and representing this information in a faithful and convenient digital format is key to many studies [14, 15, 115, 158, 232, 260], as digital reconstructions enable neurobiologists to use computational approaches in addressing open issues in brain research, such as the relation between neuron structure and function, and the effects of neurodegenerative disease processes and drug compounds on neuron development and connectivity.

Existing approaches to tracing neurons in images can be broadly divided into global and local approaches. Global approaches consider the problem from the whole-image perspective and typically involve global image segmentation [22, 66, 285] or global optimization strategies [269, 291]. Local approaches, on the other hand, use local image exploration strategies starting from seed points [55, 192, 295] to find segments of the neuronal tree, which are then merged into a full tree representation. Both approaches have advantages and disadvantages and they are often combined to profit from their complementarity [131, 307].

A wide variety of computational concepts have been proposed in developing automated neuron tracing methods, whether global or local [2]. These include active contours [41, 160, 283], tubular models [222], principal curves [21, 205], perceptual grouping [181], path pruning [192, 291], critical point detection [5, 212], voxel scooping [217], dynamic and integer programming [268, 305], active learning [96], graph optimization [56, 269], tubularity flow field segmentation [179], marked point processes [23], iterative back-tracking [153], and more. A key characteristic relevant to the methodology presented in this chapter is that the vast majority of them are deterministic by nature. That is, they utilize models and algorithms that always assume or pass through the exact same sequence of states. While this behavior may seem virtuous and practically convenient, it is nonetheless not very realistic and not necessarily advantageous, for several reasons. For starters, expert human annotators, which are still considered to be the gold standard in evaluating methods, do not operate deterministically: their output will be (slightly) different every time they repeat a task. Also, any deterministic model is typically a (gross) simplification of reality, and consequently lacks flexibility in dealing with data variability. Finally, since every run of a deterministic algorithm will yield exactly the same output, it is not possible to accumulate evidence from multiple iterations.

In this chapter, a new method for neuron tracing in optical microscopy images is proposed that operates probabilistically rather than deterministically. Focusing on delineating the branch centerlines, it utilizes a Bayesian approach to blend two sources of information: the model (based on prior knowledge) and the measurements (from the image data). The main novelty is that it combines the problems of neuron segment detection and linking into one framework by performing simultaneous multi-object tracking. Traditional multi-object (also referred to as multi-target) tracking techniques [164, 257] typically assume the number of objects to be known and/or they

explicitly associate measurements with objects which are then Bayesian filtered individually [19]. In arbor tracing, the number of objects (neuron segments) is unknown a priori, therefore a different approach is used, based on filtering the so-called probability hypothesis density (PHD) function [163]. PHD filtering has gained popularity in recent years as a robust approach to tracking, since it is able to compensate for missing detections and to remove noise and clutter, while reducing the computational complexity from exponential to linear as the number of objects grows. Applications include radar and sonar tracking [57, 265], video surveillance [162, 284], and even motion tracking in microscopy [226, 288], but had not been explored for neuron tracing yet. Moreover, presented application differs fundamentally from other works in the sense that the filtering is applied in space rather than in time. The proposed method is evaluated on a variety of real image data (both 2D and 3D) taking expert manual annotations as the gold standard. Its performance is also compared with several state-of-the-art tools for neuron tracing [56, 205, 291].

3.2 Methods

3.2.1 Multi-object Bayesian filtering

Single-object tracking is considered as a Bayesian inference problem [20, 223]. The key idea is to estimate the posterior probability density function (pdf) $f_{k|k}(\mathrm{x}_k|\mathrm{z}_{1:k})$, where x_k denotes the object state at iteration k, and $\mathrm{z}_{1:k}$ the sequence of observations from iterations 1 to k inclusive. Estimation is accomplished by sequentially applying prior knowledge to predict the state in the next iteration and updating this estimate with available observations. Similarly, multi-object tracking can be formulated as the problem of updating predictions of the multi-object state $\mathrm{X}_k = \{\mathrm{x}_{k,1}, \ldots, \mathrm{x}_{k,N_k}\}$ with multi-object observations $\mathrm{Z}_k = \{z_{k,1}, \ldots, z_{k,M_k}\}$, where N_k and M_k denote the number of objects and observations at iteration k, respectively. The prediction is thus formulated as:

$$f_{k|k-1}(\mathrm{X}_k|\mathrm{Z}_{1:k-1}) = \int \Pi_{k|k-1}(\mathrm{X}_k|\mathrm{X}_{k-1}) f_{k-1|k-1}(\mathrm{X}_{k-1}|\mathrm{Z}_{1:k-1}) \delta\mathrm{X}_{k-1} \tag{3.1}$$

along with the update:

$$f_{k|k}(\mathrm{X}_k|\mathrm{Z}_{1:k}) = \frac{\vartheta_k(\mathrm{Z}_k|\mathrm{X}_k) f_{k|k-1}(\mathrm{X}_k|\mathrm{Z}_{1:k-1})}{\int \vartheta_k(\mathrm{Z}_k|\mathrm{X}) f_{k|k-1}(\mathrm{X}|\mathrm{Z}_{1:k-1}) \delta\mathrm{X}} \tag{3.2}$$

where $\Pi_{k|k-1}(\mathrm{X}_k|\mathrm{X}_{k-1})$ denotes the multi-object state transition probability and $\vartheta_k(\mathrm{Z}_k|\mathrm{X}_k)$ the multi-object likelihood. Filtering the multi-object posterior $f_{k|k}(\mathrm{X}_k|\mathrm{Z}_{1:k})$ suffers from serious practical obstacles, as the multi-object state can be very high-dimensional and hard to sample and integrate efficiently. Moreover, it is necessary to take into account changes in object numbers, which adds an often intractable combinatorial burden. Thus more feasible solutions are needed.

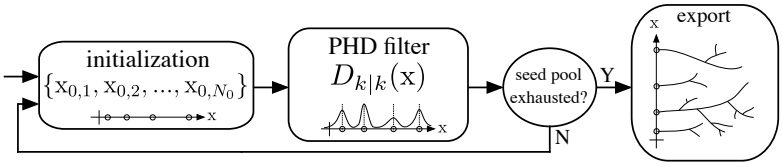


Figure 3.1: Method overview. Each multi-object filtering round is initialized with N_0 seeds. If the seed pool is not exhausted by the end of the current round, a new round is started, and this is repeated until all seeds have been processed.

3.2.2 Probability hypothesis density filtering

To overcome the difficulties of direct multi-object Bayesian filtering, presented solution proposes instead to filter the first-order statistical moment of the multi-object posterior $f_{k|k}(\mathrm{X}_k|\mathrm{Z}_{1:k})$, computed as

$$D_{k|k}(\mathrm{x}|\mathrm{Z}_{1:k}) = \int \delta_{\mathrm{X}}(\mathrm{x}) f_{k|k}(\mathrm{X}|\mathrm{Z}_{1:k}) \delta\mathrm{X} \tag{3.3}$$

where δ_{X} denotes the sum of Dirac deltas at elements of X. For the sake of notational convenience the left-hand side of (3.3) is abbreviated to $D_{k|k}(\mathrm{x})$ in the sequel. This function, known as the probability hypothesis density (PHD) [163], is a non-negative function whose integral $\int D_{k|k}(\mathrm{x})\mathrm{dx}$ yields the expected number of objects $\nu_k \in \mathbb{R}$. PHD filtering allows for joint detection and estimation of an unknown and varying number of objects and their individual states using the Bayesian prediction and update framework. Here, multi-object state X_k and observation Z_k are modeled as so-called random finite sets RFS, with randomness in set size as well as set element values [19], accommodating phenomena such as object initiation, clutter, and partitioning (spawning).

Formally stated, PHD filtering proceeds by iterating the sequence consisting of the prediction, formulated as

$$D_{k|k-1}(\mathrm{x}) = \gamma_{k|k-1}(\mathrm{x}) \ + \langle \beta_{k|k-1}(\mathrm{x}|\cdot) + p_{S,k|k-1}(\cdot)\pi_{k|k-1}(\mathrm{x}|\cdot), D_{k-1|k-1}(\cdot)\rangle \tag{3.4}$$

followed by the update, formulated as

$$D_{k|k}(\mathrm{x}) = (1 - p_{D,k}(\mathrm{x}))D_{k|k-1}(\mathrm{x}) \ + \sum_{\mathrm{z}\in\mathrm{Z}_k} \frac{p_{D,k}(\mathrm{x})g_k(\mathrm{z}|\mathrm{x})D_{k|k-1}(\mathrm{x})}{C_k(\mathrm{z}) + \langle p_{D,k}(\cdot)g_k(\mathrm{z}|\cdot), D_{k|k-1}(\cdot)\rangle} \tag{3.5}$$

where $\gamma_{k|k-1}$ denotes the intensity function of newborn objects from iteration $k-1$ to k, $\beta_{k|k-1}$ the spawning object transition density, $p_{S,k|k-1}$ the object survival probability, $\pi_{k|k-1}$ the single-object transition density, $p_{D,k}$ the object detection probability, g_k the single-object likelihood, C_k the clutter intensity function, and $\langle g(\cdot), f(\cdot)\rangle \equiv \int f(\xi)g(\xi)\mathrm{d}\xi$ (see e.g. [280] for details). An analytical solution to (3.4)-(3.5) is provided by the Gaussian-mixture PHD (GM-PHD) filter [280] but it is based on linear Gaussian assumptions regarding object birth and dynamics. A more general solution is offered by sequential Monte-Carlo PHD (SMC-PHD) filtering [216,281,302],

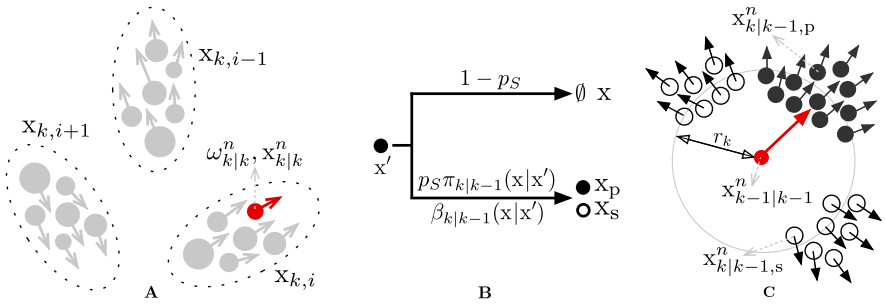


Figure 3.2: PHD filtering using a particle representation. (A) Each object i at iteration k has a state $\mathrm{x}_{k,i}$ that is represented by random particles $\mathrm{x}^n_{k|k}$ with corresponding weights $\omega^n_{k|k}$. (B) In the transition from iteration $k-1$ to k an object (x') may disappear ($\emptyset$), persist (x_p), or spawn (x_s) according to the corresponding transition functions. Here p_S is shorthand notation for $p_{S,k|k-1}(\mathrm{x}')$, since in practice a constant is used (Table 3.1). (C) For each particle a prediction $\mathrm{x}^n_{k-1|k-1} \rightarrow \mathrm{x}^n_{k|k-1}$ is made within radius r_k according to the transition functions for persistence (p) and spawning (s).

which approximates the PHD with a set of N random particles $\mathrm{x}^n_{k|k}$ and corresponding weights $\omega^n_{k|k}$ as

$$D_{k|k}(\mathrm{x}) \approx \sum_{n=1}^{N} \omega^n_{k|k} \delta_{\mathrm{x}^n_{k|k}}(\mathrm{x}) \tag{3.6}$$

so that the classic particle filtering scheme [12, 78, 215] can be applied.

3.2.3 PHD-filtering based neuron tracing

3.2.3.1 Definition and initialization

The multi-object filtering scheme proposed for neuron tracing defines the object state as an oriented location:

$$\mathrm{x} = [\mathrm{p}_\mathrm{x}, \mathrm{v}_\mathrm{x}] = [x, y, z, v_x, v_y, v_z] \tag{3.7}$$

where $\mathrm{p}_\mathrm{x} = [x, y, z]$ denotes the location and $\mathrm{v}_\mathrm{x} = [v_x, v_y, v_z]$ the local orientation of a tubular segment. Filtering starts from a set of N_0 seeds (Fig. 3.1) sampled from a seed pool consisting of the local maxima of the tubularity image $\tau(x, y, z)$ computed from the original image using Hessian-based multiscale line filtering [225] and min-max normalized to $[0, 1]$. Local maxima are sorted in descending order so that seeds with high tubularity (meaning high confidence in the underlying image structure being a neuron branch) are processed first. To avoid seeds being selected too close together, in other words to ensure good spatial coverage of the neuron with seeds, for each selected seed (while going from top to bottom of the sorted list) the seeds within a circular neighborhood with radius r_0 are ignored in the current round. If, after SMC-PHD filtering (described in following Sec. 3.2.3.2), the seed pool is not exhausted, a

new round is started by selecting a new set of seeds. During filtering, the observation consists of the location and corresponding tubularity value:

$$\mathrm{z} = [\mathrm{p}_\mathrm{z}, \tau_\mathrm{z}] = [x, y, z, \tau] \tag{3.8}$$

3.2.3.2 SMC-PHD algorithm

The proposed method implements neuron tracing by SMC-PHD filtering. It is based on an approximation of $D_{k|k}(\mathrm{x})$ in (3.6) using $N = \rho N_k$ particles, where N_k denotes the number of objects to be filtered, and ρ the number of particles per object. That is, the state of object i at iteration k, denoted $\mathrm{x}_{k,i}$, is represented by ρ random particles $\mathrm{x}^n_{k|k}$ with corresponding weights $\omega^n_{k|k}$ (Fig. 3.2A). The multi-object state transition in the prediction step (3.4) is a collection of single-object transitions (Fig. 3.2B) that are approximated with transitions at the particle level (Fig. 3.2C). More specifically, at the initial iteration $k = 0$, N_0 seeds are selected and ρ particles are sampled in a circular neighborhood with radius r_0 around each seed location using the tubularity value for importance sampling to determine the weights, resulting in the weighted particle set $\{\omega^n_{0|0}, \mathrm{x}^n_{0|0}\}_{n=1}^{\rho N_0}$. The initial local orientation of each particle, $\mathrm{v}_{\mathrm{x}^n_{0|0}}$, is the unit vector pointing from the seed location to the particle location $\mathrm{p}_{\mathrm{x}^n_{0|0}}$. Subsequently, the prediction (3.4) and update (3.5) steps are executed for iterations $k = 1, 2, 3, \ldots$, until convergence. The transition and observation models (described next) allow to incorporate application-specific knowledge in this process. At iteration k, the set of weighted particles $\{\omega^n_{k-1|k-1}, \mathrm{x}^n_{k-1|k-1}\}_{n=1}^{\rho N_{k-1}}$ from iteration $k-1$ is used to predict η new particles for each persistent and spawned object (Fig. 3.2C). In the update step (3.5), a set of observations $\{\mathrm{z}_{k,j}\}_{j=1}^{M_k}$ is used to update the predicted particle weights, followed by estimation of the states $\{\hat{\mathrm{x}}_{k,i}\}_{i=1}^{N_k}$. The detailed algorithm pseudo code of the introduced neuron tracing method is presented in Alg. 1.

Algorithm 1 Neuron tracing

$k = 0$ ▷ Initialize
$\{\omega^n_{0|0}, \mathrm{x}^n_{0|0}\}_{n=1}^{\rho N_0}$ ▷ Initial particle and observation set
$\{\hat{\mathrm{x}}_{0,i}\}_{i=1}^{N_0}$ ▷ Initial estimate
repeat
  $k = k + 1$
  $\mathrm{p}^n_i \sim h(\mathrm{p}|\hat{\mathrm{x}}_{k-1,i}) \quad n \in [1, \rho N_{k-1}]$ ▷ Draw observation particles
  $\mathrm{p}^n_{i,j} \in \mathcal{C}_j, \quad j \in [1, M_k], \quad n \in [1, |\mathcal{C}_j|]$ ▷ Cluster observation particles
  $\mathrm{z}_{k,j} = \left[\mathrm{p}^{\hat{n}}_{i,j}, \tau(\mathrm{p}^{\hat{n}}_{i,j})\right]$ ▷ Select representative sample
  $\mathrm{Z}_k = \{\mathrm{z}_{k,j}, \ldots, \mathrm{z}_{k,M_k}\}$ ▷ Construct observations
  $\{\omega^n_{k|k}, \mathrm{x}^n_{k|k}\}_{n=1}^{\rho N_k}, \nu_k, \{\hat{\mathrm{x}}_{k,i}\}_{i=1}^{N_k} \leftarrow \text{SMC-PHD}(\{\omega^n_{k-1|k-1}, \mathrm{x}^n_{k-1|k-1}\}_{n=1}^{\rho N_{k-1}}, \mathrm{Z}_k)$ ▷ Algorithm 2
until $[\nu_k] = 0$ ▷ $[\cdot] \equiv$ nearest integer

3.2.3.3 Transition model

In the prediction step (3.4), three types of objects are assumed: newborn, persisting, and spawned objects [280, 281]. By design, newborn objects are not considered in presented tracing algorithm, since the seeding is used, hence $\gamma_{k|k-1}(\mathrm{x}) = 0$.

Persisting objects in the current iteration correspond directly to existing objects in the previous iteration. In algorithm, the transition density for predicting persistent object x given object x' in the previous iteration, is calculated as

$$\pi_{k|k-1}(\mathrm{x}|\mathrm{x}';\kappa,r_k) = \frac{1}{\tilde{\pi}} e^{\frac{-(|\mathrm{p_x}-\mathrm{p_{x'}}|-r_k)^2}{2(r_k/3)^2}} \frac{e^{\kappa(\mathrm{v_x}\cdot\mathrm{v_{x'}})}}{2\pi I_0(\kappa)} \tag{3.9}$$

where $\tilde{\pi}$ is a normalization factor such that the sum of $\pi_{k|k-1}$ over $|\mathrm{p_x} - \mathrm{p_{x'}}| \leq 2r_k$ is unity, and I_0 is the zero-order Bessel function of the first kind. The first factor corresponds to a radial profile that peaks at the prediction step size r_k. The second factor is a circular normal distribution (von Mises) parametrized with the unit direction vector $\mathrm{v_{x'}}$ from the previous iteration and circular variance κ. Here, $\mathrm{v_x} = (\mathrm{p_x} - \mathrm{p_{x'}})/|\mathrm{p_x} - \mathrm{p_{x'}}|$, which connects the predicted location $\mathrm{p_x}$ with the location $\mathrm{p_{x'}}$ from the previous iteration. Particles $\mathrm{x}^n_{k|k-1,\mathrm{p}}$ (Fig. 3.2C) are drawn using $\pi_{k|k-1}$ as importance sampling function.

A spawned object is a new instance derived (spawned) from an existing object in the previous iteration. This allows dealing with bifurcations during tracing. In the showcased algorithm, the transition density for predicting a spawned object x given x' in the previous iteration, is calculated as

$$\beta_{k|k-1}(\mathrm{x}|\mathrm{x}';\kappa,r_k) = \frac{1}{\tilde{\beta}} e^{\frac{-(|\mathrm{p_x}-\mathrm{p_{x'}}|-r_k)^2}{2(r_k/3)^2}} \cdot \prod_{i=0}^{1} \left(1 - \frac{e^{\kappa(-1^i\mathrm{v_x}\cdot\mathrm{v_{x'}})}}{2\pi I_0(\kappa)}\right) \tag{3.10}$$

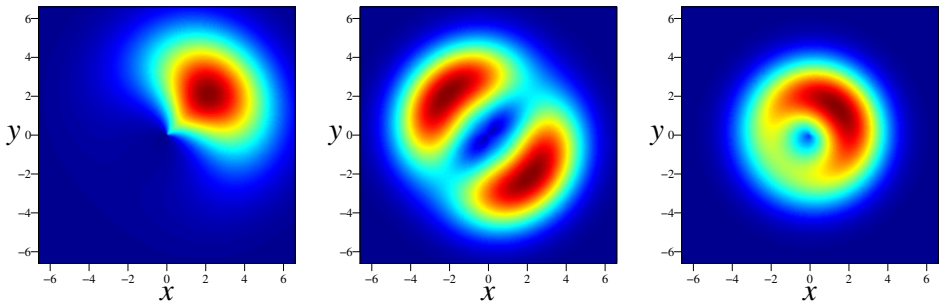
where $\tilde{\beta}$ is a normalization factor such that the sum of $\beta_{k|k-1}$ over $|\mathrm{p_x} - \mathrm{p_{x'}}| \leq 2r_k$ is unity. The first factor has the same form as in (3.9) and the second factor is the aggregate of the complementary circular normal distributions used for spawning objects in positive and negative direction. An example of the intensity profile of $\pi_{k|k-1}$ and $\beta_{k|k-1}$ is shown in Fig. 3.3. Particles $\mathrm{x}^n_{k|k-1,\mathrm{s}}$ (Fig. 3.2C) are drawn using $\beta_{k|k-1}$ as importance sampling function.

3.2.3.4 Observation model

In the update step (3.5), a set of observations $\{\mathrm{z}_{k,j}\}_{j=1}^{M_k}$ is used to update the predictions from (3.4). Observations have a corrective role as they carry information about the neuron centerline locations and corresponding tubularity values (3.8). The importance sampling function h

$$h(\mathrm{p}|\mathrm{x}';\kappa,r_k) = \frac{1}{\tilde{h}} e^{\frac{-(|\mathrm{p}-\mathrm{p_{x'}}|-r_k)^2}{2(r_k/3)^2}} \frac{e^{\kappa(\mathrm{v_p}\cdot\mathrm{v_{x'}})}}{2\pi I_0(\kappa)} \tau(\mathrm{p}) \tag{3.11}$$

is used to obtain the observations, where $\tilde{h}$ is a normalization factor such that the sum of h over $|\mathrm{p} - \mathrm{p_{x'}}| \leq 2r_k$ is unity, and $\mathrm{v_p} = (\mathrm{p} - \mathrm{p_{x'}})/|\mathrm{p} - \mathrm{p_{x'}}|$. The first two factors



a) $\pi_{k|k-1}(\mathrm{x}|\mathrm{x}')$ b) $\beta_{k|k-1}(\mathrm{x}|\mathrm{x}')$ c) $h(\mathrm{p}|\mathrm{x}')$

Figure 3.3: Transition densities (2D examples) for persistent a) and spawned b) objects with $z = 0$, $\mathrm{x}' = \left[0, 0, 0, \frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}, 0\right]$, $\kappa = 2$, and $r_k = 3$. c) Importance sampling used in the observation model without the tubularity component, $\tau(\mathrm{p}) = 1$, and $\kappa = 0.5$. Rainbow color coding is used running from blue (indicating low values) to red (indicating high values).

have the same form as in (3.9) but here κ is typically lower to make the update step more restrictive than the prediction step. The third factor is the normalized tubularity measure τ [225] at location p, which makes the observations correspond preferably to regions with high tubularity, which are indeed more likely to contain neuron structures.

To obtain the observations at iteration k, for each object i from the previous iteration a set of particles $\{\mathrm{p}_i^n\}_{n=1}^{\rho N_{k-1}}$ is drawn from h using $\mathrm{x}' = \hat{\mathrm{x}}_{k-1,i}$ (the object state estimate), with particle weight proportional to the tubularity value at that location. All these particles together are subsequently clustered in an unsupervised manner using mean-shifting [52], resulting in a set of clusters $\{\mathcal{C}_j\}_{j=1}^{M_k}$, with each cluster $\mathcal{C}_j$ having a subset $\{\mathrm{p}_{i,j}^n\}_{n=1}^{|\mathcal{C}_j|}$ of the particles. For each cluster, a representative sample $\mathrm{p}_{i,j}^{\hat{n}}$ is calculated using least-squares optimization,

$$\hat{n} = \arg\min_n \sum_{m \in [1, |\mathcal{C}_j|]} \theta(\mathrm{p}_{i,j}^m, \mathrm{p}_{\hat{\mathrm{x}}_{k-1,i}}, \mathrm{p}_{i,j}^n) \tag{3.12}$$

where $\theta(\mathrm{p}_0, \mathrm{p}_1, \mathrm{p}_2)$ denotes the squared Euclidean distance from point p_0 to the line segment defined by p_1 and p_2, calculated as

$$\theta(\mathrm{p}_0, \mathrm{p}_1, \mathrm{p}_2) = \begin{cases} |\mathrm{p}_0 - \mathrm{p}_1|^2 & \text{if } (\mathrm{p}_0 - \mathrm{p}_1) \cdot (\mathrm{p}_2 - \mathrm{p}_1) \leq 0 \\ |\mathrm{p}_0 - \mathrm{p}_2|^2 & \text{if } (\mathrm{p}_0 - \mathrm{p}_2) \cdot (\mathrm{p}_1 - \mathrm{p}_2) \leq 0 \\ \frac{|(\mathrm{p}_2 - \mathrm{p}_1) \times (\mathrm{p}_1 - \mathrm{p}_0)|^2}{|\mathrm{p}_2 - \mathrm{p}_1|^2} & \text{otherwise} \end{cases} \tag{3.13}$$

so that the line segment that best fits the cluster elements determines the selected location. From this the observation is obtained as $\mathrm{z}_{k,j} = [\mathrm{p}_{i,j}^{\hat{n}}, \tau(\mathrm{p}_{i,j}^{\hat{n}})]$. The process is

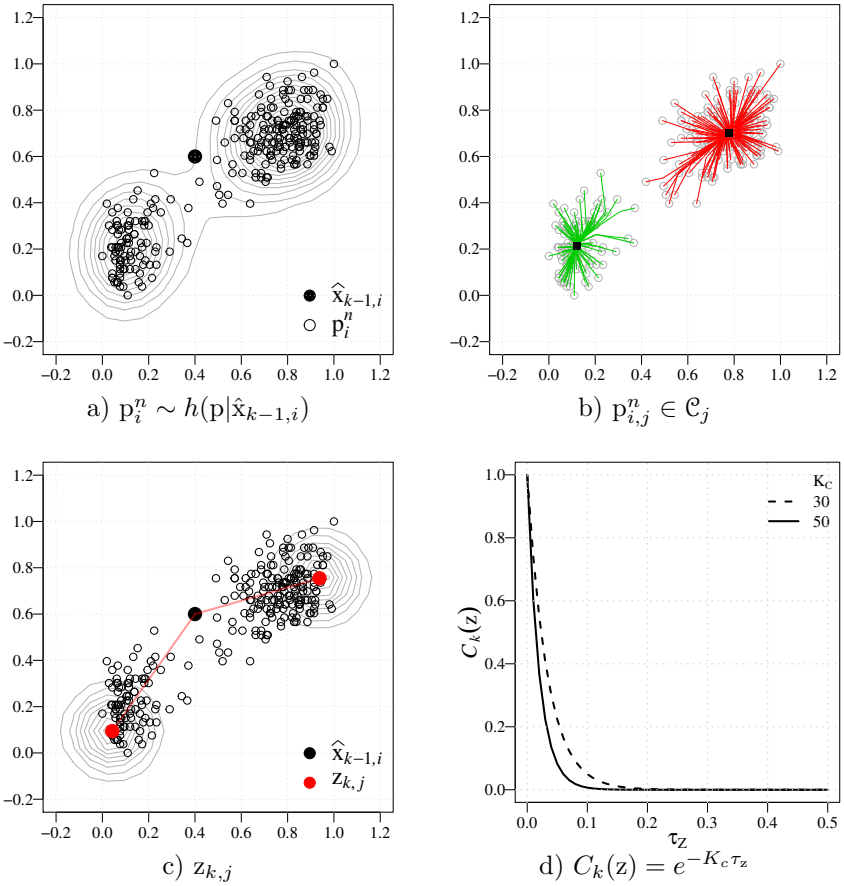


Figure 3.4: Formation of the observations (2D example). a) For each object i from iteration $k-1$, particles p_i^n are sampled from the importance sampling function h, using the state estimate $\hat{\mathrm{x}}_{k-1,i}$. The solid dot indicates the location of $\hat{\mathrm{x}}_{k-1,i}$ and the contours represent lines of equal particle weight. (B) The particles are processed by mean-shifting resulting in clusters $\mathcal{C}_j$ whose labeled particles are denoted as $\mathrm{p}_{i,j}^n$. (C) Each observation $\mathrm{z}_{k,j}$ is obtained from the representative cluster particle $\mathrm{p}_{i,j}^{\hat{n}}$ as described in the main text. Contours represent lines of equal observation likelihood. (D) The clutter intensity function.

illustrated in Fig. 3.4. For the single-object likelihood in Eq. 3.5 a Gaussian function centered at the spatial location of the observation is used, $g_k(\mathrm{z}|\mathrm{x}) = \exp(-|\mathrm{p_z} - \mathrm{p_x}|^2/2\sigma_\mathrm{z}^2)$, giving more importance to predictions closer to z. The clutter intensity function is defined as an exponential dependency on the observation tubularity value, $C_k(\mathrm{z}) = \exp(-K_c\tau_\mathrm{z})$, implying that the clutter increases as the tubularity value goes to zero. In practice, clutter plays a role in detecting terminal points, causing tracings with low particle weights (due to their proximity to regions with low tubularity values) to not be resampled and thus dropped after the update step.

Algorithm 2 SMC-PHD filtering

Input:$\{(\omega^n_{k-1|k-1}, \mathrm{x}^n_{k-1|k-1})\}_{n=1}^{\rho N_{k-1}}, \{\mathrm{z}_{k,j}\}_{j=1}^{M_k}$ ▷ $D_{k-1}(\mathrm{x})$ approx. observation Z_k
for $n = 1, \ldots, \rho N_{k-1}$ **do**
  for $m = 1, \ldots, \eta$ **do**
    $i = (n-1)\eta + m$
    Draw: $\mathrm{x}_{k|k-1,\mathrm{p}} \sim \pi_{k|k-1}(\mathrm{x}|\mathrm{x}^n_{k-1|k-1}) \rightarrow \mathrm{x}^i_{k|k-1,\mathrm{p}}$ ▷ Persistent object particles
    Compute: $\omega^i_{k|k-1,\mathrm{p}} = p_S \frac{1}{\eta}\omega^n_{k-1|k-1}$
    Draw: $\mathrm{x}_{k|k-1,\mathrm{s}} \sim \beta_{k|k-1}(\mathrm{x}|\mathrm{x}^n_{k-1|k-1}) \rightarrow \mathrm{x}^i_{k|k-1,\mathrm{s}}$ ▷ Spawning object particles
    Compute: $\omega^i_{k|k-1,\mathrm{s}} = p_S \frac{1}{\eta}\omega^n_{k-1|k-1}$
  end for
end for
$\{(\omega^n_{k|k-1}, \mathrm{x}^n_{k|k-1})\}_{n=1}^{S_k} = \{(\omega^n_{k|k-1,\mathrm{p}}, \mathrm{x}^n_{k|k-1,\mathrm{p}})\}_{n=1}^{\rho\eta N_{k-1}} \cup \{(\omega^n_{k|k-1,\mathrm{s}}, \mathrm{x}^n_{k|k-1,\mathrm{s}})\}_{n=1}^{\rho\eta N_{k-1}}$ ▷ Union of particle sets
for $n = 1, \ldots, S_k$ **do**
  Update: $\omega^n_{k|k} = (1 - p_D)\omega^n_{k|k-1} + \sum_{\mathrm{z}\in\mathrm{Z}_k} \frac{p_D g_k(\mathrm{z}|\mathrm{x}^n_{k|k-1})\omega^n_{k|k-1}}{C_k(\mathrm{z}) + \sum_{n=1}^{S_k} p_D g_k(\mathrm{z}|\mathrm{x}^n_{k|k-1})\omega^n_{k|k-1}}$
end for
$\nu_k = \sum_{n=1}^{S_k} \omega^n_{k|k}$ ▷ Cardinality calculation
Estimate: $\hat{\mathrm{x}}_{k,i} \leftarrow \{\omega^n_{k|k}, \mathrm{x}^n_{k|k-1}\}_{n=1}^{S_k}$ ▷ Mean-shift clustering
Resample: $N_k = [\nu_k], \{\omega^n_{k|k}, \mathrm{x}^n_{k|k-1}\}_{n=1}^{S_k} \rightarrow \{\omega^n_{k|k}, \mathrm{x}^n_{k|k}\}_{n=1}^{\rho N_k}, \omega^n_{k|k} = \nu_k/(\rho N_k)$
▷ Systematic resampling with ρ particles per object

3.2.3.5 Implementation details

Algorithms 1 and 2 provide a step-by-step overview of the introduced PHD-filtering based neuron tracing method. For testing purposes the method was implemented in Java as a plugin for ImageJ [1]. The method has several parameters for which default parameters are given in Table 3.1. Based on the acquired experience, most of them do not require extensive tuning and the default values were used for the showcased experiments. An important aspect of any SMC-based algorithm is to use a sufficient number of particles in the approximations. The conducted experiments indicate that values of 10-20 are sufficient for ρ and η since the objects of interest in

Parameter	Default	Description
K_c	30	Clutter intensity function decay
N_0	20	Number of seed points per round
p_D	0.9	Object detection probability
p_S	0.9	Object survival probability
r_k	3 voxels	Radial estimation step size
ρ	≥ 10	Number of particles per object
η	≥ 10	Number of predictions per particle
κ	2	Circular variance in (3.9) & (3.10)
	0.5	Circular variance in (3.11)

Table 3.1: Parameters of the proposed method with their default values. In accompanying implementation constants values were used for the object detection probability $p_D = p_{D,k}$ and the object survival probability $p_S = p_{S,k|k-1}$.

neuron-related applications are approximately 1D structures in 3D space and therefore are easily covered. Higher values can lead to higher accuracy and precision but at proportionally higher computational cost. The most important parameters are the numbers of seeds N_0 and rounds (Fig. 3.2) and in the experiments (described next) the performance of the presented algorithm was tested for different values of these parameters.

3.3 Results

3.3.1 Neuron data sets

For evaluating the performance of the proposed method for both 2D and 3D neuron tracing, three data sets (Fig. 3.5) were used. All datasets consist of real neuron images acquired with fluorescence microscopy. Two data sets are 3D image stacks from the DIADEM challenge [39]: neocortical layer-1 axons (NCL1A) with 16 image stacks and olfactory projection fibers (OPF) with 9 image stacks. The third data set (HCN) consists of 30 2D images of hippocampal neurons [255]. Together the data sets show a good variety of image contrast and structural complexity. Further details about the images can be retreived from the corresponding, cited publications.

3.3.2 Performance measures

The accuracy of the tracings produced by the proposed method was assessed by comparison with the gold-standard obtained by manual delineation of the neuron structures [102, 171]. To this end, two categories of evaluation measures are used. The first consists of measures summarizing the spatial Euclidean distances between the nodes of two tracings to be compared: the average spatial distance (SD), the average substantial spatial distance (SSD), and the fraction of nodes whose distance is at least the substantial distance (%SSD). Similar to previous studies using these measures [196], the substantial distance was set to 2 (pixels in 2D and voxels in 3D).

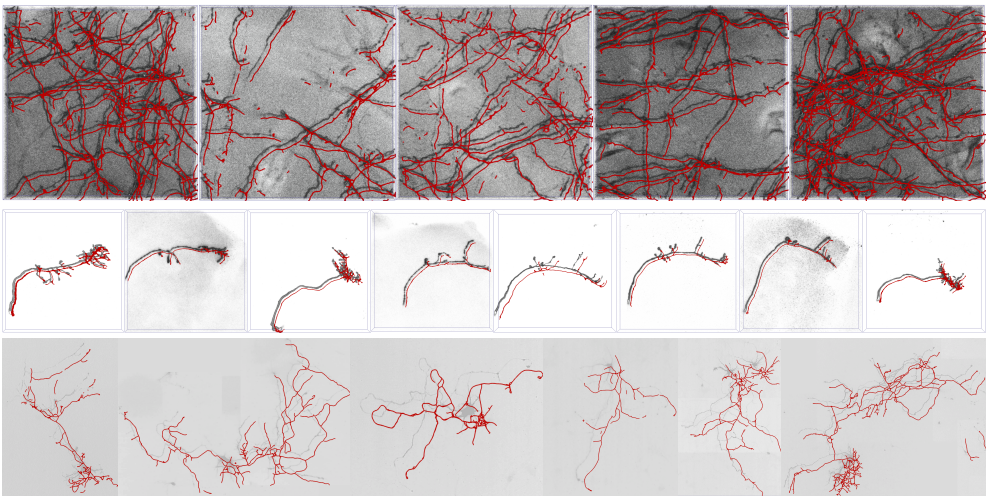


Figure 3.5: Example images with tracing results of the data sets used in the evaluation. Top row: NCL1A image stacks (volume rendered) showing a network of neocortical layer-1 axons. Middle row: OPF image stacks (volume rendered) showing olfactory projection fibers. Bottom row: HCN images showing hippocampal neurons. The tracings (overlaid in red) were obtained with the proposed method using 20 seeds and at most 10 rounds (up to 40 for the top row to capture more detail). For illustration purposes the image intensities are inverted in these visualizations compared to the originals, and the tracings are offset with respect to the neuron structures for better visual comparison.

The second category of evaluation measures are based on the numbers of true-positive (TP), false-positive (FP), and false-negative (FN) nodes according to the substantial distance. From these, it is straightforward to compute the recall, $R = TP/(TP+FN)$, and precision, $P = TP/(TP+FP)$, summarized using the F-score, $F = 2PR/(P+R)$. Prior to computing these measures the tracings (from the method and the gold-standard) were resampled with an equal step size of 1 pixel using Vaa3D [196].

3.3.3 Evaluation of method behavior

First, behavior of the presented method was evaluated as a function of the number of seeds and rounds. For this experiment P, R, and F values were measured for 1) a single round of filtering with different numbers of seeds and 2) multiple rounds of filtering using a fixed number of seeds. Since the algorithm showcased in this chapter operates probabilistically, the results of five repetitions of the experiment were averaged. The results for the NCL1A data set are shown in Fig. 3.7 and for the other data sets in Fig. 3.6. As expected, R and F generally increase, but P slightly decreases as the number of seeds and rounds increase, indicating an increase in the number of FP detections. The specific patterns may differ depending on the image content, but the observations indicate that as a function of the number of seeds, the increase of R and F levels off beyond about 40, therefore this value was subsequently used. As a function of the number of rounds, R and F level off after about 4 rounds, indicating there is no need in practice to run the method exhaustively on all possible seed points. This can be explained from the fact that seed selection proceeds from highest to lowest tubularity value, so that later seeds correspond to less and less valuable image structures, and the resulting tracings will be dropped due to low particle weights. Examples of traced neurons for the different data sets are shown in Fig. 3.5. As can be observed from the examples in the top row of the figure, images with more fuzzy and fragmented structures may require more rounds to capture more detail. Alternatively, a better tubularity filter may be needed.

3.3.4 Comparison with other methods

Next the performance of the introduced method (PHD) was compared with several alterative methods, namely all-path pruning (APP2) [291], NeuroGPS-Tree (GPS) [205], minimum spanning tree (MST) tracing as used in the BigNeuron project [190], and Neural Circuit Tracer (NCT) [56]. For each of these methods the scores were optimized by trying all possible parameter values on a grid. The results for the NCL1A data set are shown in Fig. 3.12, for the OPF data sets in Fig. 3.8 and HCN in Fig. 3.9. Further observation shows that showcased method (results indicated in red) performs comparably or even better than the state-of-the-art methods. This suggests there may indeed be an advantage in using probabilistic approaches such as the one proposed in this work. It is also noticeable that NCT (results indicated in blue), while performing superiorly in most cases, required significant user interaction and manual correction to enable export of the tracings to the standard SWC file format used throughout the evaluations. Thus the results of this method include a high level of expert input and could serve as a reference. All other methods including

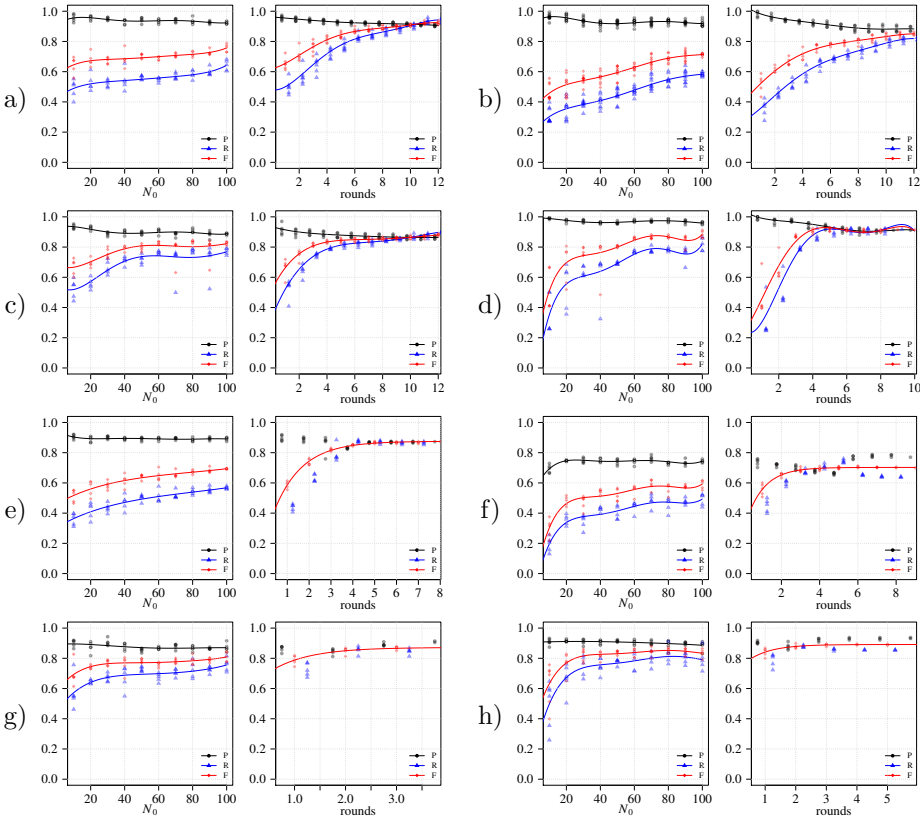


Figure 3.6: Performance as a function of numbers of seeds and rounds for four example cases from the OPF (a-d) and the HCN (e-h) data set. Similar trends were observed for all cases in the respective data sets. Left panel per case: Precision (P), recall (R), and F-score (F) after one round initialized with different numbers of seeds (N_0). Right panel per case: The scores after multiple rounds with a fixed number of seeds ($N_0 = 40$). Fifth-order polynomial curves were fit to the data to show approximate trends.

the presented one were fully automatic after parameter selection.

To further demonstrate the advantage of the presented method over the others in challenging situations, the case when neuron fibers meet, run closely parallel to each other for some distance, and then diverge again was studied. In order to analyze the behavior of the different methods in a controlled manner, with increasing distance between the fibers, the images with two fibers of similar intensity and scale were synthesized. The results, shown in supplementary Fig. 3.10, demonstrate that the method introduced in this chapter (PHD), similar to GPS, yields more faithful tracings than APP2 and MST. NCT was not included in these experiments for reasons mentioned above. Not surprisingly, all methods break down when the fibers overlap completely. In addition, even more challenging case was created, with three fibers of different intensity and scale. The results, shown in supplementary Fig. 3.11, illustrate that the proposed method outperforms even the best alternatives.

In terms of computational efficiency it turned out difficult to directly compare the methods. This was mainly due to the use of different programming languages (Java versus C++) and the varying efficiencies of underlying software libraries used on the different operating systems which were considered and used for the computation (Linux Ubuntu and Mac OS). Moreover further observations suggest that the absolute as well as the relative execution times of the different methods varied widely depending on the image content. Generally, APP2 method was found to be the fastest (on the order of seconds per image), and PHD up to about one order of magnitude slower, while GPS and MST were either slower or faster than PHD depending on the configuration. NCT is ignored here for mentioned reasons. From these observations it

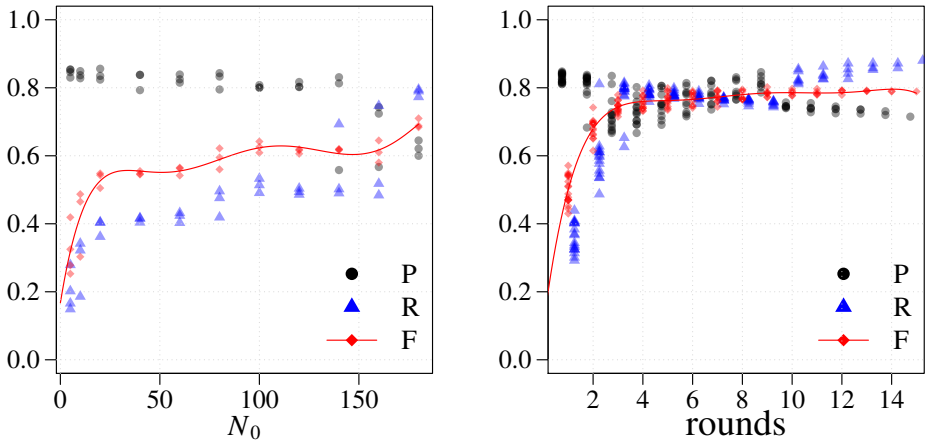


Figure 3.7: Performance of the proposed method as a function of numbers of seeds and rounds for an example image stack from the NCL1A data set. Similar trends were observed for all stacks in the data set. Left panel: Precision (P), recall (R), and F-score (F) after one round initialized with different numbers of seeds (N_0). Right panel: The scores after multiple rounds with a fixed number of seeds ($N_0 = 40$). Fifth-order polynomial fitting was used to show the approximate F-score trend.

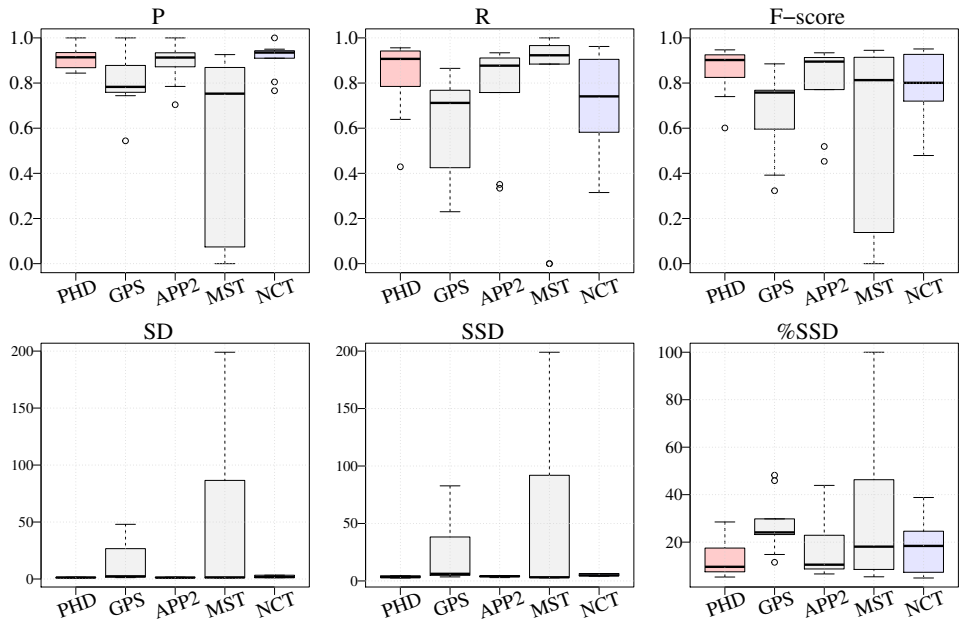


Figure 3.8: Performance comparison of the proposed method with several other methods on the OPF data set. For each method and each measure, the plotted box indicates the 25-75 percentile, the horizontal bar indicates the median score, and the whiskers and outliers are drawn using the default settings of R.

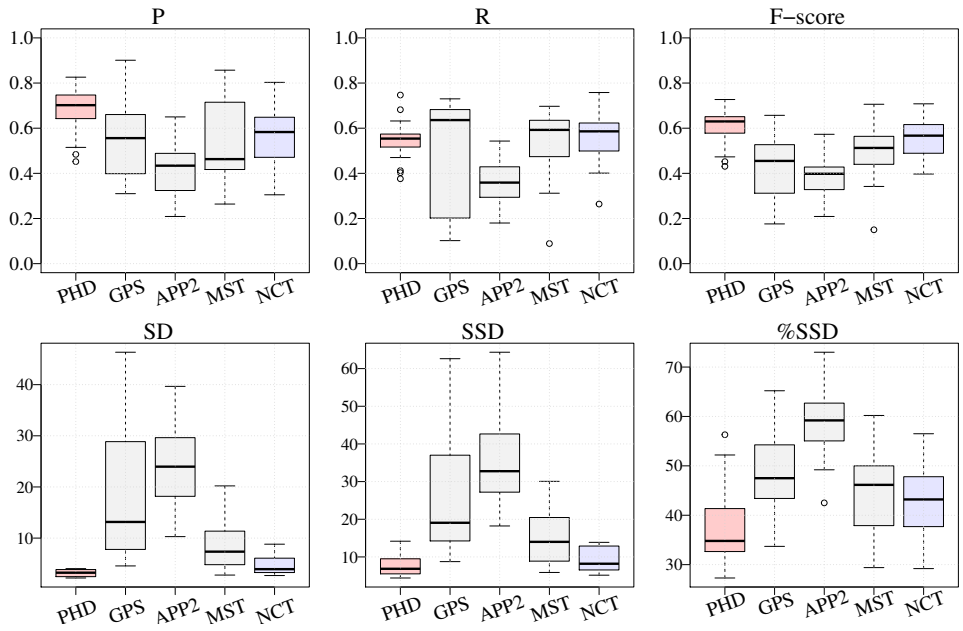


Figure 3.9: Performance comparison of the proposed method with several other methods on the HCN data set. For each method and each measure, the plotted box indicates the 25-75 percentile, the horizontal bar indicates the median score, and the whiskers and outliers are drawn using the default settings of R.

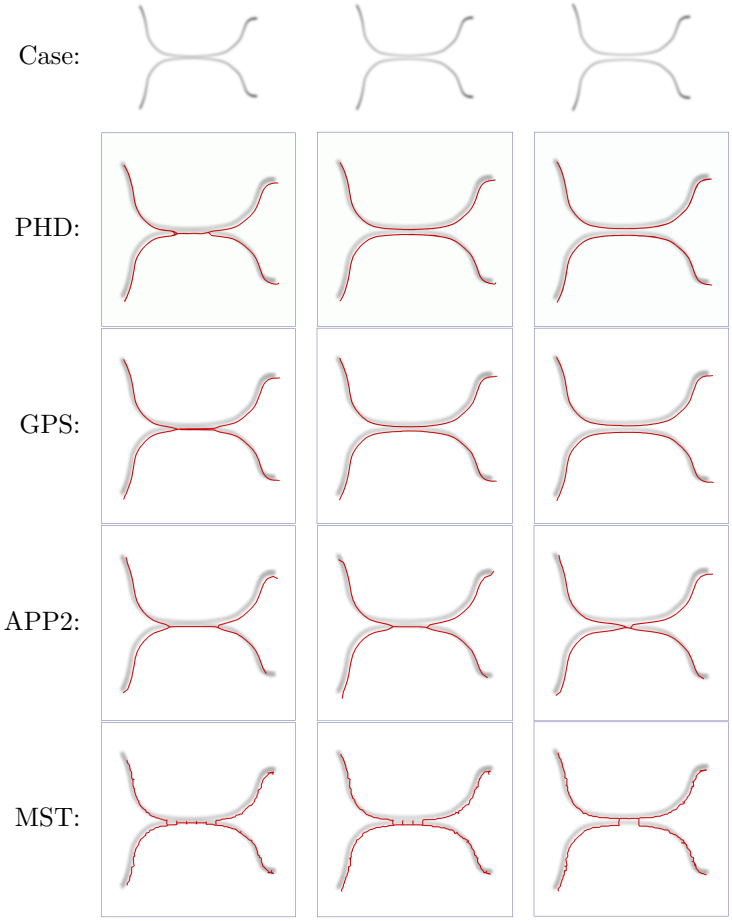


Figure 3.10: Ability of the tested methods to separate two fibers of similar intensity and scale running closely in parallel. The examples show cases with gradually increasing distance between the fibers: overlap (left column), just separated (middle column), and clearly separated (right column). The tracing results of PHD, GPS, APP2, MST are overlaid (with slight offset) in red color.

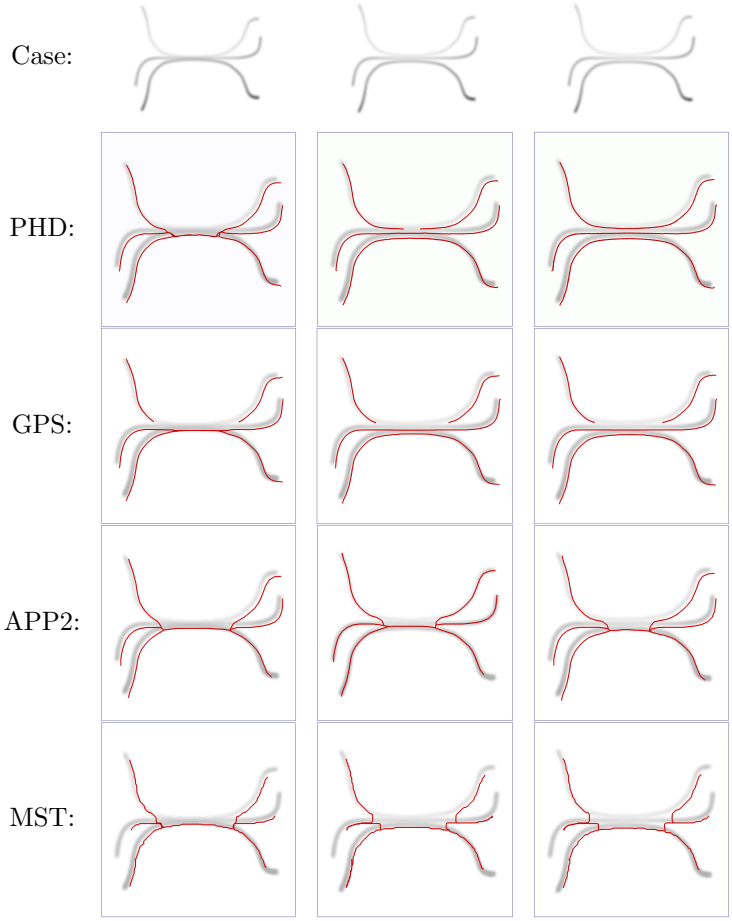


Figure 3.11: Ability of the tested methods to separate three fibers with different intensity and scale running closely in parallel. The examples show cases with gradually increasing distance between the fibers: overlap (left column), just separated (middle column), and clearly separated (right column). The tracing results of PHD, GPS, APP2, MST are overlaid (with slight offset) in red color.

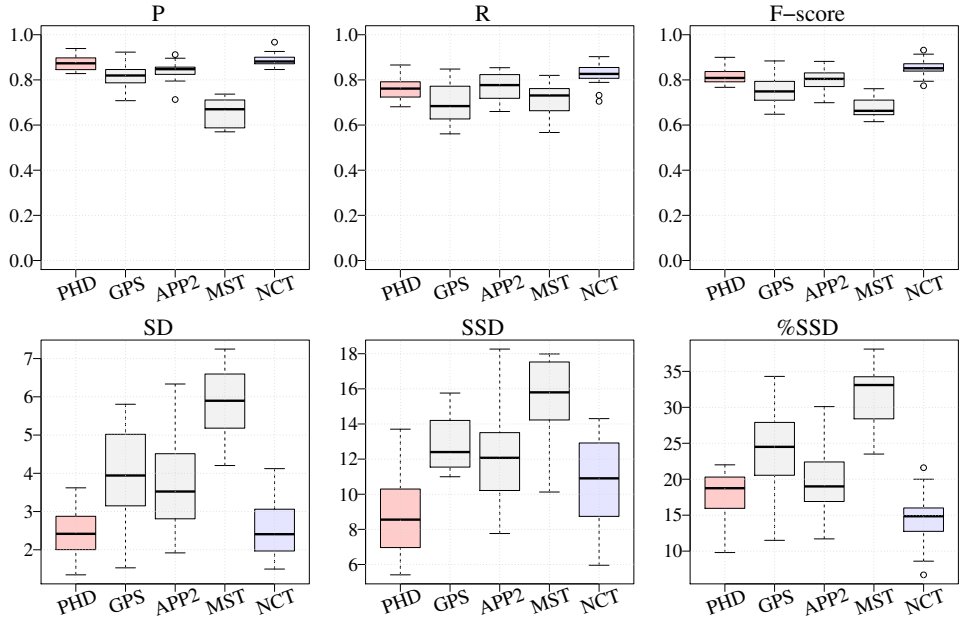


Figure 3.12: Performance comparison of the method with several other methods on the NCL1A data set. For each method and each measure, the plotted box indicates the 25-75 percentile, the horizontal bar indicates the median score, and the whiskers and outliers are drawn using the default settings of R.

is possible to draw a conclusion that the efficiency of showcased method is comparable to the state of the art.

3.4 Conclusions

A new method for tracing the branch centerlines of neurons based on Bayesian multi-object tracking using probability hypothesis density (PHD) filtering was presented. The method is able to simultaneously trace out multiple neuron structures in a probabilistic fashion so that the same neuron segments may be covered multiple times and are thus supported by more evidence. PHD filtering solves the computational problems of direct Bayesian multi-object tracking and allows convenient handling of bifurcations and terminations during the tracing process by modeling of spawned objects and observation clutter. The results of experiments on various fluorescence microscopy image data sets of real neurons showed that the proposed method performs comparably or better than alternative state-of-the-art neuron tracing methods.

The current version of the proposed method is initialized with seed points sampled from the local maxima (from highest to lowest) of the tubularity filter response. This is a rather rudimentary approach that may sometimes result in missed branches (false negatives). Ideally, seeds should be strategically distributed so that they cover as

many branches of the neuron structure as possible while avoiding background artifacts, and this is an important topic for further research. In addition, the current mechanism responsible for trace termination, based on the clutter term of the PHD filter, relies strongly on the tubularity score and thus is sensitive to local interruptions in neuron staining. This could be remedied by using a better tubularity filter and/or refining the clutter model. Thus, the future work would involve further study over the possibility of further improvements achieved using different transition and observations models. One of the future aims is also to extend the method to perform local branch radius estimation during tracing in order to obtain complete neuron reconstructions.

Software implementing the proposed neuron tracing method was written in the Java programming language as a plugin for the ImageJ platform. Source code is freely available for non-commercial use at `https://bitbucket.org/miroslavradojevic/phd`.

Chapter Four

Automated neuron reconstruction from 3D fluorescence microscopy images using sequential Monte Carlo estimation

Microscopic images of neuronal cells provide essential structural information about the key constituents of the brain and form the basis of many neuroscientific studies. Computational analyses of the morphological properties of the captured neurons require first converting the structural information into digital tree-like reconstructions. Many dedicated computational methods and corresponding software tools have been and are continuously being developed with the aim to automate this step while achieving human-comparable reconstruction accuracy. This pursuit is hampered by the immense diversity and intricacy of neuronal morphologies as well as the often low quality and ambiguity of the images. This chapter presents a novel method developed in an effort to improve the robustness of digital reconstruction against these complicating factors. The method is based on probabilistic filtering by sequential Monte Carlo estimation and uses prediction and update models designed specifically for tracing neuronal branches in microscopic image stacks. Moreover, it uses multiple probabilistic traces to arrive at a more robust, ensemble reconstruction. The proposed method was evaluated on fluorescence microscopy image stacks of single neurons and dense neuronal networks with expert manual annotations serving as the gold standard, as well as on synthetic images with known ground truth. The results indicate that proposed method performs well under varying experimental conditions and compares favorably to state-of-the-art alternative methods.

Based upon: M. Radojević, E. Meijering, "Automated neuron reconstruction from 3D fluorescence microscopy images using sequential Monte Carlo estimation", *Neuroinformatics*, *in press*, 2018.

4.1 Introduction

The brain is regarded as one of the most complex and enigmatic biological structures. Composed of an intricate network of tree-shaped neuronal cells [13], together forming a powerful information processing unit, it performs a myriad of functions that are essential to living organisms [132]. Obtaining a blue print of the architecture of this network, including the morphologies and interconnectivities of the neurons in various subunits, helps to understand how the brain works [14, 63, 75], including how neurodegenerative disease processes alter its function. A key instrument in this endeavor is microscopic imaging, as it allows detailed visualization of neuronal cells in isolation and in tissue, thus providing the means to study their structural properties quantitatively [232].

Quantitative measurement and statistical analysis of neuronal cell and network properties from microscopic data rely on the ability to obtain accurate digital reconstructions of the branching structures [115] in the form of a directional tree of connected nodes [15]. The ever increasing amount of available image data calls for automated computational methods and software tools for this purpose, as manual delineation of neurons is extremely cumbersome even in single image stacks, and is downright infeasible in processing large numbers of images [232, 260]. Automating neuron reconstruction requires solving fundamental computer vision problems such as detecting and segmenting tree-like image structures [2, 76, 169]. This is complicated by the large diversity of neuron types, imperfections in cell staining, optical distortions, inevitable image noise, and other causes of ambiguity in the image data. Consequently, with the current state-of-the-art, manual proof-editing of automatically obtained digital reconstructions is often necessary [193]. Recent international initiatives such as the DIADEM challenge [102] and the BigNeuron project [190, 194] have catalyzed research in automated neuron reconstruction but have also clearly revealed that further improvement is still very much needed before computers can fully replace manual labor in performing this task.

The aim of the methodology presented in this chapter is to contribute to the developments in the field by proposing a novel fully automated neuron reconstruction method based on probabilistic filtering techniques. Starting from seed points that have a high probability of being centered at neuronal branches, presented method recursively traces these branches by sequential Monte Carlo estimation, using state transition and measurement models designed specifically for this purpose. This results in a series of possibly overlapping but probabilistically independent estimates of the branches, which are subsequently combined into a refined estimate of the actual branch centerlines using mean-shifting. The early versions of the method were presented at conferences [209, 211] and one implementation of it (named Advantra) donated for inclusion in the BigNeuron benchmarking study [190, 194]. Since then the method has been improved and its software implementation and have significantly extended its experimental evaluation. A detailed description of the method is provided in this chapter, its implementation, and the experimental results, and show that it performs favorably compared to several state-of-the-art neuron reconstruction methods from the BigNeuron project as well as an alternative probabilistic method [208].

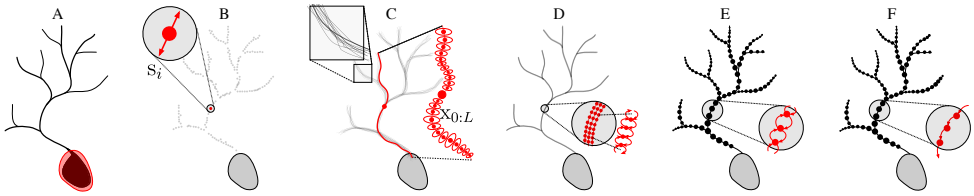


Figure 4.1: Schematic overview of the six main steps of the proposed method: (A) soma extraction, (B) seed extraction, (C) branch tracing, (D) trace refinement, (E) node grouping, (F) tree construction.

4.2 Related work

Early methods and tools for digital neuron reconstruction were semi-automatic and required extensive manual intervention for their initialization and operation or the curation of faulty results [45, 103, 104, 167]. With the increasing capabilities of computers it became possible to store and process 3D images of neurons [25, 58]. More recently, the state-of-the-art in the field has moved towards full automation of neuron reconstruction, and various freely available software tools are now available for this purpose [155, 189, 196, 197], though the need for flexible editing tools has remained unabated [71, 159].

Neuron reconstruction methods typically have a modular design where each module or stage of the processing pipeline deals with different structural objects. Depending on the subproblems being solved, modules can operate independently, or work together for example to combine local and global processing, possibly requiring multiple iterations. Several subproblems that can be identified in the literature include image prefiltering and segmentation [178, 247, 269, 310], soma (cell body) detection and segmentation [204], landmark points extraction [5,55,212,259,283], neuron arbor tracing [144, 153, 208, 291, 307], and assembling the final tree-like graph structure [269, 300, 309]. The techniques for solving each of these subproblems are briefly reviewed in the remainder of this section. Since presenting a novel method is the primary goal of this chapter, the review is not meant to be exhaustive, but to put the new method into context.

The pool of neuron reconstruction methods is very diverse [2, 76, 169, 190] but there are also many commonalities. For example, image prefiltering to enhance tubular structures is typically carried out using Hessian or Jacobian based processing [5, 283, 293, 300]. And to cope with uneven staining, adaptive thresholding [310], perceptual grouping [181], and vector field convolution [179] have been used. For image segmentation (separating foreground from background), a wide variety of methods has been proposed, including the use of feature-based classifiers [51, 131, 269], tubularity based supervised regression [247], and even deep learning [149]. The general difficulty of supervised methods, however, is their need for extensive manual annotation for training to arrive at usable segmentation models. This is avoided in the proposed method by using carefully designed explicit models.

For the detection and segmentation of the neuronal somas, which typically have

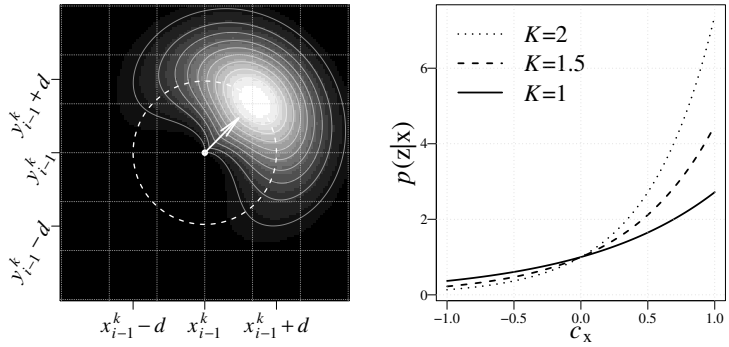
a much larger diameter than the dendritic and axonal branches, a simple and efficient solution is to apply morphological closing and adaptive thresholding [294]. An alternative is to use shape fitting approaches [204]. Next, to initialize and/or guide the segmentation of the arbor, landmark points are often extracted using image filters that specifically enhance tubular structures [55,212,259,269,283], a popular one being the so-called "vesselness filter" [92]. Classical approaches have been adopted for soma and seed point detection throughout the proposed method, as detailed in the next section.

Segmentation or tracing of all branches of the dendritic and axonal trees is the main challenge of the reconstruction problem. A widely used approach to overcome the difficulties caused by imperfect staining and image noise is to use techniques that find globally optimal paths between seed points by minimizing a predefined cost function [155, 171, 192, 205]. But many other concepts have been proposed as well, including model fitting [228, 307], contour extraction [144], active contour segmentation [160, 283], level-set or fast-marching approaches [24, 291], path-pruning from oversegmentation [192], distance field tracing [295], marching rayburst sampling [175], marked point processing [23], iterative back-tracking [153], and learning based approaches [51, 96, 222]. The works presented in previous chapters of this thesis [208,209,211] have shown the great potential of probabilistic approaches to neuron tracing which formed the basis for the new fully automated neuron reconstruction method presented and evaluated in the next sections.

The final aspect of neuron reconstruction is the assembling of the complete neuronal tree structure from possibly many partial or overlapping traces and putting it into a format that is both representative and suitable for further automated analysis. This is typically solved by graph optimization strategies such as the minimum spanning tree (MST), the alternative K-MST [107,269], or integer programming [267]. To deal with very large data sets it has also been proposed to assemble the 3D graph representation through tracing in 2D projections and applying reverse mapping [309]. However, with the advent of sophisticated assemblers such as UltraTracer [198], it is possible to extend any base tracing algorithm to deal with arbitrarily large volumes of neuronal image data [198]. Therefore, the projections are not used in the method proposed in this chapter. Instead, the tracing is performed in the original image (sub)volumes. And to obtain the graph representation a new approach is proposed to refining and grouping the individual traces.

4.3 Proposed method

The pipeline of the proposed method consists of six steps (Fig. 4.1) described in detail in the following subsections. The basic underlying assumption suggests that image stacks contain a single neuron (one soma) or just an arbor (no soma) as in the DIADEM [39] and BigNeuron data [190]. In short, the soma is extracted first and a set of seeds, which serve to initialize the probabilistic branch tracing scheme. The resulting traces are iteratively refined and their corresponding nodes spatially grouped into a representative node set that is traversed to form the final reconstruction.



(a) $\mathrm{x}_i^k \sim p(\mathrm{x}_i|\mathrm{x}_{i-1}^k)$ (b) $p(\mathrm{z}|\mathrm{x}) = \mathrm{e}^{Kc_\mathrm{x}}$

Figure 4.2: Functions used in the prediction and update steps of the SMC filtering: (a) the prediction importance sampling distribution (for ease of visualization a 2D example is given) and (b) the measurement likelihood function for different values of K.

4.3.1 Soma extraction

The soma typically has a considerably larger diameter than the individual branches of the neuronal arbor (Fig. 4.1A). Thus it can be easily extracted using morphological filtering operations [294]. Specifically, in the method showcased in this chapter, grayscale erosion is applied to remove all branches and leave only the (eroded) soma. To this end, the radius r_s of the structuring element needs to be larger than the largest expected branch radius in a given data set, and smaller than the expected soma radius. The resulting image is then smoothed using a Gaussian filter with standard deviation equal to r_s and segmented using max-entropy thresholding [212] to obtain a blob corresponding to the soma. For computational efficiency both the erosion and the Gaussian smoothing operation are carried out by separable filtering. In this chapter the soma is modeled in the final graph representation of the neuron as a single spherical node with position equal to the centroid of the segmented blob and radius equal to the average distance of the blob voxels to the centroid. Alternatively, the soma could be modeled with a set of nodes that together represent the blob as accurately as needed, but in introduced applications this is not needed.

4.3.2 Seed extraction

To initialize the branch tracing, a set of seed points is extracted (Fig. 4.1B). These seeds are points with very high likelihood of being centered on a branch. In presented method, this likelihood is estimated using a Hessian-based multiscale tubularity filter [92]. For each voxel location $\mathrm{p} = [x, y, z]$ this filter yields not only an estimate of the tubularity of the local image structure, but also an estimate of the structure's orientation $\mathrm{v} = [v_x, v_y, v_z]$ as derived from the Hessian eigenvector corresponding to the smallest absolute eigenvalue, and an estimate of its spatial scale as derived from the Gaussian σ at which the filter yields the highest tubularity value. From the

resulting tubularity map, seeds $\mathrm{s}_i = [\mathrm{p}_i, \mathrm{v}_i, \sigma_i]$ are selected whose tubularity value is the highest in a cylindrical neighborhood with radius $3\sigma_i$, centered at p_i, and oriented along v_i. A find-maxima function ported from ImageJ is used here. It applies a noise tolerance τ to prune insignificant local maxima [88].

4.3.3 Branch tracing

For each seed s_i, this method traces the local image structure in two directions, $+\mathrm{v}_i$ and $-\mathrm{v}_i$, producing a pair of local traces (Fig. 4.1C). A trace is considered to consist of a sequence of hidden states, $\mathrm{x}_{0:L} = (\mathrm{x}_0, \ldots, \mathrm{x}_L)$, where x_0 is the initial state extrapolated from the seed s_i, and x_L is the last state of the trace. Similar to the seeds, the states $\mathrm{x}_i = [\mathrm{p}_i, \mathrm{v}_i, \sigma_i]$ contain estimates of the position $\mathrm{p}_i = [x_i, y_i, z_i]$, the direction $\mathrm{v}_i = [v_{x_i}, v_{y_i}, v_{z_i}]$, and the scale σ_i of the underlying neuron branch. The states are estimated sequentially in a probabilistic fashion using Bayes' rule:

$$p(\mathrm{x}_i|\mathrm{z}_{0:i}) \propto p(\mathrm{z}_i|\mathrm{x}_i) \int p(\mathrm{x}_i|\mathrm{x}_{i-1}) p(\mathrm{x}_{i-1}|\mathrm{z}_{0:i-1}) \mathrm{d}\mathrm{x}_{i-1} \tag{4.1}$$

where $p(\mathrm{x}_i|\mathrm{z}_{0:i})$ is the posterior probability distribution of the state x_i given measurements $\mathrm{z}_{0:i}$ from the first to the current iteration, $p(\mathrm{x}_i|\mathrm{x}_{i-1})$ is the state transition prior, and $p(\mathrm{z}_i|\mathrm{x}_i)$ is the likelihood of measuring z_i given state x_i. It is assumed that the state transition is a Markovian process and the measurements are independent. To allow for nonlinearities in the process, the estimation problem (4.1) is solved using sequential Monte Carlo (SMC) filtering [77], also known as particle filtering [12]. Here the posterior is approximated using a set of N samples x_i^k with corresponding weights w_i^k as:

$$p(\mathrm{x}_i|\mathrm{z}_{0:i}) \approx \sum_{k=1}^{N} w_i^k \delta(\mathrm{x}_i - \mathrm{x}_i^k) \tag{4.2}$$

where the weights are normalized so that $\sum_k w_i^k = 1$.

Each iteration in SMC filtering consists of a prediction step and an update step. In the prediction step, given the samples x_{i-1}^k from the previous iteration, N new samples x_i^k are drawn using the state transition prior. The importance sampling distribution that is used for this is (Fig. 4.2a):

$$p(\mathrm{x}_i|\mathrm{x}_{i-1}^k) = \begin{cases} \dfrac{\exp\left(\kappa\, \mathrm{v}_i \cdot \mathrm{v}_{i-1}^k - \frac{(d_i - d)^2}{2(d/3)^2} - \frac{(\sigma_i - \sigma_{i-1}^k)^2}{2\zeta^2}\right)}{2\pi I_0(\kappa)\eta} \\ \qquad \text{for } d_i \leq 2d \wedge \sigma_i \leq 3\zeta \\ 0 \qquad \text{otherwise} \end{cases} \tag{4.3}$$

where I_0 denotes the zero-order Bessel function of the first kind, κ is the circular variance parameter, η is a normalization factor that makes the prediction over all N samples integrate to unity, $d_i = ||\mathrm{p}_i - \mathrm{p}_{i-1}^k||$ is the Euclidean distance between the predicted position and the sample position in the previous iteration, d is the tracing step size, and ζ the scale variance parameter. Each predicted state is assigned a unit direction $\mathrm{v}_i = (\mathrm{p}_i - \mathrm{p}_{i-1}^k)/||\mathrm{p}_i - \mathrm{p}_{i-1}^k||$ defined by two consecutive positions. And

$\sigma_i - \sigma_{i-1}^k$ represents the difference in scales, which contributes to the importance sampling function by a Gaussian component, giving a higher value to state samples that retain the scale.

In the update step, the newly drawn samples are updated using the following likelihood function (Fig. 4.2b):

$$p(\mathrm{z}|\mathrm{x}) = e^{Kc_\mathrm{x}} \tag{4.4}$$

where K determines the sensitivity to the normalized cross-correlation $c_\mathrm{x} \in [-1, 1]$, which quantifies the similarity of the underlying image structure for $\mathrm{x} = [\mathrm{p}, \mathrm{v}, \sigma]$ to a cylindrical template model with Gaussian profile (Fig. 4.3):

$$c_\mathrm{x} = \frac{\sum_{k,l,m} \big(I(\mathrm{p}') - \bar{I}\,\big)\,\big(G_\sigma - \bar{G}\,\big)}{\sqrt{\sum_{k,l,m} \big(I(\mathrm{p}') - \bar{I}\,\big)^2 \sum_{k,l,m} \big(G_\sigma - \bar{G}\,\big)^2}} \tag{4.5}$$

$$\mathrm{p}' = \mathrm{p}'(k,l,m) = \mathrm{p} + k\mathrm{u} + l\mathrm{w} + m\mathrm{v} \tag{4.6}$$

$$G_\sigma = G_\sigma(k,l,m) = G_\sigma(k,l) = \exp\big(-\big(k^2 + l^2\big)/2\sigma^2\big) \tag{4.7}$$

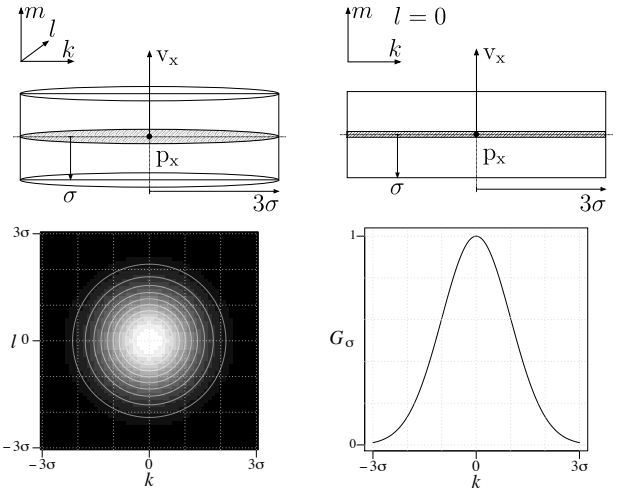
where (k, l, m) are the template coordinates, which transform to p' in image coordinates since the template is centered at p and is oriented in the direction v and has scale σ of x, and by definition $\mathrm{u} \perp \mathrm{v}$, $\mathrm{w} \perp \mathrm{v}$, and $\mathrm{u} \perp \mathrm{w}$. The summation is limited to $\lfloor -3\sigma \rfloor \leq k, l \leq \lceil 3\sigma \rceil$ and $\lfloor \sigma \rfloor \leq m \leq \lceil \sigma \rceil$ which corresponds to the spatial extent of the template. $\bar{I}$ and $\bar{G}$ denote the mean of the image intensities and of the template intensities, respectively, within the mentioned limits. The value of c_x is independent of intensity scalings and offsets and thus provides us with a robust measure of structural resemblance, which may range from -1 (inverse correlation), to 0 (no correlation), to $+1$ (full correlation). The weights of the samples are updated accordingly as:

$$w_i^k \propto w_{i-1}^k p(\mathrm{x}_i^k|\mathrm{x}_{i-1}^k)\,\mathrm{e}^{Kc_{\mathrm{x}_i^k}} \tag{4.8}$$

and renormalized so that $\sum_k w_i^k = 1$. To avoid weight deterioration, systematic resampling [135] is performed each time the effective sample size N_eff [137] falls below 80% of N. The final state estimate after each iteration i, which constitutes a node of the trace, is computed from the weighted samples as the centroid:

$$\hat{\mathrm{x}}_i = \sum_k w_i^k \mathrm{x}_i^k \tag{4.9}$$

Filtering is terminated if the average correlation value $\sum_k c_{\mathrm{x}_i^k}/N$ drops below the threshold c_min, indicating the end of the underlying neuron branch in the image, or if the iteration limit L is reached. Since the filtering is done for each seed, and in both (opposite) directions, the same neuron branch may be traced many times over, but in a probabilistically independent way, providing accumulating evidence about the presence and location of the branches. However, to avoid excessive over-tracing and to reduce the computation time, the method also monitors the node density D_n per image volume unit $n \times n \times n$ and terminate the tracing if the density in the current position exceeds the limit δ_n.



(a) 3D (b) 2D

Figure 4.3: Cylindrical template intensity model G_σ. The model has a Gaussian profile in coordinates k and l and is constant in coordinate m. Both the 3D (a) and the 2D (b) version is shown.

4.3.4 Trace refinement

After the tracing step, each neuron branch may have multiple corresponding traces, and each trace node has bidirectional links to neighboring nodes (Figs. 4.1D and 4.4A) to allow trace traversal in any of the possible directions in the final tree construction step. Denoting the total number of traces by T, and the nodes of any given trace t by n_i^t, $i = 1, \ldots, M^t$, it is possible to write the complete set of nodes as:

$$\mathcal{N} = \left\{ \left\{ n_1^1, \ldots, n_{M^1}^1 \right\}, \ldots, \left\{ n_1^T, \ldots, n_{M^T}^T \right\} \right\} \tag{4.10}$$

but in the sequel, the elements of $\mathcal{N}$ are written more generally as n_k, $k = 1, \ldots, M$, where $M = \sum_{t=1}^{T} M^t$. Each node n_k contains an estimate of the center position (x, y, z) and the cross-sectional radius (r) of the underlying branch structure, as well as the cross-correlation (c) with the cylindrical Gaussian template model, and a set $(\mathcal{I})$ containing the indices in $\mathcal{N}$ of the neighboring nodes:

$$n_k = \{x_k, y_k, z_k, r_k, c_k, \mathcal{I}_k\} \tag{4.11}$$

where $\mathcal{I}_k$ has either two elements (in the case of a body node) or just one (in the case of a terminal node).

The goal of the trace refinement step is to exploit the cumulative evidence provided by the over-tracing in the previous step to improve the individual node estimates. Specifically, each node n_k is updated to:

$$\bar{n}_k = \{\bar{x}_k, \bar{y}_k, \bar{z}_k, \bar{r}_k, \bar{c}_k, \bar{\mathcal{I}}_k\} \tag{4.12}$$

by applying mean-shifting [52], resulting in an updated node set $\bar{\mathcal{N}}$. Mean-shifting iteratively moves each node element to the local mean of the nodes in its vicinity. This reduces the variance of the estimates but preserves the linking of the nodes: $\bar{\mathcal{I}} = \mathcal{I}$. In practice, five iterations are sufficient to reach satisfactory radial trace alignment (Fig. 4.4B). The kernel size used in the mean-shifting process is taken to be the initial radius of each node. In the implementation, prior to mean-shifting, all traces are resampled with a step size of one voxel to get a more fine-grained result.

Algorithm 3 Node grouping.

Require: $\bar{\mathcal{N}}$, r_g ▷ refined node list and grouping radius
$G = [0, \ldots, 0]$ ▷ initialize node group mapping list $|G| = |\bar{\mathcal{N}}| = M$
$\hat{\mathcal{N}} = \{\}$ ▷ initialize group node set $|\hat{\mathcal{N}}| = 0$
for $k = \arg\max_k \bar{c}_k, \ldots, \arg\min_k \bar{c}_k$ **do** ▷ descending correlation
  if $G[k] = 0$ **then** ▷ initialize new group if yet ungrouped
    $m = |\hat{\mathcal{N}}| + 1$ ▷ next node group index
    $G[k] = m$ ▷ fill node group mapping
    $t = 1$ ▷ index group elements
    $(x'_t, y'_t, z'_t, r'_t, c'_t) = (\bar{x}_k, \bar{y}_k, \bar{z}_k, \bar{r}_k, \bar{c}_k)$ ▷ initialize centroid
    $\mathcal{I}'_t = \bar{\mathcal{I}}_k$ ▷ initialize link
    for $l = 1, \ldots, k-1, k+1, \ldots, M$ **do** ▷ all other nodes
      if $(\bar{x}_l - \bar{x}_k)^2 + (\bar{y}_l - \bar{y}_k)^2 + (\bar{z}_l - \bar{z}_k)^2 \leq r_g^2$ **then**
        $t = t + 1$
        $x'_t = \frac{t-1}{t} x'_{t-1} + \frac{1}{t} \bar{x}_l$ ▷ iterative mean
        $y'_t = \frac{t-1}{t} y'_{t-1} + \frac{1}{t} \bar{y}_l$
        $z'_t = \frac{t-1}{t} z'_{t-1} + \frac{1}{t} \bar{z}_l$
        $r'_t = \frac{t-1}{t} r'_{t-1} + \frac{1}{t} \bar{r}_l$
        $c'_t = \frac{t-1}{t} c'_{t-1} + \frac{1}{t} \bar{c}_l$
        $\mathcal{I}'_t = \mathcal{I}'_{t-1} \cup \bar{\mathcal{I}}_l$ ▷ accumulate node linkage
        $G[l] = m$ ▷ fill node group mapping
      end if
    end for
    $\hat{n}_m = (x'_t, y'_t, z'_t, r'_t, c'_t, \mathcal{I}'_t)$ ▷ assign group values
    $\hat{\mathcal{N}} = \hat{\mathcal{N}} \cup \{\hat{n}_m\}$ ▷ add node group
  end if
end for
for $k = 1, \ldots, P$ **do** ▷ $P = |\hat{\mathcal{N}}|$
  $\hat{\mathcal{I}}_k = \text{group}(\hat{\mathcal{I}}_k, G)$ ▷ turn node to group node indices
  $\hat{\mathcal{I}}_k = \text{unique}(\hat{\mathcal{I}}_k)$ ▷ remove repeating indexes
  $\hat{\mathcal{I}}_k = \hat{\mathcal{I}}_k \setminus \{k\}$ ▷ remove self-links
end for

4.3.5 Node grouping

Although the previous step results in refined node estimates, it keeps the total number of nodes and corresponding multiple traces. The next step is to merge overlapping

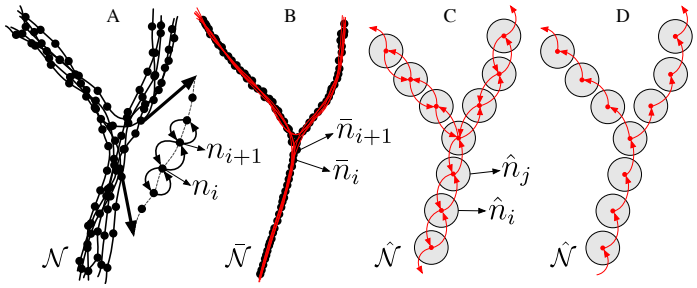


Figure 4.4: Trace merging: (A) accumulated traces, (B) trace refinement, (C) node grouping, (D) tree traversal.

traces and obtain a single trace for each neuron branch. This is accomplished with the node grouping process (Figs. 4.1E and 4.4C) as detailed in Algorithm 3. It iteratively takes from the refined set $\bar{\mathcal{N}}$ an as-yet ungrouped node with the highest cross-correlation value, finds all its neighboring nodes within the predefined Euclidean distance r_g, and groups them by calculating the mean value of each element while accumulating all node links and mapping their indexes to the group node index list. This results in a new set $\hat{\mathcal{N}} = \{\hat{n}_1, \ldots, \hat{n}_P\}$, $P \leq M$, of group nodes:

$$\hat{n}_k = \{\hat{x}_k, \hat{y}_k, \hat{z}_k, \hat{r}_k, \hat{c}_k, \hat{\mathcal{I}}_k\} \tag{4.13}$$

and any two $\hat{n}_i$ and $\hat{n}_j$ are connected if there exists a link between any of the refined nodes captured by these two, as revealed by the accumulated index sets $\hat{\mathcal{I}}_i$ and $\hat{\mathcal{I}}_j$. Thus, all existing inter-node connections $\bar{\mathcal{I}}$ are preserved, and are projected into the inter-group connections $\hat{\mathcal{I}}$.

4.3.6 Tree construction

The final step of the showcased method is the construction of a graph representing the complete neuronal arbor. This is facilitated by the bidirectional connectivity of the group nodes in $\hat{\mathcal{N}}$. However, similar to a real neuron, the final graph must be a tree, in which the nodes are unidirectionally linked (Figs. 4.1F and 4.4D), as also required by the SWC file format for storing digital neuron reconstructions [44, 256]. Starting from the soma node, or from the group node with the highest cross-correlation value if no soma was found in the image, the nodes in $\hat{\mathcal{N}}$ are iteratively traversed using a breadth-first search (BFS) algorithm. In this process it is possible to discard any isolated branches and single-node terminal branches (false positives).

4.3.7 Implementation details

The method showcased in this chapter, named Probabilistic Neuron Reconstructor (PNR)[1], was implemented in C++ as a plugin for the freely available and extendable

[1]The source code of the method is freely available for non-commercial use from `https://bitbucket.org/miroslavradojevic/pnr`

Parameter	Value	Description
r_s	6 [voxels]	Erosion radius
σ	$\{2, 4, 6\}$ [voxels]	Scale combinations
τ	10 [8-bit scale]	Local maxima tolerance
N	20	Number of samples
κ	3 [voxels]	Circular variance
d	3 [voxels]	Tracing step size
ζ	1 [voxels]	Scale variance
K	20	Likelihood sensitivity
$c_{\min}$	0.5	Correlation threshold
L	200	Iteration limit
n	1 [voxels]	Density volume
δ_n	4 [count/voxel]	Node density limit
r_g	2 [voxels]	Grouping radius

Table 4.1: Parameters of the method and default values. The ordering is according to first mention in the main text.

bioimage visualization and analysis tool Vaa3D [189, 196].[2] As mentioned in the preceding sections, the method has a number of free parameters, which are summarized in Table 4.1, where the default values are also listed.

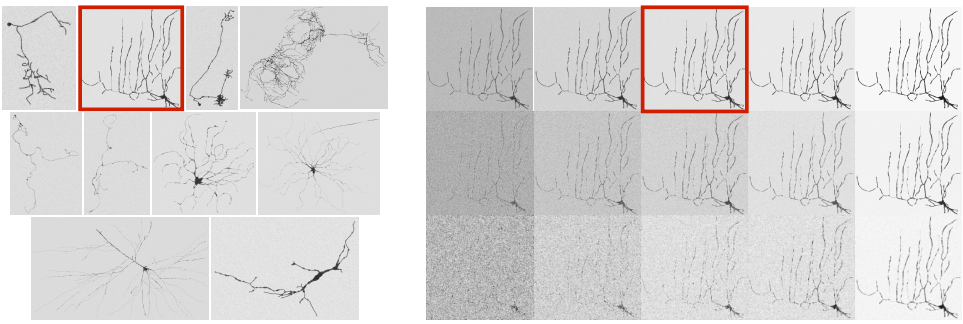
4.4 Experimental results

The performance of the PNR method was evaluated using both synthetic and real fluorescence microscopy image stacks of single neurons and was compared to several alternative 3D neuron reconstruction methods that yielded favorable performance in the BigNeuron project [190]. These included the second all-path pruning method (APP2) [291], NeuroGPS-Tree (GPS) [205], BigNeuron's minimum spanning tree (MST) method, along with the alternative probabilistic method based on probability hypothesis density filtering (PHD) [208] presented in separate chapter of this thesis.

To quantify performance, evaluation adopted the commonly used measures of distance and overlap of neuron reconstructions with respect to the ground truth (in the case of synthetic images) or the gold-standard reconstructions obtained by manual annotation (in the case of real images). The distance measures were the average minimal reciprocal spatial distance (SD) between nodes in the reconstructions being compared, the substantial spatial distance (SSD) using only the nodes with a spatial distance larger than a threshold S, and the percentage of these substantially distant nodes (%SSD), all computed after densely resampling each reconstruction to reduce the distance between its adjacent nodes to one voxel (see [196] for details). The overlap measures were precision (P), recall (R), and the F score [201], computed from the numbers of true-positive (TP), false-positive (FP) and false-negative (FN) nodes according to the spatial distance threshold S.

All experiments were performed on a MacBook Pro with 2.2 GHz Intel Core i7

[2] http://vaa3d.org



(a) (b)

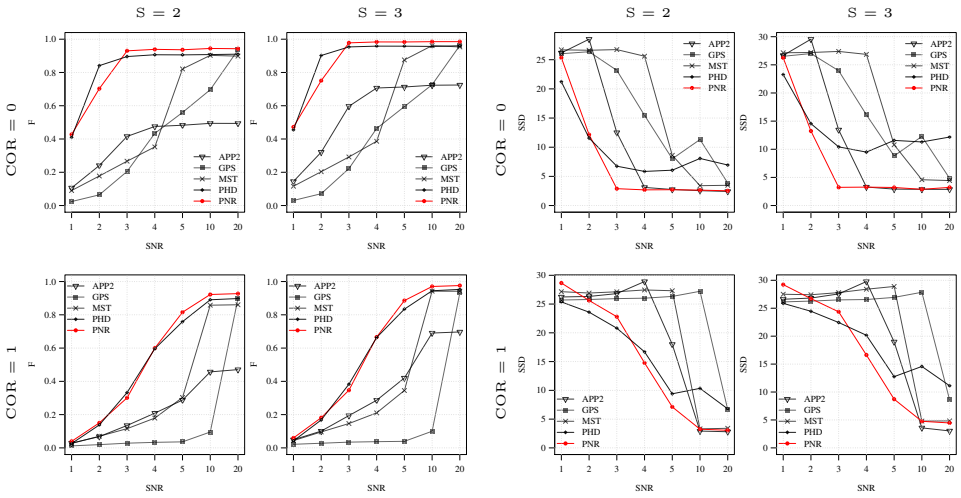
Figure 4.5: Illustration of the synthetic neuron data set used in the presented experiments. (a) Example images of the 10 selected neurons simulated at SNR = 4 and COR = 0.0. (b) Different simulations of the neuron indicated by the red outline in (a) for SNR = 2, 3, 4, 5, and 10 (from left to right) and COR = 0.0, 1.0, and 2.0 (from top to bottom). The marked image in (b) is the same as the marked image in (a). All examples shown here are maximum intensity projections of the 3D synthetic images with inverted intensities for better visualization.

processor and 16 GB RAM memory to test the practicality of the methods on a typical computer system. For each method the score was optimized for each performance measure by exploring a grid of possible parameter values around the default ones (see Table 4.1 for PNR method and the cited publications for the other methods). To keep the experiments feasible, the maximum allowed processing time per stack and method was set to 2 hours. For the conciseness, only the F scores (higher is better) and SSD scores (lower is better) are shown in the sequel, but the conclusions are based on the complete body of results.

4.4.1 Experiments on synthetic neuron images

Prior to evaluating how well introduced method emulates expert manual reconstruction in real neuron images, a controlled experiment was first performed using synthetic neuron images, with known ground-truth reconstructions and predefined levels of signal-to-noise ratio (SNR) and inter-voxel correlation (COR). This allowed us to study the robustness of the method compared to the others as a function of these image quality factors. For this experiment 10 neurons were selected from the BigNeuron training data set [190], representative of the range of morphological complexities in the data set, and for which node radius information (non-default) was available in the corresponding gold-standard reconstructions in SWC format. A dedicated plugin for ImageJ [229] called SWC2IMG was developed for this purpose[3]. The plugin takes any SWC file as input and simulates fluorescence microscopy imaging of all neuronal branches in the file at a specified SNR and COR level, producing an image stack whose true digital reconstruction is the very input. It assumes that in practice, because of

[3]https://github.com/imagescience/SWC2IMG



(a) Examples are shown for COR = 0 (top) and 1 (bottom) in combination with S = 2 (left) and 3 (right).

(b) Examples are shown for COR = 0 (top) and 1 (bottom) in combination with S = 2 (left) and 3 (right).

Figure 4.6: Average score of the methods for the synthetic images as a function of SNR: a) F score and b) SSD score.

the relatively large spatial extent of even a single neuron with its complete arbor, the combination of optical magnification factor and digital image matrix size in real neuron images is typically such that the voxel size is larger than the point spread function (PSF), implying that the partial-volume effect of digitization is more prominent than the optical blurring by the microscope. Based on this, the plugin simulates the imaging simply by estimating for each voxel which fraction of its volume is occupied by the neuron. Next, it simulates noise by using the Poisson noise model representative of optical imaging, which defines SNR as the image intensity inside the neuron above the background, divided by the standard deviation of the noise inside [241]. And finally, to allow for correlated signal and noise, which was found to improve the visual realism of the simulated images, the plugin also offers the possibility to apply Gaussian smoothing at a specified scale, being the COR parameter, while preserving the SNR level. Generally, the lower the SNR and/or the higher the COR level, the more challenging the data and the reconstruction problem.

Using this plugin a synthetic data set was created containing image stacks for a range of SNR and COR values for each neuron (Fig. 4.5). Specifically, SNR = 1, 2, 3, 4, 5, 10, 20, and COR = 0, 0.5, 1, 1.5, 2 were considered. Thus, the generated synthetic data set consisted of 10 (neurons) × 7 (SNR levels) × 5 (COR levels) = 350 image stacks, attempted to reconstruct optimally using the five considered methods (APP2, GPS, MST, PHD, PNR) and a parameter grid-search approach. However, some of the images were very challenging, especially the ones with many branches and low SNR or high COR values, causing the methods to sometimes require excessive computation times or even to get stuck altogether. Because of the mentioned time

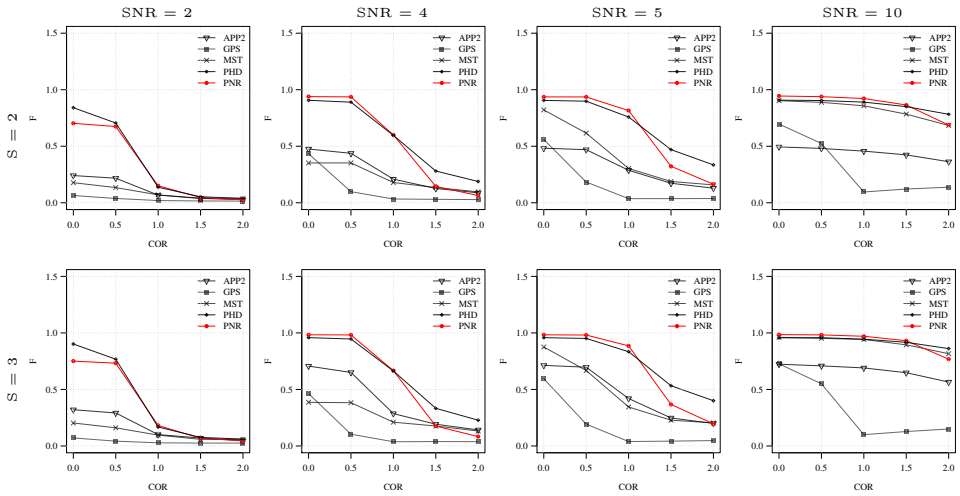


Figure 4.7: Average F score of the methods for the synthetic images as a function of COR. Examples are shown for S = 2 (top) and 3 (bottom) in combination with SNR = 2, 4, 5, 10 (left to right).

constraint, not all methods were able to complete all the reconstructions, and it turned out that only 7 out of the 10 neurons could be reconstructed by all the methods for all SNR and COR values. Therefore the results are presented only for those.

From the average F and SSD scores of the methods as a function of SNR for a few sample values of COR and S (Figs. 4.6a and 4.6b) it is possible to observe that, as expected, increasing the SNR generally improves the performance of all methods (increasing F and decreasing SSD scores). It is also noteworthy that the two probabilistic methods (PHD and PNR) are more robust against noise (especially according to F) and that the method proposed in this chapter (PNR) is often superior overall. The results also show that, as expected, increasing the value of COR (which yields more difficult images) has a strong negative impact on the performance of all methods (lower F and higher SSD scores for the same SNR). This is confirmed when looking more in-depth at the results as a function of COR (Figs. 4.7 and 4.8). Additionally, again as expected in all cases, further observations indicate that increasing the value of S (meaning being more lenient in matching reconstructions to the ground truth) may also strongly affect the scores of all methods (meaning higher F scores, but in this case also higher SSD scores, as the latter includes only node distances larger than S). This is confirmed when looking explicitly at the performance of the methods as a function of S (Figs. 4.9 and 4.10). These results reveal that both the absolute and the relative performance of different methods being compared may depend on S. This is an important observation, since in all studies available in the known existing literature, the somewhat arbitrary value of S = 2 is taken for granted in calculating performance and ranking the methods. Presented results (Figs. 4.9 and 4.10) show that taking other values of S may yield a different ranking. Notwithstanding this

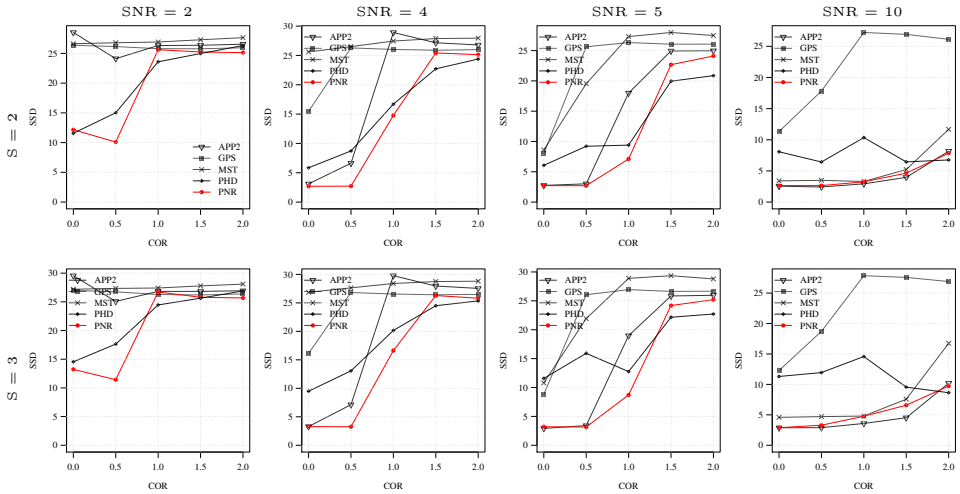


Figure 4.8: Average SSD score of the methods for the synthetic images as a function of COR. Examples are shown for S = 2 (top) and 3 (bottom) in combination with SNR = 2, 4, 5, 10 (left to right).

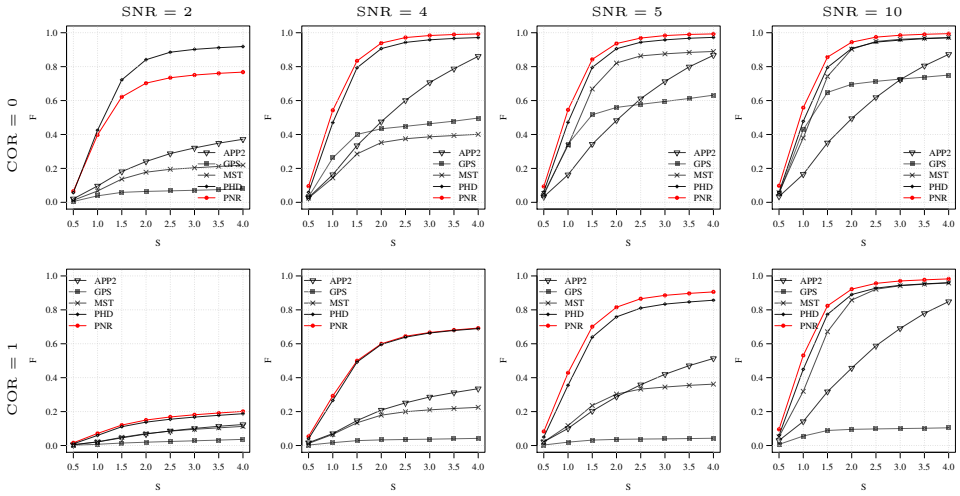


Figure 4.9: Average F score of the methods for the synthetic images as a function of S. Examples are shown for COR = 0 (top) and 1 (bottom) in combination with SNR = 2, 4, 5, 10 (left to right).

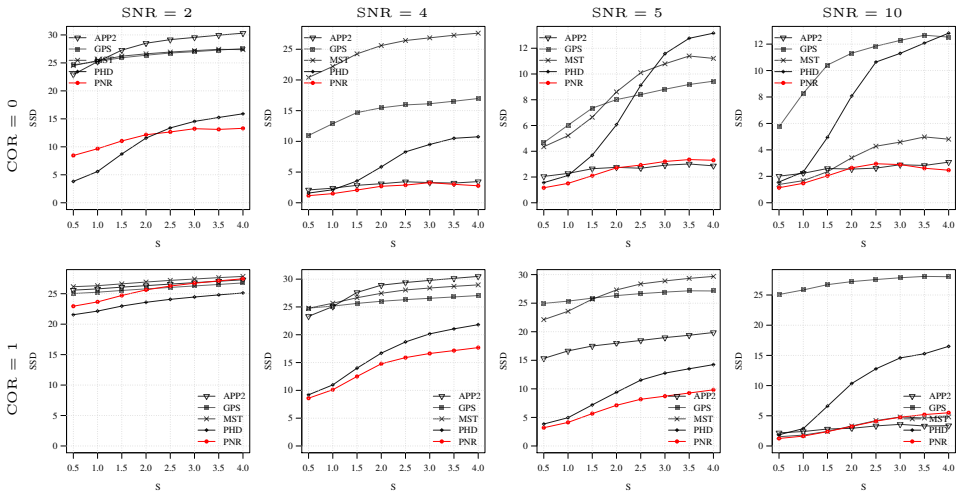


Figure 4.10: Average SSD score of the methods for the synthetic images as a function of S. Examples are shown for COR = 0 (top) and 1 (bottom) in combination with SNR = 2, 4, 5, 10 (left to right).

finding, displayed results also show that under most experimental conditions (SNR, COR, S), the proposed method (PNR) yields superior results. While the alternative, probabilistic neuron tracing method (PHD) [208] is often a strong competitor, the results indicate that the PNR method is more favorable, which is possible to ascribe to its better models for seed point extraction and branch tracing.

Together, the results of the experiments on synthetic neuron images suggest that tracing the image structures repeatedly and in a statistically independently way, indeed yields more evidence about the underlying neuron branches and leads to better reconstructions. This also follows from a visual comparison of the reconstructions (Fig. 4.11). Especially at low SNRs, pruning and fast-marching based methods tend to oversegment the images, while the probabilistic methods still perform relatively well regardless. Even at high SNRs, when most of the methods perform comparably, the proposed method follows the branch structures more closely (see zooms in the last row of Fig. 4.11).

4.4.2 Experiments on real neuron images

In addition to synthetic data, three real neuron image data sets were used to evaluate the absolute and relative performance of the presented method. The first two are the olfactory projection fibers (OPF) data set (9 image stacks) and neocortical layer-1 axons (NCL1A) data set (16 image stacks) from the DIADEM challenge [39], and the third is part of the BigNeuron (BGN) training data set (76 image stacks) [190], all imaged with fluorescence microscopy (confocal or two-photon) and manually annotated as described in detail in the cited works and corresponding resources. Being the smallest of the three, in terms of both neuronal volume and complexity, OPF is

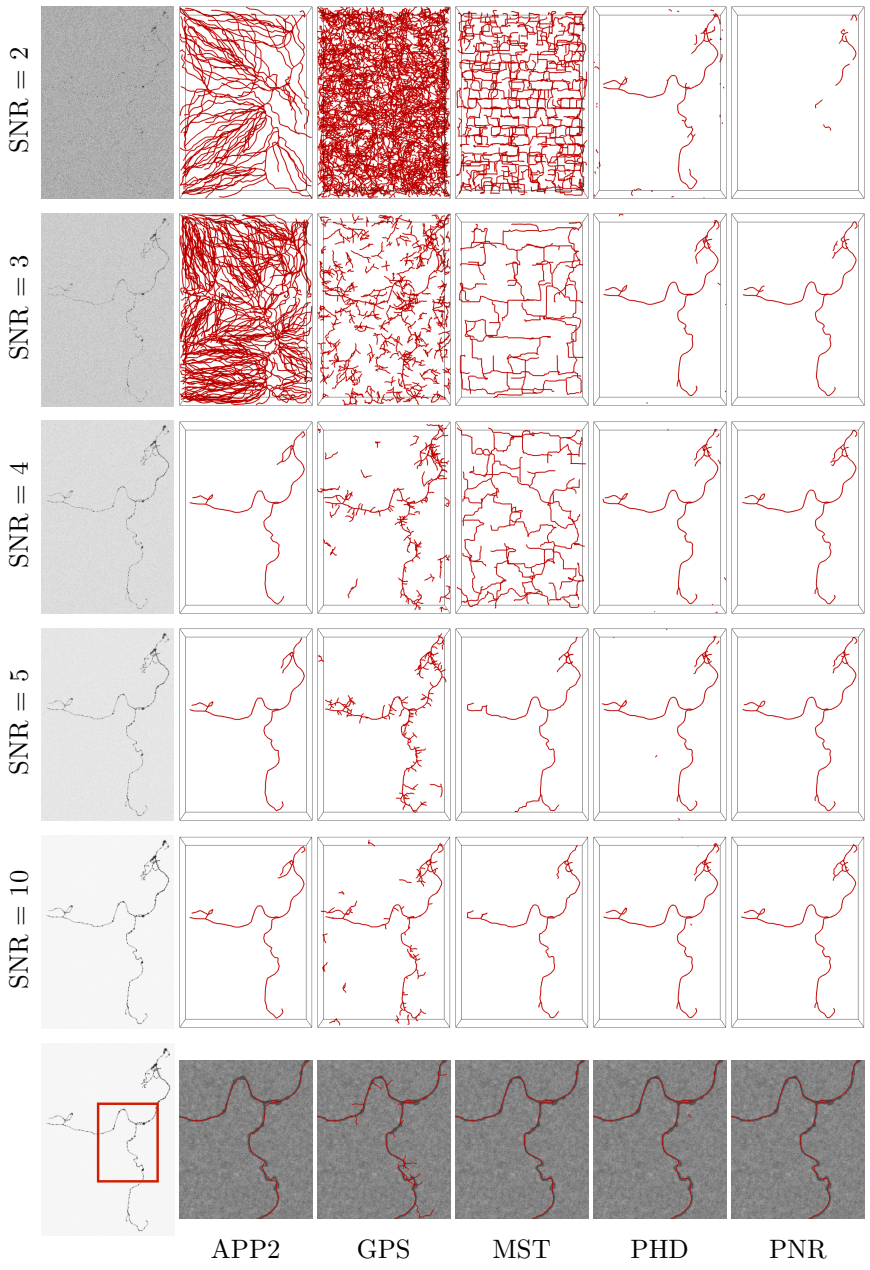


Figure 4.11: Visual comparison of neuron reconstructions produced by the considered methods from synthetic image stacks of a single neuron at different SNR levels. The image stacks (generated used COR = 0) are shown as inverted maximum intensity projections (left column) and the reconstructions of the different methods (remaining columns) are shown in red as surface renderings.

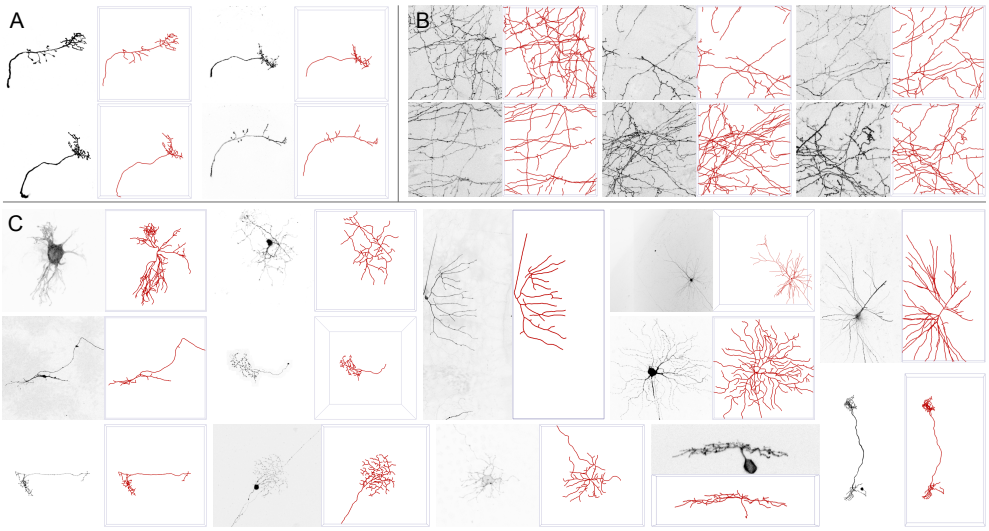


Figure 4.12: Illustration of the real neuron image data sets used in the presented experiments. Examples are shown of (A) the OPF data set (4 of 9 stacks), (B) the NCL1A data set (6 of 16 stacks), and (C) the BGN data set (13 of 76 stacks). Each example shows the maximum intensity projection of the image stack (left panel) but with inverted intensities for better visualization, and the corresponding manual reconstruction (right panel) as a surface rendering (in red), both generated using Vaa3D [196].

probably the most often used data set in the field. NCL1A is often used as it contains neuronal network-like structures and no clear somas. And BGN is the largest, most diverse, and thus most challenging data set for evaluating neuron reconstruction methods. Together, the 100+ image stacks in these data sets have a wide variety of image qualities and volumes (10 MB to 2 GB per stack) and portray a wide range of neuronal shapes and complexities (Fig. 4.12), representative of many studies. For some stacks in the BGN data set, the voxel size was unknown, and in these cases the default x:y:z voxel aspect ratio of 1:1:2 was used, reflecting the typically lower resolution in the depth dimension. Also, because of the mentioned processing time constraint, 3 of the 76 image stacks could not be reconstructed by all methods, so the presented results are based on the remaining 73. The results of the experiments on these three real data sets (Figs. 4.13-4.15) indicate that, as in the experiments on synthetic data, the probabilistic methods PHD and PNR typically show superior performance in terms of both F and SSD score. Of these two methods, the PNR method proposed in this chapter consistently shows the smallest performance spread, indicating it is more robust than the alternative PHD method. For the BGN data set, being the most diverse of the three, the performance spread (including outliers) of all methods is the largest, and the increase in performance as a function of S is the smallest, as expected. Nevertheless, the PNR method consistently shows the best overall performance especially for this data set. In other words, for any given data

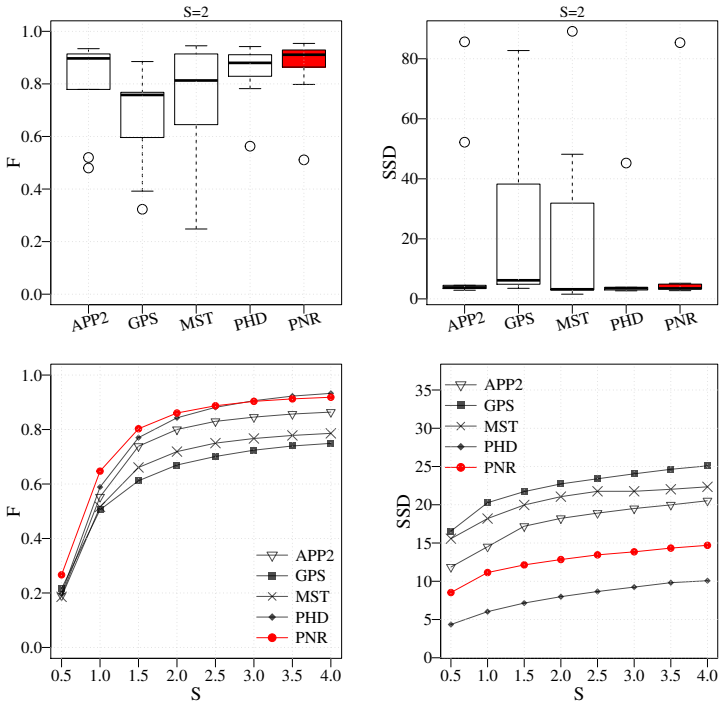


Figure 4.13: Performance comparison for the OPF data set. Results are shown for the F measure (left column) and SSD measure (right column) and in the form of distributions for S = 2 (standard R box plots in top row) and averages as a function of S (bottom row).

set similar to those considered in this study, PNR is the favorable method a priori. Obviously this does not necessarily mean that PNR will give the best reconstruction for each and every image stack in the data set, but simply that the chances are higher. This is confirmed when looking at a few example image stacks from the three data sets and the corresponding best reconstructions produced by the different methods by maximizing the F score in the parameter grid search (Figs. 4.16-4.18). As these examples show, although PNR often outperforms the other methods, in specific cases one of the other methods may give better reconstructions. But altogether the experimentation results suggest the conclusion that the PNR method showcased throughout this chapter is a valuable addition to the neuron reconstruction toolbox.

4.5 Conclusions

A new fully automated probabilistic neuron reconstruction method (PNR) based on sequential Monte Carlo filtering is presented in this chapter. It traces individual neuron branches from automatically detected seed points repeatedly but statistically

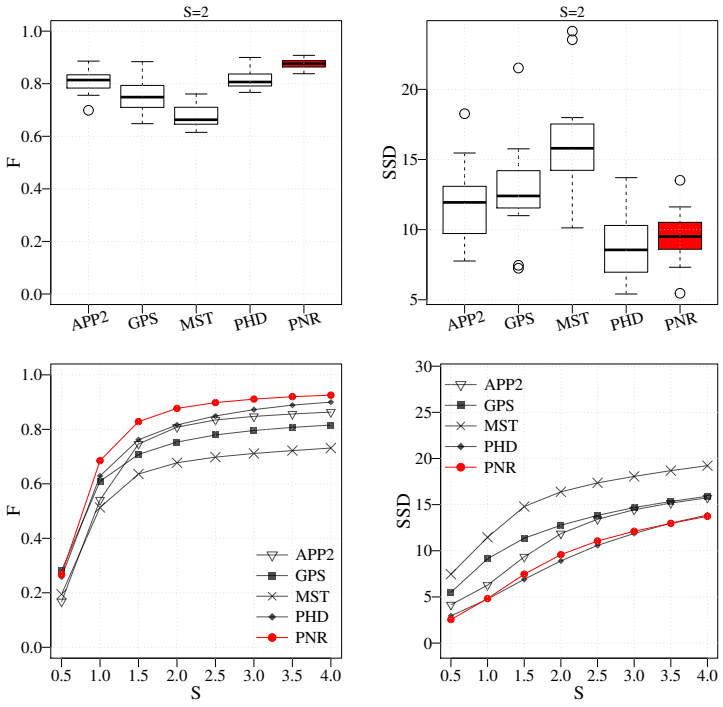


Figure 4.14: Performance comparison for the NCL1A data set. Results are shown for the F measure (left column) and SSD measure (right column) and in the form of distributions for S = 2 (standard R box plots in top row) and averages as a function of S (bottom row).

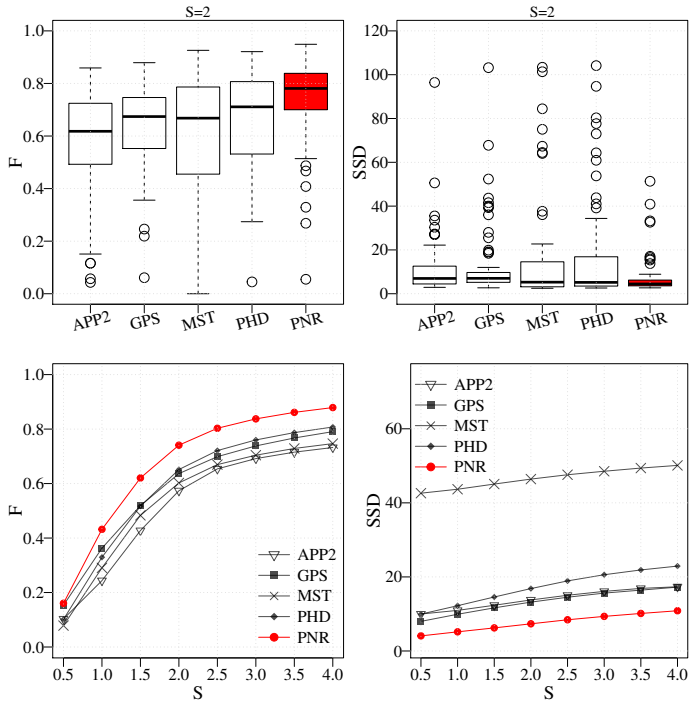


Figure 4.15: Performance comparison for the BGN data set. Results are shown for the F measure (left column) and SSD measure (right column) and in the form of distributions for S = 2 (standard R box plots in top row) and averages as a function of S (bottom row).

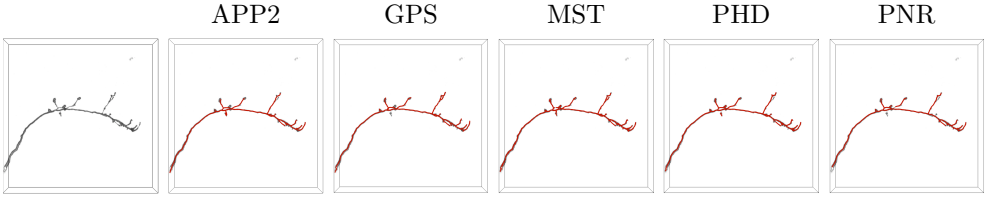


Figure 4.16: Example neuron reconstructions of an image stack from the OPF data set. Shown are the original arbor (volume rendering on the left) and the reconstructions (overlaid surface renderings in red) of the different methods (indicated at the top) corresponding to the best F score (given below each reconstruction) for S = 2 with respect to the available manual reconstruction.

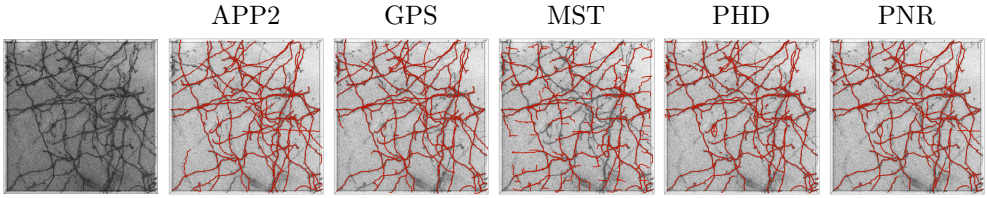


Figure 4.17: Example neuron reconstructions of an image stack from the NCL1A data set. Shown are the original arbor (volume rendering on the left) and the reconstructions (overlaid surface renderings in red) of the different methods (indicated at the top) corresponding to the best F score (given below each reconstruction) for S = 2 with respect to the available manual reconstruction.

independently to acquire more evidence and to be more robust to noise and other artifacts. The traces are subsequently refined, merged, and put into a tree representation for further analysis. The method was evaluated on both synthetic and real neuron images and compared against various other state-of-the-art neuron reconstruction methods (APP2, GPS, MST, PHD) using commonly used quantitative performance measures (earlier presented F and SSD scores). To obtain realistic synthetic data, a novel simulator (SWC2IMG) was developed that can turn any given SWC file into an image stack of specified quality whose ground truth reconstruction is the input. The evaluation on real data used about 100 single-neuron fluorescence microscopy image stacks of widely varying quality and complexity, with corresponding manual reconstructions serving as the gold standard, from three different data sets used in the DIADEM and BigNeuron studies. The results show conclusively that the proposed method is generally favorable and also outperforms the alternative probabilistic neuron reconstruction method based on probability hypothesis density (PHD) filtering, presented in a dedicated chapter of the thesis. Nevertheless, there still remains much room for further improvement, as none of the quantitative scores were near perfect for any of the considered methods even for high SNR levels and very lenient distance thresholds. Possible directions for future work within the presented probabilistic framework would be to explore other state transition and measurement models. Alternatively, since no single method always performs best on all images of a given data set, and the results of different methods are likely complementary, another possible direction could be to combine multiple methods either during tracing or in a post-processing step. The latter approach is already being explored in the BigNeuron project. But regardless of the outcome of this effort it is possible to draw a conclusion that the method proposed throughout this chapter may already prove to be of great use in many cases.

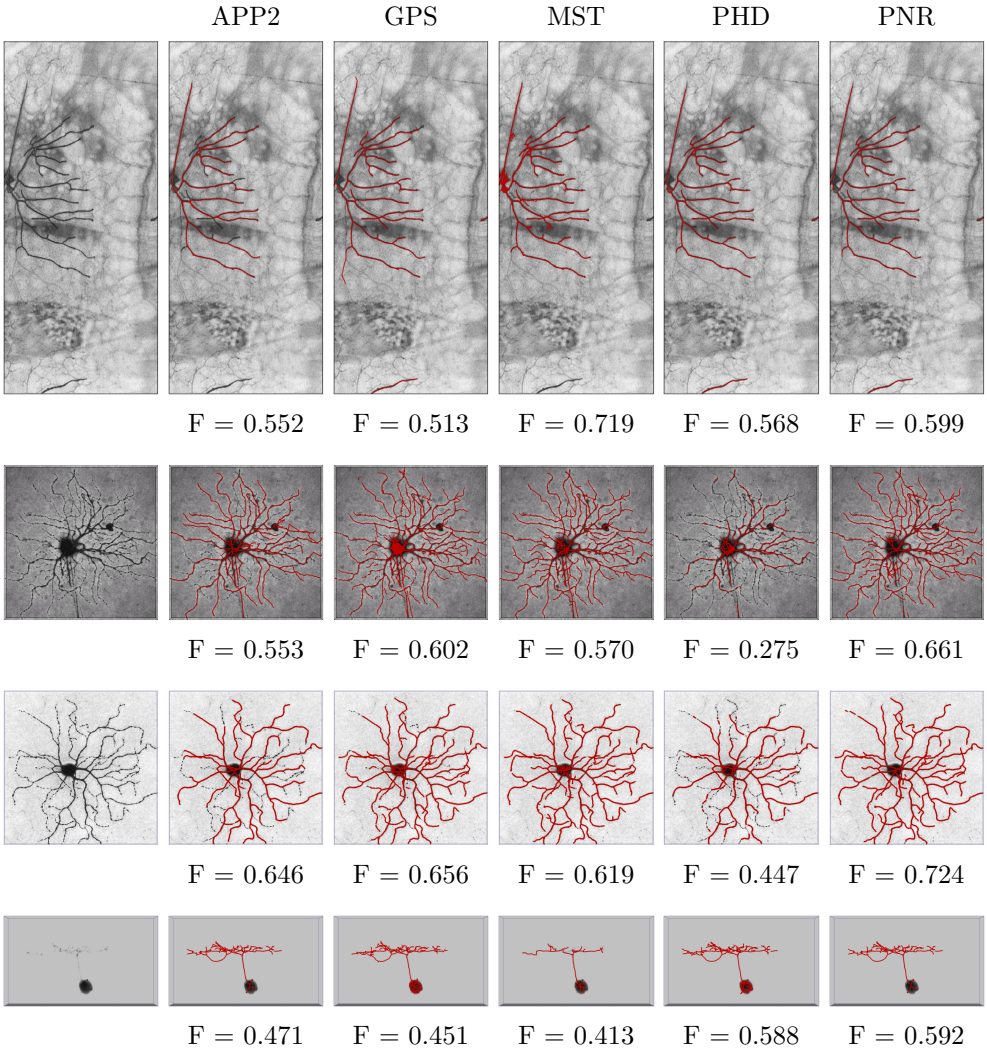


Figure 4.18: Example neuron reconstructions of four image stacks from the BGN data set. Shown are the original arbors (volume renderings on the left) and the reconstructions (overlaid surface renderings in red) of the different methods (indicated at the top) corresponding to the best F score (given below each reconstruction) for S = 2 with respect to the available manual reconstruction.

Chapter Five

Automated neuron detection in high-content fluorescence microscopy images using machine learning

The study of neuronal morphology in relation to function, and the development of effective medicines to positively impact this relationship in patients suffering from neurodegenerative diseases, increasingly involves image-based high-content screening and analysis. The first critical step toward fully automated high-content image analyses in such studies is to detect all neuronal cells and distinguish them from possible non-neuronal cells or artifacts in the images. In this chapter, the performance of well-established machine learning techniques is investigated for this purpose. These include support vector machines, random forests, k-nearest neighbors, and generalized linear model classifiers, operating on an extensive set of image features extracted using the compound hierarchy of algorithms representing morphology, and the scale-invariant feature transform. The experiments performed on a dataset of rat hippocampal neurons are presented in order to find the most suitable classifier(s) and subset(s) of features in the common practical setting where there is very limited annotated data for training. The results indicate that a random forests classifier using the right feature subset ranks best for the considered task, although its performance is not statistically significantly better than some support vector machine based classification models.

Based upon: G. Mata, M. Radojević, C. Fernandez-Lozano, I. Smal, N. Werij, M. Morales, E. Meijering, J. Rubio, "Automated neuron detection in high-content fluorescence microscopy images using machine learning", *Neuroinformatics*, *in press*, 2018.

5.1 Introduction

Neurons are special cells in the sense that they codify and transmit information in the form of action potentials. Networks consisting of many billions of neurons, such as in the brains of higher organisms, are extraordinarily complex and perform many different functions. Since the pioneering work of [214] it is well known that the morphology of neurons vary widely in different parts of the brain and that neuronal morphology and function are intricately linked. Moreover, in healthy conditions, neuronal (sub)networks within the brain are dynamic and continuously readjust their connections during the lifetime of an organism in response to external stimuli, in order to refine existing functions or learn new ones [13]. Conversely, in pathological conditions, disease processes destructively alter neuronal morphology and cause progressive loss of function, such as in Alzheimer's and Parkinson's disease, but also in aging [272]. Thus the study of neuronal cell morphology in relation to function, in health and disease, is of high importance for developing suitable drugs and therapies [169].

A convenient tool to visualize large numbers of cultured cells for phenotypic profiling and analysis in drug discovery is high-content fluorescence microscopy imaging [8, 35, 245, 290]. By automated acquisition it produces very large amounts of image data, which cannot be analyzed manually but require automated high-content analysis (HCA) in order to take full advantage of all captured information. HCA is also used increasingly in neuroscience research [6, 80, 128] and various image processing pipelines have been developed for quantitative analysis of neuronal cells in high-content images [47, 70, 207, 249, 271, 289, 306]. However, especially in screening applications, where the image quality is often relatively low and may vary widely between experiments, the challenge remains to develop more accurate and more robust image analysis methods [139, 170, 251].

The first critical step in any HCA pipeline is the detection of the objects of interest in the images. It is well recognized now in many areas of microscopic image analysis that machine learning based classification methods are an excellent choice for this task and typically outperform non-learning methods based on manually defined rules [10, 125, 139, 251]. However, which classifiers work best, and on which sets of image features, may depend on the specific image data and detection task, and needs to be determined experimentally before using HCA on a routine basis in a given application.

This chapter comprises the investigation of the performance of machine learning methods for the specific task of detecting neuronal cells in high-content fluorescence microscopy images as a first step toward fully automated HCA in conduced neuroscientific studies. An early version of this work was presented at a conference [168] while the chapter reports on a significant extension of that work including more classifiers, more extensive experiments and results, and a much deeper and more solid (statistical) analysis and discussion of the findings. The classifiers are explored based on precalculated image features in order to determine which combinations of classifiers and features work best in a practical setting where there is very limited annotated data for training. Specifically, various state-of-the-art classifiers are considered based on support vector machines (SVM), random forests (RF), k-nearest neighbors (KNN), and generalized linear models (in particular GLMNET), operating on more than a thousand image features extracted using the compound hierarchy of algorithms rep-

resenting morphology (CHARM) and the scale-invariant feature transform (SIFT).

5.2 Materials and methods

The published image data were used and publicly available software tools were employed to facilitate the reproducibility of presented study. This section successively describes the image dataset, the used methods for extracting image features, and the considered machine learning methods[1].

5.2.1 Image dataset

The high-content image data used in this study originates from the ongoing research aimed at discovering effective treatments for neurological disorders [62, 82, 83]. The acquisition of the images is described, their annotation, and the strategy used to obtain a well-balanced dataset for training of the machine learning algorithms.

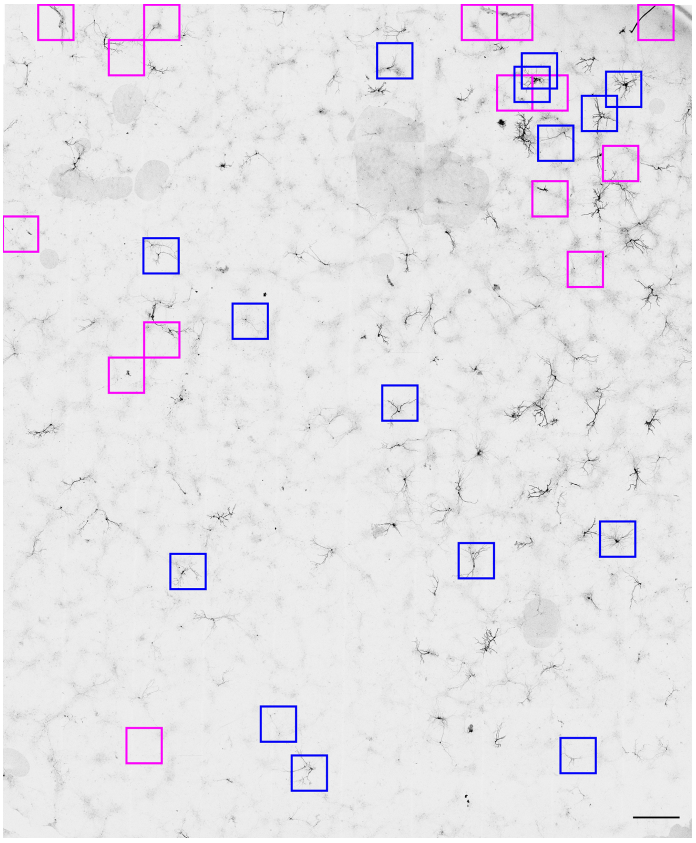
5.2.1.1 Image acquisition

Rat hippocampal neurons were cultured and transfected with green fluorescent protein (GFP) and imaged with a Leica SP5 automated confocal fluorescence microscope using its Matrix modules and a 20× lens. The imaged neurons, coming from a part of the brain (the hippocampus) that is well known to be involved in higher functions such as learning and memory [253], typically have a pyramidal soma with a complex dendritric tree [109], and their in-vivo morphological features are well conserved in culture conditions. Eight two-dimensional (2D) high-content images were acquired (total size >1 GB), each with a size of about $10{,}000 \times 12{,}000$ pixels, covering approximately $70\,\mathrm{mm}^2$ of culture dish. Each image is a mosaic made up of tiles of size 1024×1024 pixels, automatically acquired and stitched using the Leica Matrix module. Prior to imaging, the user has to select the desired culture area within the field of view, and the module calculates the tiles to be imaged in order to cover the chosen area, considering 10% overlap between neighboring tiles. Each mosaic contains on the order of 40 transfected neurons (Fig. 5.1). The used specimens usually have about 100 neurons, but more than half of them are not or only partly imaged, as they are in different optical planes or close to the borders of the dish, making the automated detection of relevant image structures (complete neurons) as opposed to irrelevant image structures (incomplete neurons, astrocytes, and artifacts) quite challenging.

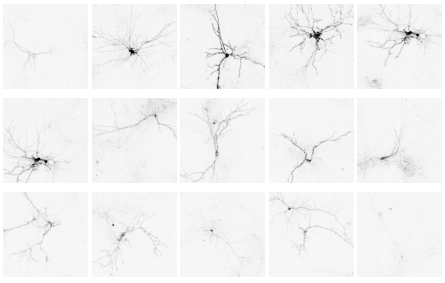
5.2.1.2 Image annotation

To obtain a reference dataset for training and testing of the machine learning methods, an expert neurobiologist manually marked all the regions of interest (ROIs) containing neurons in these images, about 400 in total. It was established upon data inspection that relevant neurons typically cover an area of around 500 × 500 pixels in the used

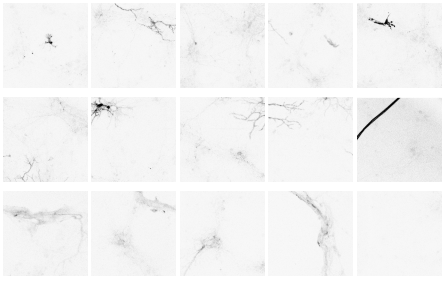
[1] Materials and methods are available from `www.unirioja.es/cu/jurubio/ANDHCFMIUML`



(a) Example high-content image. Scale bar: 500μm.



(b) Example patches considered as positives (blue squares).



(c) Example patches considered as negatives (magenta squares).

Figure 5.1: Part of a high-content fluorescence microscopy image (a) where the blue squares highlight some example patches containing neuronal structure and the magenta squares depict some example patches containing background. These squares are enlarged in (b) and (c) for a better visualization. The intensities of the shown images are inverted compared to their originals for displaying purposes.

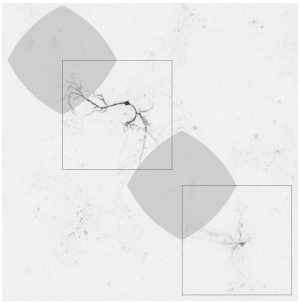


Figure 5.2: Two example neurons with their expert-marked ROIs (black squares) and their potential alternative positive patch locations (gray regions). The latter comprise all possible top-left corner positions of patches with the same size as the given ROI and having 50% or more area overlap with that ROI.

images and therefore the ROI size was fixed to these dimensions. Using the same window size, additional patches were automatically sampled from the remaining parts of the images, containing all different types of irrelevant image structures. More specifically, to ensure evenly distributed sampling of background patches across the images, a regular grid is defined, including every patch from the grid having less than 50% overlap with any of the neuron ROIs marked by the expert, resulting in approximately 4,500 non-neuron patches. In the sequel, neuron ROIs are referred to as 'positives' and the non-neuron image patches as 'negatives' (Fig. 5.1).

5.2.1.3 Dataset balancing

Due to the neuron sparseness within the image data, the patches of the negative class far outnumbered those of the positive class, with a ratio of approximately 10:1, resulting in an imbalanced dataset. It is well known that the performance of classification algorithms may be negatively impacted by the data being imbalanced [36, 49, 65, 91], as the algorithms may overfit the majority class and underfit the minority class, and favor the former, yielding biased results [98,148]. Approaches to deal with class imbalance can roughly be divided into two categories [114,118,140]: data-level approaches, which modify the collection of data samples to balance the class distributions, and algorithm-level approaches, which modify the learning algorithms to alleviate their bias, for example by introducing costs to balance the importance of the different classes. Since the class imbalance was substantial in showcased study, and existing algorithms were mostly used and aimed to evaluate their performance without tweaking them for this particular application, the decision to oversample the minority class was taken in order to obtain approximately the same number of samples in each class. To this end, the popular synthetic minority oversampling technique (SMOTE) is employed [48] of which several variants exist [108,140,219]. Specifically, for each neuron ROI marked by the expert, all patches having at least 50% overlap with that ROI (Fig. 5.2) are also considered as potential positive samples. However, the higher the overlap percentage of a patch, the higher the relevance of that patch, as it contains more neuron structure. Therefore, a weight is assigned to each potential patch corresponding to the overlap percentage. Taking this into account, random sampling was performed from the pool of all potential patches in order to avoid bias (Fig. 5.3). This resulted in a positive class and a negative class each consisting of approximately 4,500 samples in total.

5.2.2 Images features

To train the machine learning algorithms, a large number of predefined features extracted from the positive and negative image patches is used. In this study two very comprehensive feature extraction approaches were employed: the compound hierarchy of algorithms representing morphology (CHARM) and the scale-invariant feature transform (SIFT). Each of them is briefly described in this section. In the training stage of the machine learning algorithms, feature values were normalized to zero mean and unit variance per feature over the whole data set, and constant features were pruned.

5.2.2.1 CHARM features

For the extraction of the CHARM features, the open-source software library WND-CHARM [186, 237] is used, which has been successful for many pattern recognition applications in biology [236, 270] as well as in astronomy [141, 235] and in art [238]. It can extract a large number of generic image descriptors and also includes a classifier based on the weighted neighbor distance (WND) between feature vectors. However, since the performance of this classifier was rather limited in the initial results of this study [168], alternative machine learning algorithms were explored for specified classification task, but using the image features calculated using this software library. In total 1,059 CHARM features are calculated for each positive and negative patch (recent versions of WND-CHARM can extract even more features but at an increased computational cost).

The calculated image features can be divided into four categories: polynomial decompositions, high-contrast features, pixel statistics, and texture descriptors. The first category includes features based on the Zernike polynomials and Chebyshev polynomials [110] as well as Chebyshev-Fourier statistics. Features from the second category include various statistics calculated from the Prewitt edges [202], Gabor wavelets [95], and object masks obtained by Otsu thresholding [187]. The third category consists of image features calculated from the multiscale intensity histogram [113] and various statistics based on the image moments. The last category includes the Haralick [117] and Tamura [263] texture features. In addition, the software calculates various image transforms, including the Radon, Fourier, wavelet, Chebyshev, and edge transforms, as well as transforms of image transforms. For more detailed technical descriptions of all features and transforms, reader is referred to [186].

5.2.2.2 SIFT features

The SIFT algorithm [156] is another popular method which extracts meaningful features from images for pattern recognition tasks. It has been used for a very wide range of applications in thousands of studies, including in biomedical image analysis [129, 176, 183, 183, 298, 304]. The extraction of SIFT features from a patch consists of four main steps. First, a Gaussian scale space is calculated, and potentially interesting points are identified by searching over all scales and locations for extrema in the difference-of-Gaussian function. Next, key points are selected from this list of candidates based on their measures of stability, and their precise location and scale

are determined by model fitting. Then, based on the local gradient directions, each key point is assigned to one or more orientations (binned angles). And lastly, orientation histograms are constructed from the local gradients in a region around each key point, relative to the key point's assigned orientation, and the histogram entries constitute the elements of a (typically 128-dimensional) feature vector. Normalizing the feature vector results with a key point descriptor that is relatively invariant to spatial distortions and changes in illumination. All key point descriptors of a patch taken together form the SIFT features of that patch.

A problem in comparing image patches based on their SIFT features is that the number of key points, and thus the number of descriptors, may be different for each patch. The comparison is facilitated by applying a transform that represents each patch by a feature vector of fixed length [296]. A very effective and popular approach to achieve this is to use the bag-of-words (BoW) model [85]. Here, all descriptors of all available patches are divided into a fixed number of clusters by k-means clustering [161], and the mean of each cluster represents a visual 'word', a vector of the same dimensionality as the descriptors. Subsequently, for any given patch, each of its descriptors is assigned to the single cluster to which it is closest according to the Mahalanobis distance. Such mapping yields a histogram vector of fixed length k, with each vector element being the number of patch descriptors assigned to the corresponding cluster.

To obtain the SIFT-BoW feature vector for each positive and negative patch, the VLFeat software library [278] is used in conjunction with MATLAB (The MathWorks Inc.). The vector length is a user parameter, and the classification performance of the different machine learning algorithms is evaluated for lengths of 20, 40, 60, 80, 100, 150, 200, and 230.

5.2.3 Machine learning

Four different machine learning algorithms were considered for the classification task in this study. This section summarizes the algorithms and their hyperparameters, explains the resampling strategies used in the training and testing of the algorithms, and the feature selection approach.

5.2.3.1 Classification algorithms

Support Vector Machines (SVM) are one of the best known and most successful machine learning algorithms for both classification and regression problems [32,34,275, 276]. In classification problems, the principal aim of SVM is to find the hyperplane in the feature space that best separates the given samples (in this case neuron and non-neuron patches), by maximizing the distance between the samples and the hyperplane [40]. If the problem requires more complex (nonlinear) separation functions, SVM can still be used, by employing so-called kernel functions that transform the high-dimensional feature space such that a hyperplane (linear) can still be used as the separation function. Generally speaking, one could interpret a kernel as a similarity measure [279]. Different types of kernels have been proposed, the Gaussian radial basis function (RBF) being one of the most popular [61]. Two hyperparameters need

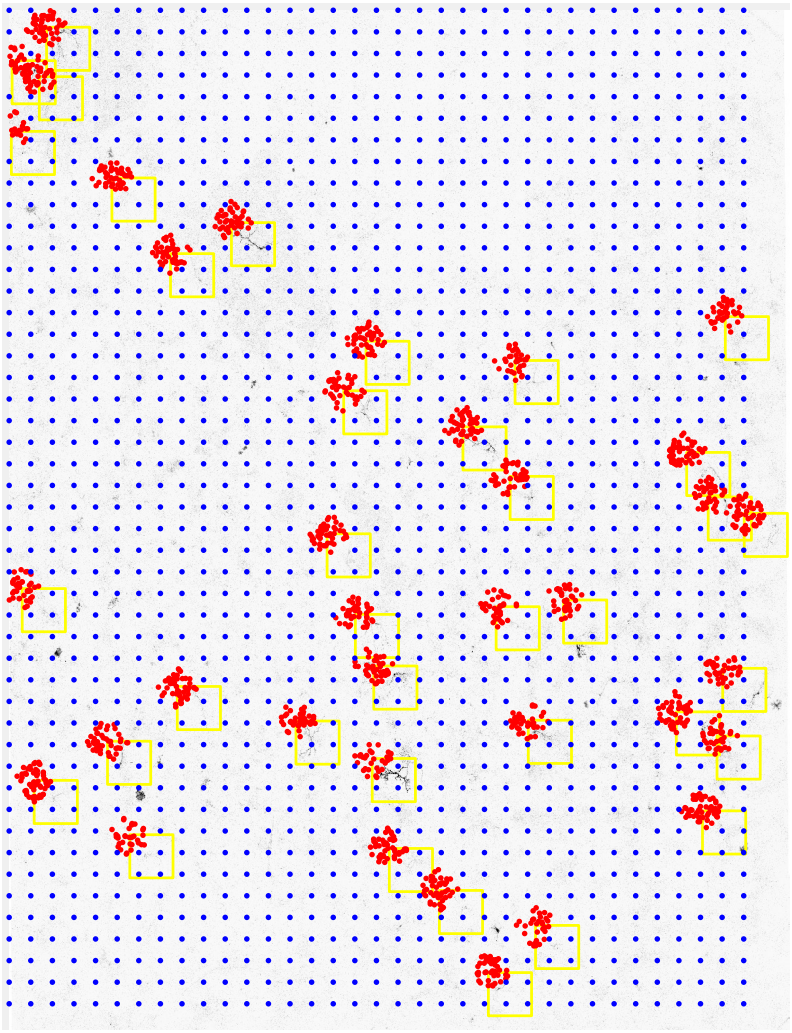


Figure 5.3: Example of positive patch oversampling. The background shows a high-content fluorescence microscopy image (with intensities inverted), and the graphical overlay shows the neuron ROIs marked by the expert (yellow squares), the top-left corners of the patches randomly sampled from all possible patches considered as alternative positives (red dots), and the intersection points (blue dots) of the regular grid used for negative patch sampling (Section 5.2.1.2).

to be optimized for best performance, one related to the SVM algorithm itself, the other related to the Gaussian RBF kernel. The first ('cost') is the trade-off between the misclassification of the samples and the simplicity of the decision surface. The second ('gamma') is the free parameter of the Gaussian function. The grid search used in the experiments of this study considers values 2^k for integer $k = -12, \ldots, 12$

for both parameters.

Random Forest (RF) is another prominent machine learning algorithm for classification and regression [38]. As a classifier, it operates by randomly taking multiple bootstrapped subsets of the data, fitting a decision tree to each one of them, and outputting the mode of the class outputs of the individual trees. This approach reduces the possibility of overfitting the training dataset and generally produces more accurate results than a single decision tree. The RF has two main hyperparameters. The first ('node size') is the minimum size of the terminal nodes of the decision trees. Throughout the experiments, integer values of $1 \dots 5$ were considered for this parameter. The second ('mtry') is the number of features randomly sampled as possible candidates at each split. For this parameter, integer values of $5 \dots 36$ were considered.

k-Nearest Neighbor (KNN) classification operates by comparing an unclassified patch to patches with known class labels (the reference set), then selecting the k most similar of these patches (the nearest neighbors) according to some distance metric in the feature space, and outputting the most frequently occurring class label of these patches [59]. In this study, a weighted KNN algorithm is used [121,221] which employs the Minkowski distance and classifies patches using the maximum of summed kernel densities. This algorithm uses kernel functions to weigh the neighbors according to their distances. The KNN algorithm requires optimization of only one hyperparameter ('k'), for which integer values of $3 \dots 9$ are considered.

Generalized Linear Model (GLMNET) via penalized maximum likelihood [93] is a regularized statistical model whose response variable is a Bernoulli indicator used for classification. It is based on the least absolute shrinkage and selection operator (LASSO) [264]. Similar to the LASSO, this method simultaneously performs automatic feature selection and continuous shrinkage (regularization), and is able to select groups of correlated features. Specifically, GLMNET combines l_1 and l_2 penalties for regularization, and has two hyperparameters. The first ('alpha') is in the range $[0, 1]$ and linearly weighs the contributions of the different types of penalities, with value 0 corresponding to l_2 regularization, and 1 to l_1 regularization. Throughout the experiments, values 0, 0.15, 0.25, 0.35, 0.5, 0.65, 0.75, 0.85, and 1 were used. The second parameter ('lambda') determines the degree of regularization, for which values of 0.0001, 0.001, 0.01, 0.1, and 1 were considered.

The statistical computing software tool R [206] was used in the conducted experiments along with the R packages *mlr* [30], *e1071* [173], *randomForest* [150], *kknn* [227], and *GLMnet* [93], to evaluate all the machine learning algorithms. Most of the result plots presented in this chapter were generated using the R package *ggplot2* [287].

5.2.3.2 Resampling strategies

The mentioned hyperparameters of the machine learning algorithms need to be optimized for best performance. To accomplish this, and at the same time make an honest comparison of the algorithms under equal conditions, a nested resampling approach [31,243] is used, involving an inner loop and an outer loop. In this approach, the actual performance assessment of the algorithms takes place in the outer loop, implemented as three independent runs of a 10-fold cross-validation experiment, with

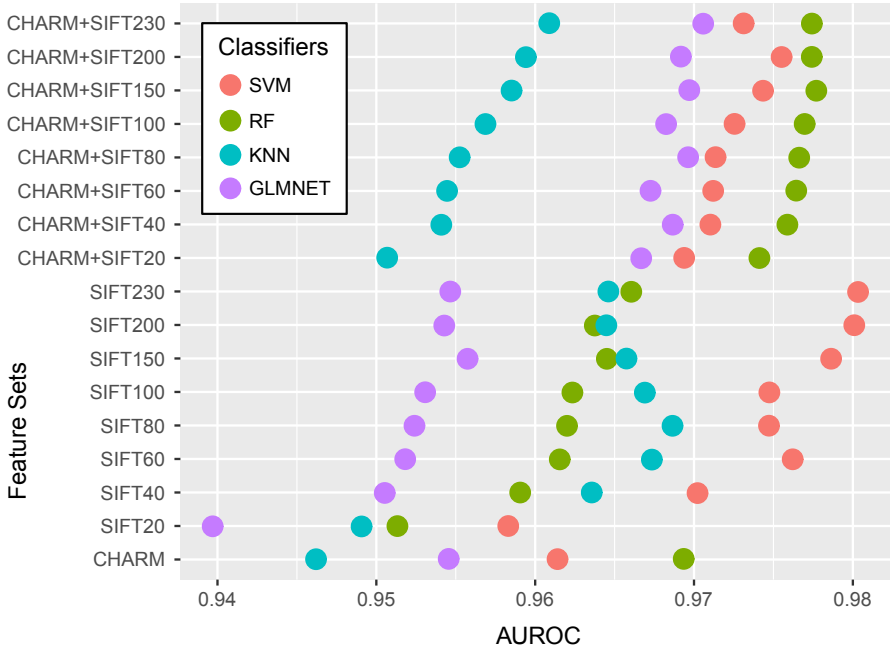


Figure 5.4: Results of the initial exploratory experiment. Each of the considered classifiers (SVM, RF, KNN, GLMNET) was evaluated for each of the described 17 feature sets according to the performance measure (AUROC) using the described simplified resampling strategy.

stratification (to ensure having the same proportion of positive and negative samples in all partitions of the cross-validation), where the final performance scores are obtained by aggregation. In each iteration of the outer loop, the corresponding training set is used in an inner loop, to find the optimal values of the hyperparameters of the algorithms. The inner loop was implemented using a holdout approach, where the given training set from the outer loop is redivided into a training subset (2/3rd of the set) and a validation subset (1/3rd of the set), and a grid search is run on the hyperparameters. The hyperparameter values that give the best performance are subsequently used to retrain the algorithms on the given training set from the outer loop. This nested resampling strategy is statistically sound but computationally expensive. To make the experiments computationally feasible, the search space is discretized using the hyperparameter values listed in the previous section.

5.2.3.3 Feature selection

Although a priori it is appropriate to consider as many features as possible, and increasing computational power allows us to construct larger and larger feature sets, in the end many features may be irrelevant or may even negatively impact the performance of the machine learning algorithms. Thus the experimentation also aimed

to investigate which of all considered features positively contribute most to the performance of the algorithms in this application. Knowledge of the best features allows one to build potentially better and computationally more efficient classifiers. Moreover, it may shed light on which image information is most relevant to the classification task, which in turn may provide useful hints to improve the imaging process. There exist various approaches for feature selection using machine learning algorithms in supervised classification problems, including filter, wrapper, and embedded approaches [218]. This study uses the filter approach, as it is independent of the classifier, fast, scalable, and needs to be applied only once, after which the different algorithms can be evaluated.

5.3 Experimental results

All experiments in this study were carried out using the BioCAI HPC cluster facility at the University of A Coruña. The area under the receiver operating curve (AUROC) measure is used to quantitatively assess and compare the performances of the machine learning algorithms as it captures both Type I and Type II errors [84]. First, an initial exploratory experiment was performed on various combinations of CHARM and SIFT feature sets to find out which of these deserved closer investigation. An in-depth performance evaluation of all the algorithms was conducted using the most promising feature sets, subsequently investigating which specific features of the complete set contributed most to the performance. Finally, an analysis was made to see whether the differences in performance of the algorithms were statistically significant or not.

5.3.1 Initial exploratory results

For the initial experiment 17 different feature sets were constructed from (combinations of) the CHARM features and the SIFT features: CHARM features only (one set), SIFT features only (eight sets, one for each of the eight BoW vector lengths), and the union of CHARM and SIFT features (eight sets). To avoid prohibitive computation times in the cross-validation experiment (described next), it was first explored which of these feature sets would likely yield the best classification results with the considered machine learning algorithms. The feature sets were preprocessed by normalizing each feature to zero mean and unit standard deviation over all patches, and removing constant features (if present), to reduce the effect of possible outliers. To make this exploratory experiment more computationally feasible, a simpler resampling strategy was used rather than the described one, namely a single 10-fold cross-validation in the outer loop, and a holdout approach in the inner loop. In the latter, the optimal hyperparameters of the classification algorithms were obtained using a grid search on 2/3rd of the training set of the outer loop, and validated on the remaining 1/3rd. Observing the results (Fig. 5.4) indicates that both the absolute and the relative performance of the classifiers was quite different for the different feature sets. Specifically, for SVM and KNN, the best results were obtained with the SIFT features alone (for sufficiently large BoW vector lengths), while the CHARM features alone produced inferior results, and with the combination of CHARM and

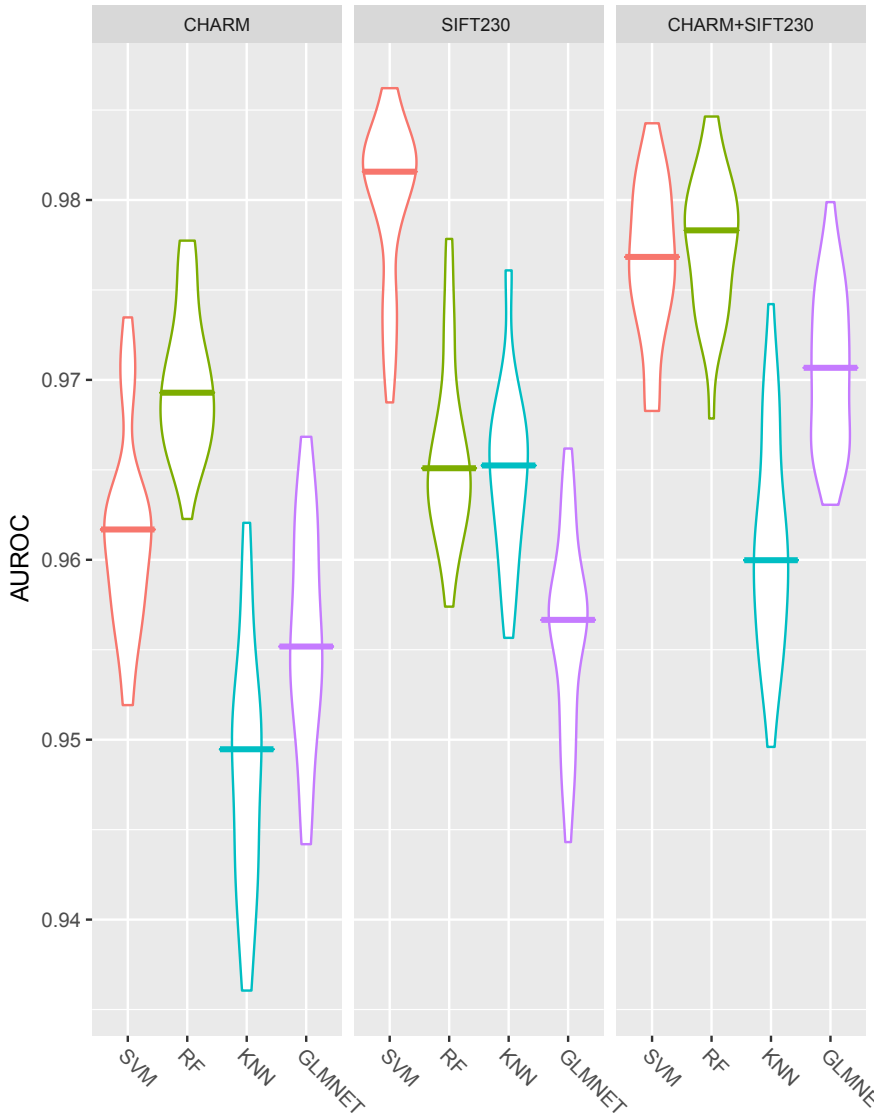


Figure 5.5: Results of the cross-validation experiment. Each of the considered classifiers (SVM, RF, KNN, GLMNET) was evaluated for each of the selected feature sets (CHARM, SIFT230, CHARM+SIFT230) using the performance measure (AUROC). The results are shown as violin plots, where the horizontal bar indicates the median value, the vertical extent is the interquartile range, and the width indicates the estimated probability density.

SIFT features these classifiers performed somewhere in between. For RF and GLMNET, on the other hand, the SIFT features alone yielded inferior results, and with

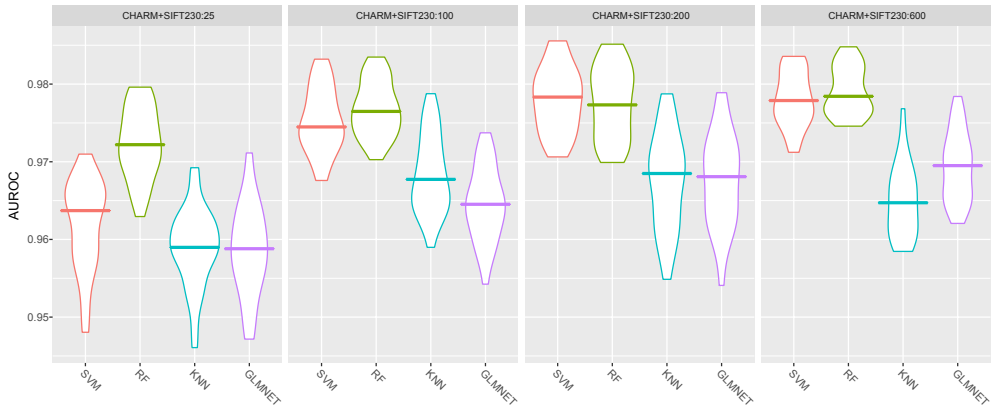


Figure 5.6: Performance (AUROC) of the considered classifiers (SVM, RF, KNN, GLMNET) for different feature subsets (the top 25, 100, 200, and 600 features from the CHARM+SIFT230 set). The results are shown as violin plots, where the horizontal bar indicates the median value, the vertical extent is the interquartile range, and the width indicates the estimated probability density.

the CHARM features alone these classifiers did not fare much better, but the combination of CHARM and SIFT features (for all BoW vector lengths) produced the best results.

Thus it was concluded that the cross-validation experiment should include both the CHARM and SIFT feature sets alone, as well as their combination, and the only way to reduce the computational cost of that experiment was to select a specific SIFT-BoW vector length. Overall, the results seemed to indicate that in most cases it is better to use larger vector lengths, and simply taking the maximum considered length (230) is a good choice.

5.3.2 Cross-validation results

Based on the results of the initial exploratory experiment three feature sets corresponding to CHARM features only, SIFT230 features only, and CHARM+SIFT230 features are selected to evaluate the four machine learning classifiers using a cross-validation experiment, involving an outer loop (3×10-fold) for performance assessment and an inner loop (holdout) for hyperparameter optimization as described. The results (Fig. 5.5) show that virtually all classifiers achieved AUROC values of >95% and, generally, SVM and RF outperformed KNN and GLMNET. Considering the different feature sets, it is observable that all classifiers except RF achieved better performance with the SIFT230 feature set than with the CHARM feature set. This is interesting since the latter is much more extensive (1,059 features of many different types) than the former (230 BoW clusters). Apparently the SIFT230 features are more descriptive of the image content in showcased application. This is confirmed by the results with the CHARM+SIFT230 feature set, which are consistently better than with the CHARM feature set alone. However, whereas RF and GLMNET

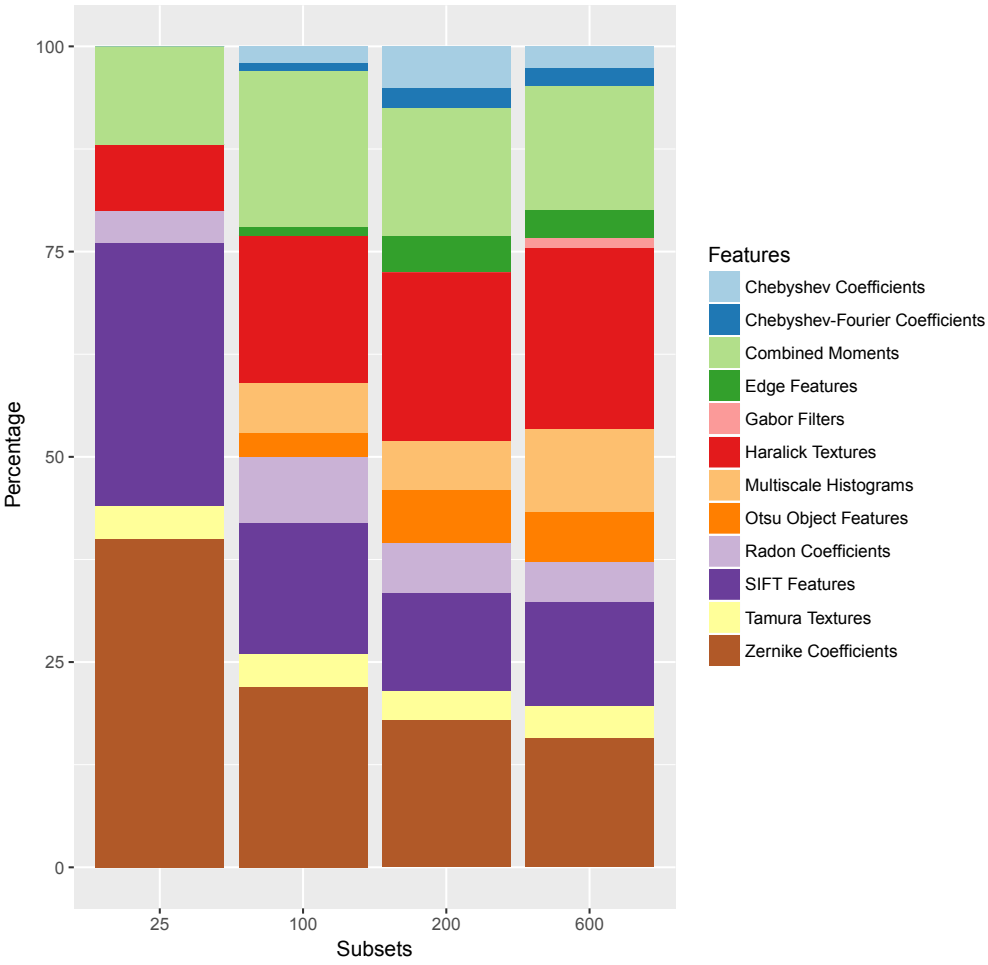


Figure 5.7: Cumulative percentages of the different types of features contained in the four subsets (the top 25, 100, 200, and 600 features selected from the CHARM+SIFT230 set).

performed best using the more extensive CHARM+SIFT230 set, SVM and KNN performed best using the SIFT230 set alone. Overall, the best results were obtained with the SVM classifier using the SIFT230 feature set, although SVM and RF using the combined CHARM+ SIFT230 features performed comparably (statistical significance is discussed in Section 5.3.4).

5.3.3 Feature selection results

Next, the complete CHARM+SIFT230 feature set was subjected to a feature selection experiment. Specifically, the aim was to find out which features contributed most to

the performance of the different classifiers, and whether these features alone could yield similar or even better classification performance than using the complete set, as that would make the classification task computationally cheaper. To this end,

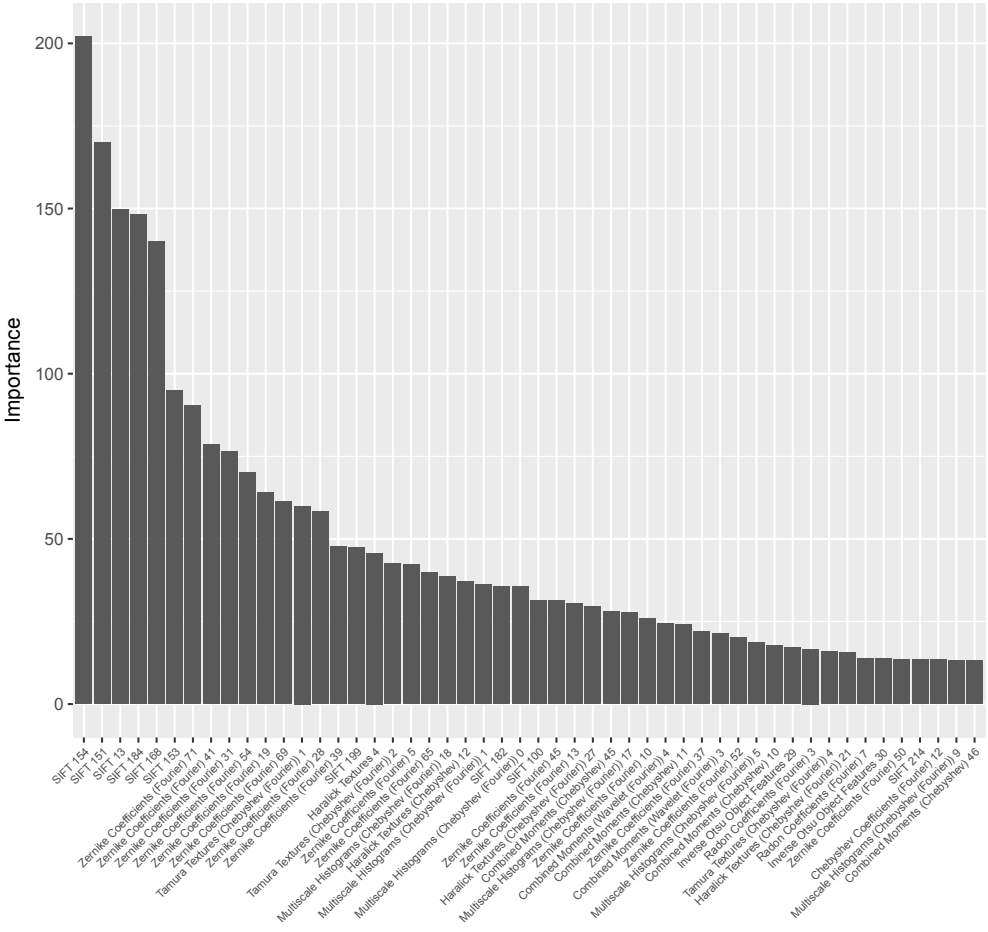


Figure 5.8: The 50 most important features from the CHARM+SIFT230: 600 feature subset used by the best performing classifier. Importance was calculated according to the Gini index of the RF classifier. The importance value for each feature was averaged over all runs and folds of the cross-validation experiment.

all 1,289 features are ranked using a *CForest* test [258] and considered four subsets, consisting of the top 25, 100, 200, and 600 features. The results (Fig. 5.6) agree with those of the previous experiment in that SVM and RF consistently outperformed KNN and GLMNET for all feature subsets. The results further indicate that the larger the number of top features, the better the performance of all four classifiers, but for most of them there was little improvement beyond the top 200 features. In fact, the scores of the best performing classifiers, SVM and RF, were very similar

for the CHARM+SIFT230:200 subset and the full CHARM+SIFT230 set, and with smaller standard deviations (statistical significance is discussed in Section 5.3.4). This indicates that the non-selected features provided noise rather than useful information to the classifiers.

Analyzing the types of features contained in the four subsets (Fig. 5.7), indicates that the top 25 subset is dominated by the SIFT features and the Zernike coefficients from CHARM, whereas the top 100, 200, and 600 subsets include many other types of features (about twice as many), in roughly similar proportions. These additional features contribute important information to the classification process, as follows from the fact that the performance of the larger subsets is considerably better than that of the top 25 subset. However, it remains elusive why these specific types of features are dominant. According to the feature selection results (Fig. 5.6), the best performing classification model is the RF using the CHARM+SIFT230:600 feature subset (AUROC = 0.9784), followed very closely by the SVM using the CHARM+SIFT230:200 feature subset (AUROC = 0.9783). Studying the importance of the features in the former model according to the *Gini* index [38], reveals (Fig. 5.8) that the most important features are indeed from the SIFT set together with the Zernike coefficients from the CHARM set. Other important top features from the CHARM set in decreasing order include the Tamura and Haralick textures, multiscale histograms, combined moments, and others (Fig. 5.7).

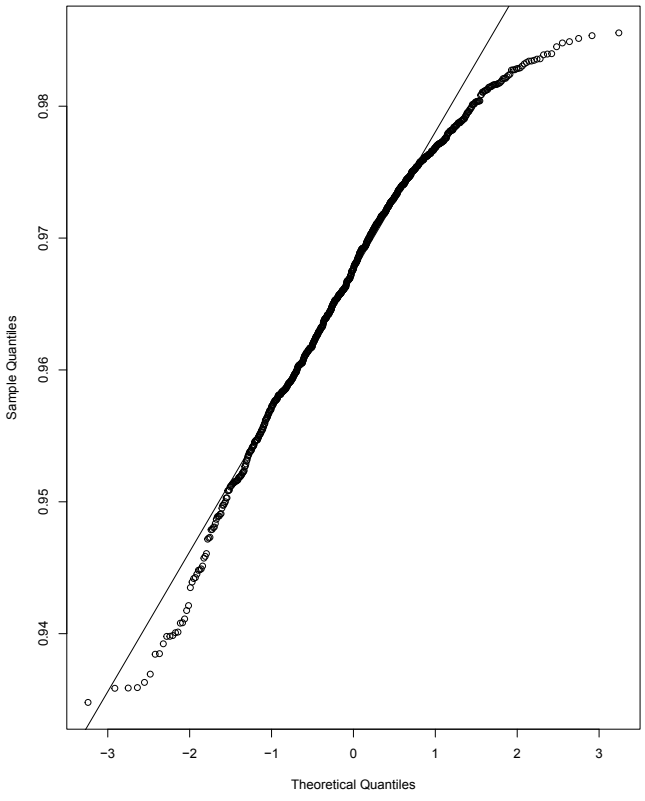


Figure 5.9: Quantile-quantile (Q-Q) plot of the theoretical normal distribution and the data samples used. Clearly, the computed values (small circles) deviate substantially from a straight line (the solid line is the least squares fit) and reveal a nonlinear relationship, leading to the conclusion that the used data is not normally distributed.

5.3.4 Statistical analysis results

Finally the statistical significance of the results (AUROC values) of the considered classification algorithms on the selected feature (sub)sets was examined, to see if any particular model (combination of features and classifier with corresponding optimal hyperparameters) is to be preferred for this application. There exist mainly two types of statistical test to do this: parametric and non-parametric. Although parametric tests can be more powerful, they require normality, independence, and heteroscedasticity of the data [87]. The first condition was checked using the Shapiro-Wilk test [239] with the null hypothesis that the used data follows normal distribution, eventually rejecting the null hypothesis with very significant values of $W = 0.97324$ and $p < 2.723 \cdot 10^{-11}$ (see also the Q-Q plot in Fig. 5.9). Since this already disqualifies parametric testing, there was no need to check the other conditions. Thus

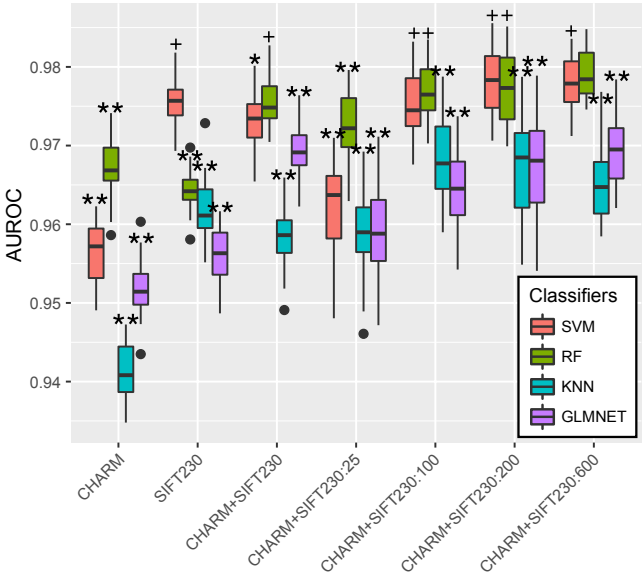


Figure 5.10: Results of the Friedman-Finner test showing the statistical significance of the differences in performance of the considered models (classifiers SVM, RF, KNN, and GLMNET, using any of the selected feature (sub)sets CHARM, SIFT230, CHARM+SIFT230, and the top 25, 100, 200, and 600 features of the latter) with respect to the control model (RF using CHARM+SIFT230:600). Performance values (AUROC) of each model from all runs and folds of the cross-validation experiment are summarized using the ggplot2 box plot. Significance with respect to the control model is indicated for $p > 0.05$ (+), and $0.01 < p < 0.05$ (*), and $p < 0.01$ (**).

a non-parametric test - the Friedman test [94] was used. It is known to yield conservative results in the case of relatively small numbers of algorithms and datasets [97]. The null hypothesis that all models yield the same performance on used data was rejected with very significant values of $\chi^2 = 657$ and $p < 2.25 \cdot 10^{-10}$. Since this means that at least some models are statistically significantly better or worse than

others, subsequent examination tested for significant differences between all pairs of models using the post-hoc Finner test [89], with the control model being the RF classifier using the CHARM+SIFT230:600 feature set, as it performed best in the feature selection experiment (Fig. 5.6).

The results (Fig. 5.10) show that several other models performed statistically similar to the control model. These include the SVM classifier using the SIFT230 feature set or the top 100, 200, or 600 features of the CHARM+SIFT230 set. Other statistically similar models include the RF classifier using the CHARM+SIFT230 feature set, or just the top 100 or 200 features of the latter. None of the models based on the KNN and GLMNET classifiers performed statistically similar to the control model.

5.4 Discussion and conclusions

The aim of the presented study was to find out which machine learning based classification algorithms and which commonly used feature extraction algorithms would be most suited for the task of detecting neurons in high-content fluorescence microscopy image data typically acquired in screening experiments. To this end, four popular classifiers (SVM, RF, KNN, GLMNET) and two popular feature extraction tools (CHARM and SIFT) were considered, and various experiments and statistical analyses performed to narrow down and compare the many possible models (combinations of classifiers and (sub)sets of features).

Showcased results point to the conclusion that of all considered classifiers, SVM and RF generally work best, provided they are fed with the right sets of features. Statistically similar performance was observed with the following models: SVM using SIFT (230 features), SVM using CHARM+SIFT (the top 100, 200, or 600 features), and RF using CHARM+SIFT (the full 1,289 features or only the top 100, 200, or 600 features). In the course of conducted study, the potential of several alternative features was also explored, such as the histogram of oriented gradients (HOG) [64] and spatial pyramid matching (SPM) [143] based on sparse coding (ScSPM) [296], but the results were not as good. In the spirit of Occam's razor principle [81,123,126], which considers the simplest explanation of natural phenomena to be the closest to the truth, additional experimentation sought for the smallest possible classification model capable of determining with high accuracy whether or not a new unseen image patch contains neuron structures. Generally speaking, in order to achieve good generalization in a classification task, it is required to have a sufficient number of samples and to minimize model complexity [112]. Since the data used in this study is currently rather limited, the investigation was initiated by considering state-of-the-art classification algorithms involving explicit calculation of features, and using state-of-the-art algorithms for extracting a very wide variety and large number of features. In the future, when more annotated data becomes available for this study, deep learning approaches are expected to be good and possibly superior alternatives, as they have been very successful in many other applications [29,111,145,152,234,240,262]. To get an impression of their performance on current data, a pilot experiment was performed with three convolutional neural networks (CNNs). The first was a home-built network

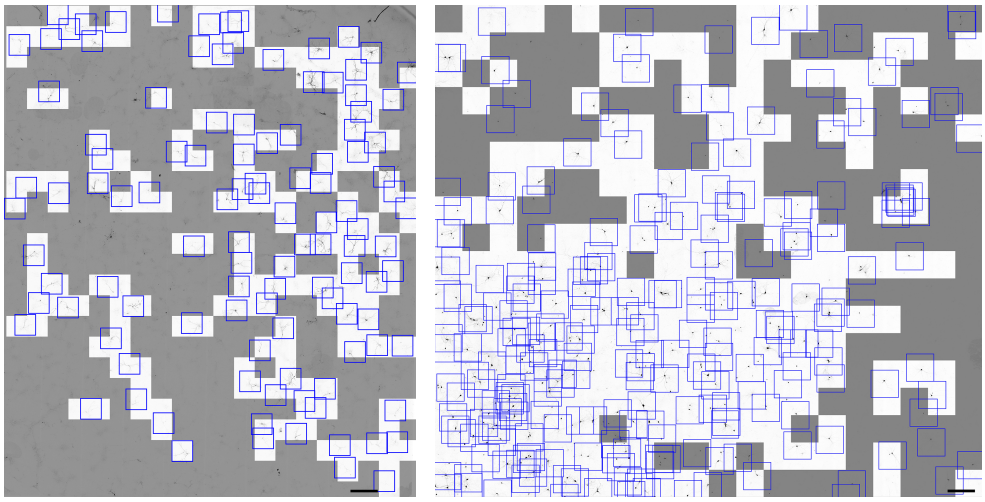


Figure 5.11: Example of neuron detection in high-content fluorescence microscopy images. The images are shown with inverted intensities (dark grayscale parts) compared to the original. Left: One of the eight images used in the cross-validation experiment. Right: A new image acquired in a later experiment and not used in the cross-validation experiment. This detection example uses the SVM classifier with the SIFT230 feature set to classify square patches from a superimposed grid as neuron (bright grayscale) versus non-neuron (dark grayscale). The detected neuron regions correspond very well with the expert human annotations (blue squares). Scale bars: 500 μm.

(HBN17) with 17 convolutional layers, interspersed with six max-pooling layers, and followed by two fully connected layers outputting the two class probabilities (neuron versus background). The second network was VGG19 [244], with incorporated modification due to the image patch sizes. Namely, in the presented study, patch sizes are more than four times larger than what VGG19 was originally designed for. This increases the number of network parameters and thus the memory usage to the point that the ability to train the network on our available computers was severely limited. Therefore the number of filters was reduced in the convolution layers by a factor of 16. Also, the network returns only two class probabilities to match this application. The third network was ResNet50 [119] modified so as to return only two class probabilities. Categorical cross-entropy [101] was used as the loss function when training the networks, and Adam [134] as the optimizer. The networks were trained on the same balanced data set as the classifiers studied in this work and were tested using the same 3 $\times$ 10-fold cross-validation approach. The results showed that VGG19 performed best (median AUROC of 0.960), followed by ResNet50 (median AUROC of 0.947), and HBN17 (median AUROC of 0.936). Clearly the networks are as yet outperformed by the best classifiers considered in this study. Better results may be achieved not only by acquiring more data but also by applying stronger data

augmentation than done here. Another potential direction for future research would be to reformulate the problem as a multiclass detection challenge, distinguishing not only between neurons and background, but also incomplete or out-of-focus neurons, astrocytes, and artifacts.

Achieving AUROC values between 0.97 and 0.98, the best models considered in the present study are already very suitable for detecting neurons in high-content fluorescence microscopy images. As an example the model using the SVM classifier and the SIFT230 feature set was applied to one of the dataset images (Fig. 5.11). In addition, to investigate the generalizability, it was also applied to a new, "unseen" image from a new experiment. In that experiment, to introduce some variability, a transfection method with higher efficiency [37] was used, resulting in higher intensities and larger numbers of neurons in the field of view. In both images, to detect the neurons, a very simple and low-cost detection approach was used, where square patches (same patch size as used throughout this study) from a superimposed grid were classified individually as neuron versus non-neuron. If needed, more sophisticated (but more computationally costly) detection schemes with higher localization precision could be easily made, by using finer grids with overlapping patches (keeping the same patch size) and segmenting the positive responses. In high-content fluorescence microscopy neuron image analysis, detection is only the first step in a much more comprehensive pipeline developed for fully automated neuron screening, where the actual neuron reconstruction and downstream morphological analysis is based on much higher-resolution images taken at the locations detected in the low-resolution high-content images. The results presented in this study indicate that machine learning approaches are very suitable for the initial detection task and can drastically reduce the high-resolution scan time and analysis.

Bibliography

[1] M. D. Abràmoff, P. J. Magalhães, S. J. Ram, "Image processing with ImageJ", *Biophotonics International*, vol. 11, no. 7, pp. 36–43, 2004.

[2] L. Acciai, P. Soda, G. Iannello, "Automated neuron tracing methods: an updated account", *Neuroinformatics*, vol. 14, no. 4, pp. 353–367, 2016.

[3] G. Agam, S. G. Armato III, C. Wu, "Vessel tree reconstruction in thoracic CT scans with application to nodule detection", *IEEE Transactions on Medical Imaging*, vol. 24, no. 4, pp. 486–499, 2005.

[4] A. M. Aibinu, M. I. Iqbal, A. A. Shafie, M. J. E. Salami, M. Nilsson, "Vascular intersection detection in retina fundus images using a new hybrid approach", *Computers in Biology and Medicine*, vol. 40, no. 1, pp. 81–89, 2010.

[5] Y. Al-Kofahi, N. Dowell-Mesfin, C. Pace, W. Shain, J. N. Turner, B. Roysam, "Improved detection of branching points in algorithms for automated neuron tracing from 3D confocal images", *Cytometry Part A*, vol. 73, no. 1, pp. 36–43, 2008.

[6] J. L. Anderl, S. Redpath, A. J. Ball, "A neuronal and astrocyte co-culture assay for high content analysis of neurotoxicity", *Journal of Visualized Experiments: JoVE*, no. 27, 2009.

[7] B. H. Anderton, L. Callahan, P. Coleman, P. Davies, D. Flood, G. A. Jicha, T. Ohm, C. Weaver, "Dendritic changes in Alzheimer's disease and factors that may underlie these changes", *Progress in Neurobiology*, vol. 55, no. 6, pp. 595–609, 1998.

[8] P. P. M. A. Antony, C. Trefois, A. Stojanovic, A. S. Baumuratov, K. Kozak, "Light microscopy applications in systems biology: opportunities and challenges", *Cell Communication and Signaling*, vol. 11, no. 1, p. 24, Apr 2013.

[9] I. Arganda-Carreras, R. Fernández-González, A. Muñoz-Barrutia, C. Ortiz-De-Solorzano, "3D reconstruction of histological sections: Application to mammary gland tissue", *Microscopy Research and Technique*, vol. 73, no. 11, pp. 1019–1029, 2010.

[10] I. Arganda-Carreras, V. Kaynig, C. Rueden, K. W. Eliceiri, J. Schindelin, A. Cardona, H. Sebastian Seung, "Trainable Weka segmentation: a machine learning tool for microscopy pixel classification", *Bioinformatics*, vol. 33, no. 15, pp. 2424–2426, 2017.

[11] R. Armañanzas & G. A. Ascoli, "Towards the automatic classification of neurons", *Trends in Neurosciences*, vol. 38, no. 5, pp. 307–318, 2015.

[12] M. S. Arulampalam, S. Maskell, N. Gordon, T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking", *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 174–188, 2002.

[13] G. Ascoli, *Trees of the Brain, Roots of the Mind*, The MIT Press, MIT Press, Cambridge, MA, 2015.

[14] G. A. Ascoli, *Computational Neuroanatomy: Principles and Methods*, Springer Science & Business Media, 2002.

[15] G. A. Ascoli, D. E. Donohue, M. Halavi, "NeuroMorpho.Org: a central resource for neuronal morphologies", *Journal of Neuroscience*, vol. 27, no. 35, pp. 9247–9251, 2007.

[16] G. A. Ascoli, J. L. Krichmar, R. Scorcioni, S. J. Nasuto, S. L. Senft, G. Krichmar, "Computer generation and quantitative morphometric analysis of virtual neurons", *Anatomy and Embryology*, vol. 204, no. 4, pp. 283–301, 2001.

[17] G. Azzopardi & N. Petkov, "Automatic detection of vascular bifurcations in segmented retinal images using trainable COSFIRE filters", *Pattern Recognition Letters*, vol. 34, no. 8, pp. 922–933, 2013.

[18] R. Bakker & P. H. E. Tiesinga, "Web-based neuron morphology viewer as an aid to develop new standards for neuron morphology file formats", *Frontiers in Neuroinformatics*, vol. 10, no. 79, 2016.

[19] Y. Bar-Shalom & X.-R. Li, *Multitarget-Multisensor Tracking: Principles and Techniques*, YBS Publishing London, UK, 1995.

[20] Y. Bar-Shalom, X. R. Li, T. Kirubarajan, *Estimation with Applications to Tracking and Navigation: Theory Algorithms and Software*, John Wiley & Sons, 2004.

[21] E. Bas & D. Erdogmus, "Principal curves as skeletons of tubular objects", *Neuroinformatics*, vol. 9, no. 2-3, pp. 181–191, 2011.

[22] S. Basu, B. Condron, A. Aksel, S. T. Acton, "Segmentation and tracing of single neurons from 3D confocal microscope images", *IEEE Journal of Biomedical and Health Informatics*, vol. 17, no. 2, pp. 319–335, 2013.

[23] S. Basu, W. T. Ooi, D. Racoceanu, "Neurite tracing with object process", *IEEE Transactions on Medical Imaging*, vol. 35, no. 6, pp. 1443–1451, 2016.

[24] S. Basu & D. Racoceanu, "Reconstructing neuronal morphology from microscopy stacks using fast marching", in *IEEE International Conference on Image Processing (ICIP)*, IEEE, pp. 3597–3601, 2014.

[25] P. V. Belichenko & A. Dahlström, "Confocal laser scanning microscopy and 3-D reconstructions of neuronal structures in human brain cortex", *Neuroimage*, vol. 2, no. 3, pp. 201–207, 1995.

[26] V. Bevilacqua, S. Cambò, L. Cariello, G. Mastronardi, "A combined method to detect retinal fundus features", in *Proceedings of IEEE European Conference on Emergent Aspects in Clinical Data Analysis*, pp. 1–6, 2005.

[27] V. Bevilacqua, L. Cariello, M. Giannini, G. Mastronardi, V. Santarcangelo, R. Scaramuzzi, A. Troccoli, "A comparison between a geometrical and an ANN based method for retinal bifurcation points extraction.", *Journal of Universal Computer Science*, vol. 15, no. 13, pp. 2608–2621, 2009.

[28] A. Bhuiyan, B. Nath, J. Chua, K. Ramamohanarao, "Automatic detection of vascular bifurcations and crossovers from color retinal fundus images", in *International Conference on Signal-Image Technologies and Internet-Based System (SITIS)*, IEEE, pp. 711–718, 2007.

[29] M. Bianchini & F. Scarselli, "On the complexity of neural network classifiers: A comparison between shallow and deep architectures", *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 8, pp. 1553–1565, 2014.

[30] B. Bischl, M. Lang, L. Kotthoff, J. Schiffner, J. Richter, E. Studerus, G. Casalicchio, Z. M. Jones, "mlr: Machine learning in R", *Journal of Machine Learning Research*, vol. 17, no. 1, pp. 5938–5942, 2016.

[31] B. Bischl, O. Mersmann, H. Trautmann, C. Weihs, "Resampling methods for meta-model validation with recommendations for evolutionary computation", *Evolutionary Computation*, vol. 20, no. 2, pp. 249–275, 2012.

[32] C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer, New York, NY, 2006.

[33] C. Blum & M. Blesa, "Combining ant colony optimization with dynamic programming for solving the k-cardinality tree problem", in *International Work-Conference on Artificial Neural Networks*, Springer, pp. 25–33, 2005.

[34] B. E. Boser, I. M. Guyon, V. N. Vapnik, "A training algorithm for optimal margin classifiers", in *Proceedings of the 5th Annual Workshop on Computational Learning Theory*, ACM, pp. 144–152, 1992.

[35] N. Bougen-Zhukov, S. Y. Loh, H. K. Lee, L.-H. Loo, "Large-scale image-based screening and profiling of cellular phenotypes", *Cytometry Part A*, vol. 91, no. 2, pp. 115–125, 2017.

[36] P. Branco, L. Torgo, R. P. Ribeiro, "A survey of predictive modeling on imbalanced domains", *ACM Computing Surveys (CSUR)*, vol. 49, no. 2, pp. 31:1–31:50, 2016.

[37] P. J. Bredenbeek, I. Frolov, C. Rice, S. Schlesinger, "Sindbis virus expression vectors: packaging of RNA replicons by using defective helper RNAs.", *Journal of Virology*, vol. 67, no. 11, pp. 6439–6446, 1993.

[38] L. Breiman, "Random forests", *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[39] K. M. Brown, G. Barrionuevo, A. J. Canty, V. De Paola, J. A. Hirsch, G. S. Jefferis, J. Lu, M. Snippe, I. Sugihara, G. A. Ascoli, "The DIADEM data sets: representative light microscopy images of neuronal morphology to advance automation of digital reconstructions", *Neuroinformatics*, vol. 9, no. 2-3, pp. 143–157, 2011.

[40] C. J. Burges, "A tutorial on support vector machines for pattern recognition", *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 121–167, 1998.

[41] H. Cai, X. Xu, J. Lu, J. W. Lichtman, S. Yung, S. T. Wong, "Repulsive force based snake model to segment and track neuronal axons in 3D microscopy image stacks", *NeuroImage*, vol. 32, no. 4, pp. 1608–1620, 2006.

[42] D. Calvo, M. Ortega, M. G. Penedo, J. Rouco, "Automatic detection and characterisation of retinal vessel tree bifurcations and crossovers in eye fundus images", *Computer Methods and Programs in Biomedicine*, vol. 103, no. 1, pp. 28–38, 2011.

[43] A. Can, H. Shen, J. N. Turner, H. L. Tanenbaum, B. Roysam, "Rapid automated tracing and feature extraction from retinal fundus images using direct exploratory algorithms", *IEEE Transactions on Information Technology in Biomedicine*, vol. 3, no. 2, pp. 125–138, 1999.

[44] R. Cannon, D. Turner, G. Pyapali, H. Wheal, "An on-line archive of reconstructed hippocampal neurons", *Journal of Neuroscience Methods*, vol. 84, no. 1, pp. 49–54, 1998.

[45] J. Capowski & M. Sedivec, "Accurate computer reconstruction and graphics display of complex neurons utilizing state-of-the-art interactive techniques", *Computers and Biomedical Research*, vol. 14, no. 6, pp. 518–532, 1981.

[46] J. J. Capowski, "Computer-aided reconstruction of neuron trees from several serial sections", *Computers and Biomedical Research*, vol. 10, no. 6, pp. 617–629, 1977.

[47] P. Charoenkwan, E. Hwang, R. W. Cutler, H.-C. Lee, L.-W. Ko, H.-L. Huang, S.-Y. Ho, "HCS-Neurons: identifying phenotypic changes in multi-neuron images upon drug treatments of high-content screening", *BMC Bioinformatics*, vol. 14, no. 16, p. S12, 2013.

[48] N. V. Chawla, K. W. Bowyer, L. O. Hall, W. P. Kegelmeyer, "SMOTE: synthetic minority over-sampling technique", *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, 2002.

[49] N. V. Chawla, N. Japkowicz, A. Kotcz, "Editorial: special issue on learning from imbalanced data sets", *ACM SIGKDD Explorations Newsletter*, vol. 6, no. 1, pp. 1–6, June 2004.

[50] M. K. Cheezum, W. F. Walker, W. H. Guilford, "Quantitative comparison of algorithms for tracking single fluorescent particles", *Biophysical Journal*, vol. 81, no. 4, pp. 2378–2388, 2001.

[51] H. Chen, H. Xiao, T. Liu, H. Peng, "SmartTracing: self-learning-based neuron reconstruction", *Brain Informatics*, vol. 2, no. 3, pp. 135–144, 2015.

[52] Y. Cheng, "Mean shift, mode seeking, and clustering", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 8, pp. 790–799, 1995.

[53] N. Chenouard, I. Smal, F. De Chaumont, M. Maška, I. F. Sbalzarini, Y. Gong, J. Cardinale, C. Carthel, S. Coraluppi, M. Winter, others, "Objective comparison of particle tracking methods", *Nature Methods*, vol. 11, no. 3, pp. 281–289, 2014.

[54] M. Chimani, M. Kandyba, I. Ljubić, P. Mutzel, "Obtaining optimal k-cardinality trees fast", *Journal of Experimental Algorithmics (JEA)*, vol. 14, pp. 5:2.5–5:2.23, Jan. 2010.

[55] A. Choromanska, S.-F. Chang, R. Yuste, "Automatic reconstruction of neural morphologies with multi-scale tracking", *Frontiers in Neural Circuits*, vol. 6, p. 25, 2012.

[56] P. Chothani, V. Mehta, A. Stepanyants, "Automated tracing of neurites from light microscopy stacks of images", *Neuroinformatics*, vol. 9, no. 2-3, pp. 263–278, 2011.

[57] D. Clark, I. T. Ruiz, Y. Petillot, J. Bell, "Particle PHD filter multiple target tracking in sonar image", *IEEE Transactions on Aerospace and Electronic Systems*, vol. 43, no. 1, pp. 409–416, 2007.

[58] A. Cohen, B. Roysam, J. Turner, "Automated tracing and volume measurements of neurons from 3-D confocal fluorescence microscopy data", *Journal of Microscopy*, vol. 173, no. 2, pp. 103–114, 1994.

[59] T. Cover & P. Hart, "Nearest neighbor pattern classification", *IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 21–27, 1967.

[60] G. Cox, *Optical Imaging Techniques in Cell Biology*, CRC Press, 2012.

[61] N. Cristianini, J. Shawe-Taylor, others, *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*, Cambridge University Press, 2000.

[62] G. Cuesto, L. Enriquez-Barreto, C. Caramés, M. Cantarero, X. Gasull, C. Sandi, A. Ferrús, Á. Acebes, M. Morales, "Phosphoinositide-3-kinase activation controls synaptogenesis and spinogenesis in hippocampal neurons", *Journal of Neuroscience*, vol. 31, no. 8, pp. 2721–2733, 2011.

[63] H. Cuntz, F. Forstner, A. Borst, M. Häusser, "One rule to grow them all: a general theory of neuronal branching and its practical application", *PLoS Computational Biology*, vol. 6, no. 8, p. e1000877, 2010.

[64] N. Dalal & B. Triggs, "Histograms of oriented gradients for human detection", in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, IEEE, pp. 886–893, 2005.

[65] S. Daskalaki, I. Kopanas, N. Avouris, "Evaluation of classifiers for an uneven class distribution problem", *Applied Artificial Intelligence*, vol. 20, no. 5, pp. 381–417, 2006.

[66] J. De, L. Cheng, X. Zhang, F. Lin, H. Li, K. H. Ong, W. Yu, Y. Yu, S. Ahmed, "A graph-theoretical approach for tracing filamentary structures in neuronal and retinal images", *IEEE Transactions on Medical Imaging*, vol. 35, no. 1, pp. 257–272, 2016.

[67] J. DeFelipe, L. Alonso-Nanclares, J. I. Arellano, "Microstructure of the neocortex: comparative aspects", *Journal of Neurocytology*, vol. 31, no. 3-5, pp. 299–316, 2002.

[68] J. DeFelipe & I. Fariñas, "The pyramidal neuron of the cerebral cortex: morphological and chemical characteristics of the synaptic inputs", *Progress in Neurobiology*, vol. 39, no. 6, pp. 563–607, 1992.

[69] J. DeFelipe, P. L. López-Cruz, R. Benavides-Piccione, C. Bielza, P. Larrañaga, S. Anderson, A. Burkhalter, B. Cauli, A. Fairén, D. Feldmeyer, others, "New insights into the classification and nomenclature of cortical GABAergic interneurons", *Nature Reviews Neuroscience*, vol. 14, no. 3, pp. 202–216, 2013.

[70] L. Dehmelt, G. Poplawski, E. Hwang, S. Halpain, "NeuriteQuant: an open source toolkit for high content screens of neuronal morphogenesis", *BMC Neuroscience*, vol. 12, no. 1, p. 100, Oct 2011.

[71] V. J. Dercksen, H.-C. Hege, M. Oberlaender, "The Filament Editor: an interactive software environment for visualization, proof-editing and analysis of 3D neuron morphology", *Neuroinformatics*, vol. 12, no. 2, pp. 325–339, 2014.

[72] F. Deschênes & D. Ziou, "Detection of line junctions and line terminations using curvilinear features", *Pattern Recognition Letters*, vol. 21, no. 6, pp. 637–649, 2000.

[73] E. W. Dijkstra, "A note on two problems in connexion with graphs", *Numerische Mathematik*, vol. 1, no. 1, pp. 269–271, 1959.

[74] A. Dima, M. Scholz, K. Obermayer, "Automatic segmentation and skeletonization of neurons from confocal microscopy images based on the 3-D wavelet transform", *IEEE Transactions on Image Processing*, vol. 11, no. 7, pp. 790–801, 2002.

[75] D. E. Donohue & G. A. Ascoli, "A comparative computer simulation of dendritic morphology", *PLoS Computational Biology*, vol. 4, no. 6, pp. 1–15, 06 2008.

[76] D. E. Donohue & G. A. Ascoli, "Automated reconstruction of neuronal morphology: an overview", *Brain Research Reviews*, vol. 67, no. 1, pp. 94–102, 2011.

[77] A. Doucet, N. De Freitas, N. Gordon, "An introduction to sequential Monte Carlo methods", in *Sequential Monte Carlo Methods in Practice*, Springer, pp. 3–14, 2001.

[78] A. Doucet, S. Godsill, C. Andrieu, "On sequential Monte Carlo sampling methods for Bayesian filtering", *Statistics and Computing*, vol. 10, no. 3, pp. 197–208, 2000.

[79] W. Doyle, "Operations useful for similarity-invariant pattern recognition", *Journal of the ACM (JACM)*, vol. 9, no. 2, pp. 259–267, 1962.

[80] M. Dragunow, "High-content analysis in neuroscience", *Nature Reviews Neuroscience*, vol. 9, no. 10, pp. 779–788, 2008.

[81] M. K. Ebrahimpour, M. Zare, M. Eftekhari, G. Aghamolaei, "Occam's razor in dimension reduction: Using reduced row Echelon form for finding linear independent features in high dimensional microarray datasets", *Engineering Applications of Artificial Intelligence*, vol. 62, pp. 214–221, 2017.

[82] L. Enriquez-Barreto, G. Cuesto, N. Dominguez-Iturza, E. Gavilán, D. Ruano, C. Sandi, A. Ferrandez-Ruiz, G. Martín-Vázquez, O. Herreras, M. Morales, "Learning improvement after PI3K activation correlates with de novo formation of functional small spines", *Frontiers in Molecular Neuroscience*, vol. 6, p. 54, 2014.

[83] L. Enriquez-Barreto & M. Morales, "The PI3K signaling pathway as a pharmacological target in autism related disorders and schizophrenia", *Molecular and Cellular Therapies*, vol. 4, no. 1, p. 2, 2016.

[84] T. Fawcett, "An introduction to ROC analysis", *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861–874, 2006.

[85] L. Fei-Fei & P. Perona, "A bayesian hierarchical model for learning natural scene categories", in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2, IEEE, pp. 524–531, 2005.

[86] L. Feng, T. Zhao, J. Kim, "neuTube 1.0: A new design for efficient neuron reconstruction software based on the SWC format", *Eneuro*, vol. 2, no. 1, pp. ENEURO–0049, 2015.

[87] C. Fernandez-Lozano, M. Gestal, C. R. Munteanu, J. Dorado, A. Pazos, "A methodology for the design of experiments in computational intelligence with multiple regression models", *PeerJ*, vol. 4, p. e2721, 2016.

[88] T. Ferreira & W. Rasband, "ImageJ user guide", *ImageJ/Fiji*, vol. 1, 2012.

[89] H. Finner, "On a monotonicity problem in step-down multiple test procedures", *Journal of the American Statistical Association*, vol. 88, no. 423, pp. 920–923, 1993.

[90] B. P. Flannery, W. H. Press, S. A. Teukolsky, W. Vetterling, "Numerical recipes in C", *Press Syndicate of the University of Cambridge, New York*, 1992.

[91] G. Forman & M. Scholz, "Apples-to-apples in cross-validation studies: pitfalls in classifier performance measurement", *ACM SIGKDD Explorations Newsletter*, vol. 12, no. 1, pp. 49–57, 2010.

[92] A. F. Frangi, W. J. Niessen, K. L. Vincken, M. A. Viergever, "Multiscale vessel enhancement filtering", in *International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, Springer, pp. 130–137, 1998.

[93] J. Friedman, T. Hastie, R. Tibshirani, "Regularization paths for generalized linear models via coordinate descent", *Journal of Statistical Software*, vol. 33, no. 1, pp. 1–22, 2010.

[94] M. Friedman, "A comparison of alternative tests of significance for the problem of m rankings", *The Annals of Mathematical Statistics*, vol. 11, no. 1, pp. 86–92, 1940.

[95] D. Gabor, "Theory of communication. Part 1: The analysis of information", *Journal of the Institution of Electrical Engineers-Part III: Radio and Communication Engineering*, vol. 93, no. 26, pp. 429–441, 1946.

[96] R. Gala, J. Chapeton, J. Jitesh, C. Bhavsar, A. Stepanyants, "Active learning of neuron morphology for accurate automated tracing of neurites", *Frontiers in Neuroanatomy*, vol. 8, p. 37, 2014.

[97] S. García, A. Fernández, J. Luengo, F. Herrera, "Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power", *Information Sciences*, vol. 180, no. 10, pp. 2044–2064, 2010.

[98] V. García, R. A. Mollineda, J. S. Sánchez, "A bias correction function for classification performance assessment in two-class imbalanced problems", *Knowledge-Based Systems*, vol. 59, pp. 66–74, 2014.

[99] L. M. Garcia-Segura & J. Perez-Marquez, "A new mathematical function to evaluate neuronal morphology using the Sholl analysis", *Journal of Neuroscience Methods*, vol. 226, pp. 103–109, 2014.

[100] J. C. Gensel, D. L. Schonberg, J. K. Alexander, D. M. McTigue, P. G. Popovich, "Semi-automated Sholl analysis for quantifying changes in growth and differentiation of neurons and glia", *Journal of Neuroscience Methods*, vol. 190, no. 1, pp. 71–79, 2010.

[101] A. Ghosh, H. Kumar, P. Sastry, "Robust loss functions under label noise for deep neural networks", in *AAAI Conference on Artificial Intelligence*, pp. 1919–1925, 2017.

[102] T. A. Gillette, K. M. Brown, K. Svoboda, Y. Liu, G. A. Ascoli, "DIADEMchallenge.Org: A compendium of resources fostering the continuous development of automated neuronal reconstruction", *Neuroinformatics*, vol. 9, no. 2, pp. 303–304, 2011.

[103] E. Glaser & H. Van der Loos, "A semi-automatic computer-microscope for the analysis of neuronal morphology", *IEEE Transactions on Biomedical Engineering*, no. 1, pp. 22–31, 1965.

[104] J. R. Glaser & E. M. Glaser, "Neuron imaging with Neurolucida—a PC-based system for image combining microscopy", *Computerized Medical Imaging and Graphics*, vol. 14, no. 5, pp. 307–317, 1990.

[105] M. Glickstein, "Golgi and Cajal: The neuron doctrine and the 100th anniversary of the 1906 Nobel Prize", *Current Biology*, vol. 16, no. 5, pp. R147–R151, 2006.

[106] N. Gomez, Y. Lu, S. Chen, C. E. Schmidt, "Immobilized nerve growth factor and microtopography have distinct effects on polarization versus axon elongation in hippocampal cells in culture", *Biomaterials*, vol. 28, no. 2, pp. 271–284, 2007.

[107] G. González, E. Türetken, P. Fua, others, "Delineating trees in noisy 2D images and 3D image-stacks", in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, pp. 2799–2806, 2010.

[108] A. Gosain & S. Sardana, "Handling class imbalance problem using oversampling techniques: A review", in *International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, IEEE, pp. 79–85, 2017.

[109] K. Goslin, H. Asmussen, G. Banker, "Rat hippocampal neurons in low-density culture", in *Culturing Nerve Cells*, K. Goslin & G. Banker (eds.), MIT Press, Cambridge, MA, Ch. 13, pp. 339–370, 1998.

[110] I. S. Gradshteyn & I. M. Ryzhik, *Table of integrals, series, and products*, Academic Press, 2014.

[111] H. Greenspan, B. Van Ginneken, R. M. Summers, "Guest editorial deep learning in medical imaging: Overview and future promise of an exciting new technique", *IEEE Transactions on Medical Imaging*, vol. 35, no. 5, pp. 1153–1159, 2016.

[112] P. Gupta, S. S. Batra, others, "Sparse short-term time series forecasting models via minimum model complexity", *Neurocomputing*, vol. 243, pp. 1–11, 2017.

[113] E. Hadjidemetriou, M. D. Grossberg, S. K. Nayar, "Spatial information in multiresolution histograms", in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, IEEE, pp. I–I, 2001.

[114] G. Haixiang, L. Yijing, J. Shang, G. Mingyun, H. Yuanyue, G. Bing, "Learning from class-imbalanced data: Review of methods and applications", *Expert Systems with Applications*, vol. 73, pp. 220–239, 2017.

[115] M. Halavi, K. A. Hamilton, R. Parekh, G. Ascoli, "Digital reconstructions of neuronal morphology: three decades of research trends", *Frontiers in Neuroscience*, vol. 6, p. 49, 2012.

[116] T. Hansen & H. Neumann, "Neural mechanisms for the robust representation of junctions", *Neural Computation*, vol. 16, no. 5, pp. 1013–1037, 2004.

[117] R. M. Haralick, K. Shanmugam, I. Dinstein, others, "Textural features for image classification", *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 3, no. 6, pp. 610–621, 1973.

[118] H. He & E. A. Garcia, "Learning from imbalanced data", *IEEE Transactions on Knowledge and Data Engineering*, no. 9, pp. 1263–1284, 2008.

[119] K. He, X. Zhang, S. Ren, J. Sun, "Deep residual learning for image recognition", in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

[120] W. He, T. A. Hamilton, A. R. Cohen, T. J. Holmes, C. Pace, D. H. Szarowski, J. N. Turner, B. Roysam, "Automated three-dimensional tracing of neurons in confocal and brightfield images", *Microscopy and Microanalysis*, vol. 9, no. 4, pp. 296–310, 2003.

[121] K. Hechenbichler & K. Schliep, "Weighted k-nearest-neighbor techniques and ordinal classification", *Sonderforschungsbereich*, vol. 386, no. 399, pp. 1–16, 2004.

[122] S.-Y. Ho, C.-Y. Chao, H.-L. Huang, T.-W. Chiu, P. Charoenkwan, E. Hwang, "NeurphologyJ: An automatic neuronal morphology quantification method and its application in pharmacological discovery", *BMC Bioinformatics*, vol. 12, no. 1, p. 230, 2011.

[123] X. Hong, J. Gao, S. Chen, C. J. Harris, "Particle swarm optimisation assisted classification using elastic net prefiltering", *Neurocomputing*, vol. 122, pp. 210–220, 2013.

[124] A. Hoover, V. Kouznetsova, M. Goldbaum, "Locating blood vessels in retinal images by piecewise threshold probing of a matched filter response", *IEEE Transactions on Medical Imaging*, vol. 19, no. 3, pp. 203–210, 2000.

[125] P. Horvath, T. Wild, U. Kutay, G. Csucs, "Machine learning improves the precision and robustness of high-content screens: using nonlinear multiparametric methods to analyze screening results", *Journal of Biomolecular Screening*, vol. 16, no. 9, pp. 1059–1067, 2011.

[126] G. Iacca, F. Neri, E. Mininno, Y.-S. Ong, M.-H. Lim, "Ockham's razor in memetic computing: three stage optimal memetic exploration", *Information Sciences*, vol. 188, pp. 17–43, 2012.

[127] D. Iber & D. Menshykau, "The control of branching morphogenesis", *Open Biology*, vol. 3, no. 9, p. 130088, 2013.

[128] S. Jain, R. E. van Kesteren, P. Heutink, "High content screening in neurodegenerative diseases", *Journal of visualized experiments: JoVE*, no. 59, 2012.

[129] R. M. Jiang, D. Crookes, N. Luo, M. W. Davidson, "Live-cell tracking using SIFT features in DIC microscopic videos", *IEEE Transactions on Biomedical Engineering*, vol. 57, no. 9, pp. 2219–2228, 2010.

[130] X. Jiang, S. Shen, C. R. Cadwell, P. Berens, F. Sinz, A. S. Ecker, S. Patel, A. S. Tolias, "Principles of connectivity among morphologically defined cell types in adult neocortex", *Science*, vol. 350, no. 6264, p. aac9462, 2015.

[131] D. Jiménez, D. Labate, I. A. Kakadiaris, M. Papadakis, "Improved automatic centerline tracing for dendritic and axonal structures", *Neuroinformatics*, vol. 13, no. 2, pp. 227–244, 2015.

[132] E. R. Kandel, J. H. Schwartz, T. M. Jessell, S. A. Siegelbaum, A. J. Hudspeth, others, *Principles of Neural Science*, vol. 4, McGraw-hill New York, 2000.

[133] J. N. Kapur, P. K. Sahoo, A. K. Wong, "A new method for gray-level picture thresholding using the entropy of the histogram", *Computer Vision, Graphics, and Image Processing*, vol. 29, no. 3, pp. 273–285, 1985.

[134] D. P. Kingma & J. Ba, "Adam: A method for stochastic optimization", *arXiv preprint arXiv:1412.6980*, 2014.

[135] G. Kitagawa, "Monte Carlo filter and smoother for non-Gaussian nonlinear state space models", *Journal of Computational and Graphical Statistics*, vol. 5, no. 1, pp. 1–25, 1996.

[136] R. A. Koene, B. Tijms, P. van Hees, F. Postma, A. de Ridder, G. J. A. Ramakers, J. van Pelt, A. van Ooyen, "NETMORPH: a framework for the stochastic generation of large scale neuronal networks with realistic neuron morphologies", *Neuroinformatics*, vol. 7, no. 3, pp. 195–210, 2009.

[137] A. Kong, J. S. Liu, W. H. Wong, "Sequential imputations and Bayesian missing data problems", *Journal of the American Statistical Association*, vol. 89, no. 425, pp. 278–288, 1994.

[138] A. N. Koppes, A. M. Seggio, D. M. Thompson, "Neurite outgrowth is significantly increased by the simultaneous presentation of Schwann cells and moderate exogenous electric fields", *Journal of Neural Engineering*, vol. 8, no. 4, p. 046023, 2011.

[139] O. Z. Kraus & B. J. Frey, "Computer vision for high content screening", *Critical Reviews in Biochemistry and Molecular Biology*, vol. 51, no. 2, pp. 102–109, 2016.

[140] B. Krawczyk, "Learning from imbalanced data: Open challenges and future directions", *Progress in Artificial Intelligence*, vol. 5, no. 4, pp. 221–232, 2016.

[141] E. Kuminski, J. George, J. Wallin, L. Shamir, "Combining human and machine learning for morphological analysis of galaxy images", *Publications of the Astronomical Society of the Pacific*, vol. 126, no. 944, p. 959, 2014.

[142] R. Laganiere & R. Elias, "The detection of junction features in images", in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 3, IEEE, pp. 573–576, 2004.

[143] S. Lazebnik, C. Schmid, J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories", in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE Computer Society, pp. 2169–2178, 2006.

[144] J. J. Leandro, R. M. Cesar-Jr, L. d. F. Costa, "Automatic contour extraction from 2D neuron images", *Journal of Neuroscience Methods*, vol. 177, no. 2, pp. 497–509, 2009.

[145] Y. LeCun, Y. Bengio, G. Hinton, "Deep learning", *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[146] T.-C. Lee, R. L. Kashyap, C.-N. Chu, "Building skeleton models via 3-D medial surface axis thinning algorithms", *CVGIP: Graphical Models and Image Processing*, vol. 56, no. 6, pp. 462–478, 1994.

[147] J. P. Lewis, "Fast template matching", in *Vision Interface*, vol. 95, pp. 15–19, 1995.

[148] J. Li, S. Fong, R. K. Wong, V. W. Chu, "Adaptive multi-objective swarm fusion for imbalanced data classification", *Information Fusion*, vol. 39, pp. 1–24, 2018.

[149] R. Li, T. Zeng, H. Peng, S. Ji, "Deep learning segmentation of optical microscopy images improves 3-D neuron reconstruction", *IEEE Transactions on Medical Imaging*, vol. 36, no. 7, pp. 1533–1541, 2017.

[150] A. Liaw & M. Wiener, "Classification and regression by randomForest", *R News*, vol. 2, no. 3, pp. 18–22, 2002.

[151] Y.-C. Lin & A. J. Koleske, "Mechanisms of synapse and dendrite maintenance and their disruption in psychiatric and neurodegenerative disorders", *Annual Review of Neuroscience*, vol. 33, p. 349, 2010.

[152] G. Litjens, T. Kooi, B. E. Bejnordi, A. A. A. Setio, F. Ciompi, M. Ghafoorian, J. A. van der Laak, B. Van Ginneken, C. I. Sánchez, "A survey on deep learning in medical image analysis", *Medical Image Analysis*, vol. 42, pp. 60–88, 2017.

[153] S. Liu, D. Zhang, S. Liu, D. Feng, H. Peng, W. Cai, "Rivulet: 3D neuron morphology tracing with iterative back-tracking", *Neuroinformatics*, pp. 1–15, 2016.

[154] Y. Liu, "The DIADEM and beyond", Sep 2011.

[155] M. H. Longair, D. A. Baker, J. D. Armstrong, "Simple Neurite Tracer: open source software for reconstruction, visualization and analysis of neuronal processes", *Bioinformatics*, vol. 27, no. 17, pp. 2453–2454, 2011.

[156] D. G. Lowe, "Distinctive image features from scale-invariant keypoints", *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[157] J. Lu, J. C. Fiala, J. W. Lichtman, "Semi-automated reconstruction of neural processes from large numbers of fluorescence images", *PLoS One*, vol. 4, no. 5, p. e5655, 2009.

[158] Y. Lu, L. Carin, R. Coifman, W. Shain, B. Roysam, "Quantitative arbor analytics: unsupervised harmonic co-clustering of populations of brain cell arbors based on L-measure", *Neuroinformatics*, vol. 13, no. 1, pp. 47–63, 2015.

[159] J. Luisi, A. Narayanaswamy, Z. Galbreath, B. Roysam, "The FARSIGHT trace editor: an open source tool for 3-D inspection and efficient pattern analysis aided editing of automated neuronal reconstructions", *Neuroinformatics*, vol. 9, no. 2-3, pp. 305–315, 2011.

[160] G. Luo, D. Sui, K. Wang, J. Chae, "Neuron anatomy structure reconstruction based on a sliding filter", *BMC Bioinformatics*, vol. 16, no. 1, p. 342, 2015.

[161] J. MacQueen & others, "Some methods for classification and analysis of multivariate observations", in *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, vol. 1, Oakland, CA, USA, pp. 281–297, 1967.

[162] E. Maggio, M. Taj, A. Cavallaro, "Efficient multitarget visual tracking using random finite sets", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, no. 8, pp. 1016–1027, 2008.

[163] R. P. Mahler, "Multitarget Bayes filtering via first-order multitarget moments", *IEEE Transactions on Aerospace and Electronic Systems*, vol. 39, no. 4, pp. 1152–1178, 2003.

[164] R. P. Mahler, *Statistical Multisource-multitarget Information Fusion*, Artech House, 2007.

[165] H. Markram, E. Muller, S. Ramaswamy, M. W. Reimann, M. Abdellah, C. A. Sanchez, A. Ailamaki, L. Alonso-Nanclares, N. Antille, S. Arsever, others, "Reconstruction and simulation of neocortical microcircuitry", *Cell*, vol. 163, no. 2, pp. 456–492, 2015.

[166] I. Mason, "Initiation to end point: the multiple roles of fibroblast growth factors in neural development", *Nature Reviews Neuroscience*, vol. 8, no. 8, pp. 583–596, 2007.

[167] M. Masseroli, A. Bollea, G. Forloni, "Quantitative morphology and shape classification of neurons by computerized image analysis", *Computer Methods and Programs in Biomedicine*, vol. 41, no. 2, pp. 89–99, 1993.

[168] G. Mata, M. Radojević, I. Smal, M. Morales, E. Meijering, J. Rubio, "Automatic detection of neurons in high-content microscope images using machine learning approaches", in *IEEE International Symposium on Biomedical Imaging: From Nano to Macro (ISBI)*, IEEE, pp. 330–333, 2016.

[169] E. Meijering, "Neuron tracing in perspective", *Cytometry Part A*, vol. 77, no. 7, pp. 693–704, 2010.

[170] E. Meijering, A. E. Carpenter, H. Peng, F. A. Hamprecht, J.-C. Olivo-Marin, "Imagining the future of bioimage analysis", *Nature Biotechnology*, vol. 34, no. 12, pp. 1250–1255, 2016.

[171] E. Meijering, M. Jacob, J.-C. Sarria, P. Steiner, H. Hirling, M. Unser, "Design and validation of a tool for neurite tracing and analysis in fluorescence microscopy images", *Cytometry Part A*, vol. 58, no. 2, pp. 167–176, 2004.

[172] J. M. Mendel, "Fuzzy logic systems for engineering: a tutorial", *Proceedings of the IEEE*, vol. 83, no. 3, pp. 345–377, 1995.

[173] D. Meyer, E. Dimitriadou, K. Hornik, A. Weingessel, F. Leisch, "e1071: Misc functions of the department of statistics, probability theory group (formerly: E1071), TU Wien", *Comprehensive R Archive Network (CRAN)*, 2017.

[174] M. Michaelis & G. Sommer, "Junction classification by multiple orientation detection", in *European Conference on Computer Vision (ECCV)*, Springer, pp. 101–108, 1994.

[175] X. Ming, A. Li, J. Wu, C. Yan, W. Ding, H. Gong, S. Zeng, Q. Liu, "Rapid reconstruction of 3D neuronal morphology from light microscopy images with augmented rayburst sampling", *PloS One*, vol. 8, no. 12, pp. 1–10, 12 2014.

[176] F. Mualla, S. Schöll, B. Sommerfeldt, A. Maier, J. Hornegger, "Automatic cell detection in bright-field microscope images using SIFT, random forests, and hierarchical clustering", *IEEE Transactions on Medical Imaging*, vol. 32, no. 12, pp. 2274–2286, 2013.

[177] A. Mukherjee & A. Stepanyants, "Automated reconstruction of neural trees using front re-initialization", in *Proceedings of SPIE Medical Imaging: Image Processing*, vol. 8314, International Society for Optics and Photonics, p. 83141I, 2012.

[178] S. Mukherjee & S. T. Acton, "Vector field convolution medialness applied to neuron tracing", in *IEEE International Conference on Image Processing (ICIP)*, IEEE, pp. 665–669, 2013.

[179] S. Mukherjee, B. Condron, S. T. Acton, "Tubularity flow field–a technique for automatic neuron segmentation", *IEEE Transactions on Image Processing*, vol. 24, no. 1, pp. 374–389, 2015.

[180] D. Myatt, T. Hadlington, G. Ascoli, S. Nasuto, "Neuromantic–from semi-manual to semi-automatic reconstruction of neuron morphology", *Frontiers in Neuroinformatics*, vol. 6, p. 4, 2012.

[181] A. Narayanaswamy, Y. Wang, B. Roysam, "3-D image pre-processing algorithms for improved automated tracing of neuronal arbors", *Neuroinformatics*, vol. 9, no. 2-3, pp. 219–231, 2011.

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

[183] D. Ni, Y. P. Chui, Y. Qu, X. Yang, J. Qin, T.-T. Wong, S. S. Ho, P. A. Heng, "Reconstruction of volumetric ultrasound panorama based on improved 3D SIFT", *Computerized Medical Imaging and Graphics*, vol. 33, no. 7, pp. 559–566, 2009.

[184] B. Obara, M. Fricker, D. Gavaghan, V. Grau, "Contrast-independent curvilinear structure detection in biomedical images", *IEEE Transactions on Image Processing*, vol. 21, no. 5, pp. 2572–2581, 2012.

[185] B. Obara, M. Frickerc, V. Graua, "Contrast independent detection of branching points in network-like structures", in *SPIE Medical Imaging*, International Society for Optics and Photonics, pp. 83141L–83141L, 2012.

[186] N. Orlov, L. Shamir, T. Macura, J. Johnston, D. M. Eckley, I. G. Goldberg, "WND-CHARM: Multi-purpose image classification using compound image transforms", *Pattern Recognition Letters*, vol. 29, no. 11, pp. 1684–1693, 2008.

[187] N. Otsu, "A threshold selection method from gray-level histograms", *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.

[188] R. Parekh & G. A. Ascoli, "Neuronal morphology goes digital: a research hub for cellular and system neuroscience", *Neuron*, vol. 77, no. 6, pp. 1017–1038, 2013.

[189] H. Peng, A. Bria, Z. Zhou, G. Iannello, F. Long, "Extensible visualization and analysis for multidimensional images using Vaa3D", *Nature Protocols*, vol. 9, no. 1, pp. 193–208, 2014.

[190] H. Peng, M. Hawrylycz, J. Roskams, S. Hill, N. Spruston, E. Meijering, G. A. Ascoli, "BigNeuron: large-scale 3D neuron reconstruction from optical microscopy images", *Neuron*, vol. 87, no. 2, pp. 252–256, 2015.

[191] H. Peng, F. Long, X. Liu, S. K. Kim, E. W. Myers, "Straightening Caenorhabditis elegans images", *Bioinformatics*, vol. 24, no. 2, pp. 234–242, 2007.

[192] H. Peng, F. Long, G. Myers, "Automatic 3D neuron tracing using all-path pruning", *Bioinformatics*, vol. 27, no. 13, pp. i239–i247, 2011.

[193] H. Peng, F. Long, T. Zhao, E. Myers, "Proof-editing is the bottleneck of 3D neuron reconstruction: the problem and solutions", *Neuroinformatics*, vol. 9, no. 2-3, pp. 103–105, 2011.

[194] H. Peng, E. Meijering, G. A. Ascoli, "From DIADEM to BigNeuron", *Neuroinformatics*, pp. 1–2, 2015.

[195] H. Peng, Z. Ruan, D. Atasoy, S. Sternson, "Automatic reconstruction of 3D neuron structures using a graph-augmented deformable model", *Bioinformatics*, vol. 26, no. 12, pp. i38–i46, 2010.

[196] H. Peng, Z. Ruan, F. Long, J. H. Simpson, E. W. Myers, "V3D enables real-time 3D visualization and quantitative analysis of large-scale biological image data sets", *Nature Biotechnology*, vol. 28, no. 4, pp. 348–353, 2010.

[197] H. Peng, J. Tang, H. Xiao, A. Bria, J. Zhou, V. Butler, Z. Zhou, P. T. Gonzalez-Bellido, S. W. Oh, J. Chen, others, "Virtual finger boosts three-dimensional imaging and microsurgery as well as terabyte volume image visualization and analysis", *Nature Communications*, vol. 5, p. 4342, 2014.

[198] H. Peng, Z. Zhou, E. Meijering, T. Zhao, G. A. Ascoli, M. Hawrylycz, "Automatic tracing of ultra-volumes of neuronal images", *Nature Methods*, vol. 14, no. 4, pp. 332–333, 2017.

[199] S. Polavaram, T. A. Gillette, R. Parekh, G. A. Ascoli, "Statistical analysis and data mining of digital reconstructions of dendritic morphologies", *Frontiers in Neuroanatomy*, vol. 8, p. 138, 2014.

[200] M. Pool, J. Thiemann, A. Bar-Or, A. E. Fournier, "NeuriteTracer: a novel ImageJ plugin for automated quantification of neurite outgrowth", *Journal of Neuroscience Methods*, vol. 168, no. 1, pp. 134–139, 2008.

[201] D. M. Powers, "Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation", *Journal of Machine Learning Technologies*, vol. 2, no. 1, pp. 37–63, 2011.

[202] J. M. Prewitt, "Object enhancement and extraction", *Picture Processing and Psychopictorics*, vol. 10, no. 1, pp. 15–19, 1970.

[203] Z. Püspöki, C. Vonesch, M. Unser, "Detection of symmetric junctions in biological images using 2-D steerable wavelet transforms", in *IEEE International Symposium on Biomedical Imaging: From Nano to Macro (ISBI)*, IEEE, pp. 1496–1499, 2013.

[204] T. Quan, T. Zheng, Z. Yang, W. Ding, S. Li, J. Li, H. Zhou, Q. Luo, H. Gong, S. Zeng, "NeuroGPS: automated localization of neurons for brain circuits using L1 minimization model", *Scientific Reports*, vol. 3, p. 1414, 2013.

[205] T. Quan, H. Zhou, J. Li, S. Li, A. Li, Y. Li, X. Lv, Q. Luo, H. Gong, S. Zeng, "NeuroGPS-Tree: automatic reconstruction of large-scale neuronal populations with dense neurites", *Nature Methods*, vol. 13, no. 1, p. 51, 2016.

[206] R Core Team, *R: A language and environment for statistical computing*, R Foundation for Statistical Computing, Vienna, Austria, 2015.

[207] N. M. Radio, "Neurite outgrowth assessment using high content analysis methodology", in *Neurotrophic Factors*, Springer, pp. 247–260, 2012.

[208] M. Radojević & E. Meijering, "Automated neuron tracing using probability hypothesis density filtering", *Bioinformatics*, vol. 33, no. 7, pp. 1073–1080, 2017.

[209] M. Radojević & E. Meijering, "Neuron reconstruction from fluorescence microscopy images using sequential Monte Carlo estimation", in *IEEE International Symposium on Biomedical Imaging: From Nano to Macro (ISBI)*, IEEE, pp. 36–39, 2017.

[210] M. Radojević & E. Meijering, "Automated neuron reconstruction from 3D fluorescence microscopy images using sequential Monte Carlo estimation", *Neuroinformatics*, in press.

[211] M. Radojević, I. Smal, E. Meijering, "Automated neuron morphology reconstruction using fuzzy-logic detection and Bayesian tracing algorithms", in *IEEE International Symposium on Biomedical Imaging: From Nano to Macro (ISBI)*, IEEE, pp. 885–888, 2015.

[212] M. Radojević, I. Smal, E. Meijering, "Fuzzy-logic based detection and characterization of junctions and terminations in fluorescence microscopy images of neurons", *Neuroinformatics*, vol. 14, no. 2, pp. 201–219, 2016.

[213] M. Radojević, I. Smal, W. Niessen, E. Meijering, "Fuzzy logic based detection of neuron bifurcations in microscopy images", in *IEEE International Symposium on Biomedical Imaging: From Nano to Macro (ISBI)*, IEEE, pp. 1307–1310, April 2014.

[214] S. Ramón & others, *Histología del Sistema Nervioso del Hombre y de los Vertebrados*, vol. 1, Editorial CSIC-CSIC Press, 2008.

[215] B. Ristic, S. Arulampalam, N. J. Gordon, *Beyond the Kalman filter: Particle Filters for Tracking Applications*, Artech House, 2004.

[216] B. Ristic, D. Clark, B.-N. Vo, "Improved SMC implementation of the PHD filter", in *13th International Conference on Information Fusion (FUSION)*, IEEE, pp. 1–8, 2010.

[217] A. Rodriguez, D. B. Ehlenberger, P. R. Hof, S. L. Wearne, "Three-dimensional neuron tracing by voxel scooping", *Journal of Neuroscience Methods*, vol. 184, no. 1, pp. 169–175, 2009.

[218] Y. Saeys, I. Inza, P. Larrañaga, "A review of feature selection techniques in bioinformatics", *Bioinformatics*, vol. 23, no. 19, pp. 2507–2517, 2007.

[219] J. A. Sáez, J. Luengo, J. Stefanowski, F. Herrera, "SMOTE–IPF: Addressing the noisy and borderline examples problem in imbalanced classification by a re-sampling method with filtering", *Information Sciences*, vol. 291, pp. 184–203, 2015.

[220] A. V. Samsonovich & G. A. Ascoli, "Statistical determinants of dendritic morphology in hippocampal pyramidal neurons: a hidden markov model", *Hippocampus*, vol. 15, no. 2, pp. 166–183, 2005.

[221] R. J. Samworth & others, "Optimal weighted nearest neighbour classifiers", *Annals of Statistics*, vol. 40, no. 5, pp. 2733–2763, 2012.

[222] A. Santamaría-Pang, P. Hernandez-Herrera, M. Papadakis, P. Saggau, I. A. Kakadiaris, "Automatic morphological reconstruction of neurons from multiphoton and confocal microscopy images using 3D tubular models", *Neuroinformatics*, vol. 13, no. 3, pp. 297–320, 2015.

[223] S. Särkkä, *Bayesian Filtering and Smoothing*, vol. 3, Cambridge University Press, 2013.

[224] Y. Sato, S. Nakajima, H. Atsumi, T. Koller, G. Gerig, S. Yoshida, R. Kikinis, "3D multi-scale line filter for segmentation and visualization of curvilinear structures in medical images", in *Proceedings of the First Joint Conference on Computer Vision, Virtual Reality and Robotics in Medicine and Medial Robotics and Computer-Assisted Surgery (CVRMed-MRCAS)*, Springer, pp. 213–222, 1997.

[225] Y. Sato, S. Nakajima, N. Shiraga, H. Atsumi, S. Yoshida, T. Koller, G. Gerig, R. Kikinis, "Three-dimensional multi-scale line filter for segmentation and visualization of curvilinear structures in medical images", *Medical Image Analysis*, vol. 2, no. 2, pp. 143–168, 1998.

[226] I. Schlangen, J. Franco, J. Houssineau, W. T. Pitkeathly, D. Clark, I. Smal, C. Rickman, "Marker-less stage drift correction in super-resolution microscopy using the single-cluster PHD filter", *IEEE Journal of Selected Topics in Signal Processing*, vol. 10, no. 1, pp. 193–202, 2016.

[227] K. Schliep & K. Hechenbichler, *kknn: Weighted k-Nearest Neighbors*, 2016.

[228] S. Schmitt, J. F. Evers, C. Duch, M. Scholz, K. Obermayer, "New methods for the computer-assisted 3-D reconstruction of neurons from confocal image stacks", *Neuroimage*, vol. 23, no. 4, pp. 1283–1298, 2004.

[229] C. A. Schneider, W. S. Rasband, K. W. Eliceiri, "NIH Image to ImageJ: 25 years of image analysis", *Nature Methods*, vol. 9, no. 7, pp. 671–675, 2012.

[230] R. Scorcioni, M. T. Lazarewicz, G. A. Ascoli, "Quantitative morphometry of hippocampal pyramidal cells: differences between anatomical classes and reconstructing laboratories", *Journal of Comparative Neurology*, vol. 473, no. 2, pp. 177–193, 2004.

[231] R. Scorcioni, S. Polavaram, G. A. Ascoli, "L-Measure: a web-accessible tool for the analysis, comparison and search of digital reconstructions of neuronal morphologies", *Nature Protocols*, vol. 3, no. 5, pp. 866–876, 2008.

[232] S. L. Senft, "A brief history of neuronal reconstruction", *Neuroinformatics*, vol. 9, no. 2-3, pp. 119–128, 2011.

[233] J. A. Sethian, *Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materials Science*, vol. 3, Cambridge University Press, 1999.

[234] T. Shaikhina & N. A. Khovanova, "Handling limited datasets with neural networks in medical applications: A small-data approach", *Artificial Intelligence in Medicine*, vol. 75, pp. 51–63, 2017.

[235] L. Shamir, "Automatic detection of peculiar galaxies in large datasets of galaxy images", *Journal of Computational Science*, vol. 3, no. 3, pp. 181–189, 2012.

[236] L. Shamir, J. D. Delaney, N. Orlov, D. M. Eckley, I. G. Goldberg, "Pattern recognition software and techniques for biological image analysis", *PLoS Computational Biology*, vol. 6, no. 11, p. e1000974, 2010.

[237] L. Shamir, N. Orlov, D. M. Eckley, T. Macura, J. Johnston, I. G. Goldberg, "Wndchrm–an open source utility for biological image analysis", *Source Code for Biology and Medicine*, vol. 3, no. 1, p. 13, 2008.

[238] L. Shamir & J. A. Tarakhovsky, "Computer analysis of art", *Journal on Computing and Cultural Heritage (JOCCH)*, vol. 5, no. 2, pp. 7:1–7:11, 2012.

[239] S. S. Shapiro & M. B. Wilk, "An analysis of variance test for normality (complete samples)", *Biometrika*, vol. 52, no. 3/4, pp. 591–611, 1965.

[240] D. Shen, G. Wu, H.-I. Suk, "Deep learning in medical image analysis", *Annual Review of Biomedical Engineering*, vol. 19, pp. 221–248, 2017.

[241] C. J. Sheppard, X. Gan, M. Gu, M. Roy, "Signal-to-noise ratio in confocal microscopes", in *Handbook of Biological Confocal Microscopy*, Springer, pp. 442–452, 2006.

[242] D. A. Sholl, "Dendritic organization in the neurons of the visual and motor cortices of the cat", *Journal of Anatomy*, vol. 87, no. Pt 4, p. 387, 1953.

[243] R. Simon, "Resampling strategies for model assessment and selection", in *Fundamentals of Data Mining in Genomics and Proteomics*, Springer, pp. 173–186, 2007.

[244] K. Simonyan & A. Zisserman, "Very deep convolutional networks for large-scale image recognition", *arXiv preprint arXiv:1409.1556*, 2014.

[245] S. Singh, A. E. Carpenter, A. Genovesio, "Increasing the content of high-content screening: an overview", *Journal of Biomolecular Screening*, vol. 19, no. 5, pp. 640–650, 2014.

[246] E. D. Sinzinger, "A model-based approach to junction detection using radial energy", *Pattern Recognition*, vol. 41, no. 2, pp. 494–505, 2008.

[247] A. Sironi, E. Türetken, V. Lepetit, P. Fua, "Multiscale centerline detection", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 7, pp. 1327–1341, 2016.

[248] Z. Šišková, D. Justus, H. Kaneko, D. Friedrichs, N. Henneberg, T. Beutel, J. Pitsch, S. Schoch, A. Becker, H. von der Kammer, others, "Dendritic structural degeneration is functionally linked to cellular hyperexcitability in a mouse model of Alzheimer's disease", *Neuron*, vol. 84, no. 5, pp. 1023–1033, 2014.

[249] T. Smafield, V. Pasupuleti, K. Sharma, R. L. Huganir, B. Ye, J. Zhou, "Automatic dendritic length quantification for high throughput screening of mature neurons", *Neuroinformatics*, vol. 13, no. 4, pp. 443–458, 2015.

[250] I. Smal, M. Loog, W. Niessen, E. Meijering, "Quantitative comparison of spot detection methods in fluorescence microscopy", *IEEE Transactions on Medical Imaging*, vol. 29, no. 2, pp. 282–301, 2010.

[251] C. Sommer & D. W. Gerlich, "Machine learning in cell biology – teaching computers to recognize phenotypes", *Journal of Cell Science*, vol. 126, no. 24, pp. 5529–5539, 2013.

[252] M. Sonka, V. Hlavac, R. Boyle, *Image Processing, Analysis, and Machine Vision*, Cengage Learning, 2014.

[253] L. R. Squire, "Memory and the hippocampus: A synthesis from findings with rats, monkeys, and humans", *Psychological Review*, vol. 99, no. 2, pp. 195–231, 1992.

[254] C. Steger, "An unbiased detector of curvilinear structures", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 2, pp. 113–125, 1998.

[255] P. Steiner, J.-C. Sarria, B. Huni, R. Marsault, S. Catsicas, H. Hirling, "Overexpression of neuronal Sec1 enhances axonal branching in hippocampal neurons", *Neuroscience*, vol. 113, no. 4, pp. 893–905, 2002.

[256] E. Stockley, H. Cole, A. Brown, H. Wheal, "A system for quantitative morphological measurement and electrotonic modelling of neurons: three-dimensional reconstruction", *Journal of Neuroscience Methods*, vol. 47, no. 1, pp. 39–51, 1993.

[257] L. D. Stone, R. L. Streit, T. L. Corwin, K. L. Bell, *Bayesian Multiple Target Tracking*, Artech House, 2013.

[258] C. Strobl, T. Hothorn, A. Zeileis, "Party on! a new, conditional variable importance measure for random forests available in the party package", *The R Journal*, vol. 1, pp. 14–17, 12 2009.

[259] R. Su, C. Sun, T. D. Pham, "Junction detection for linear structures based on Hessian, correlation and shape information", *Pattern Recognition*, vol. 45, no. 10, pp. 3695–3706, 2012.

[260] K. Svoboda, "The past, present, and future of single neuron reconstruction", *Neuroinformatics*, vol. 9, no. 2-3, p. 97, 2011.

[261] L. W. Swanson, E. Newman, A. Araque, J. M. Dubinsky, *Beautiful Brain: The Drawings of Santiago Ramon y Cajal*, Harry N. Abrams, 2017.

[262] N. Tajbakhsh, J. Y. Shin, S. R. Gurudu, R. T. Hurst, C. B. Kendall, M. B. Gotway, J. Liang, "Convolutional neural networks for medical image analysis: Full training or fine tuning?", *IEEE Transactions on Medical Imaging*, vol. 35, no. 5, pp. 1299–1312, 2016.

[263] H. Tamura, S. Mori, T. Yamawaki, "Textural features corresponding to visual perception", *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 8, no. 6, pp. 460–473, 1978.

[264] R. Tibshirani, "Regression shrinkage and selection via the lasso", *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 267–288, 1996.

[265] M. Tobias & A. D. Lanterman, "Probability hypothesis density-based multitarget tracking with bistatic range and Doppler observations", *IEE Proceedings-Radar, Sonar and Navigation*, vol. 152, no. 3, pp. 195–205, 2005.

[266] C.-L. Tsai, C. V. Stewart, H. L. Tanenbaum, B. Roysam, "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*, vol. 8, no. 2, pp. 122–130, 2004.

[267] E. Turetken, F. Benmansour, B. Andres, H. Pfister, P. Fua, "Reconstructing loopy curvilinear structures using integer programming", in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, pp. 1822–1829, 2013.

[268] E. Turetken, F. Benmansour, P. Fua, "Automated reconstruction of tree structures using path classifiers and mixed integer programming", in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, pp. 566–573, 2012.

[269] E. Türetken, G. González, C. Blum, P. Fua, "Automated reconstruction of dendritic and axonal trees by global optimization with geometric priors", *Neuroinformatics*, vol. 9, no. 2-3, pp. 279–302, 2011.

[270] V. Uhlmann, S. Singh, A. E. Carpenter, "CP-CHARM: segmentation-free image classification made accessible", *BMC Bioinformatics*, vol. 17, no. 1, p. 51, 2016.

[271] P. Vallotton, R. Lagerstrom, C. Sun, M. Buckley, D. Wang, M. De Silva, S.-S. Tan, J. M. Gunnersen, "Automated analysis of neurite branching in cultured cortical neurons using HCA-Vision", *Cytometry Part A: The Journal of the International Society for Analytical Cytology*, vol. 71, no. 10, pp. 889–895, 2007.

[272] J. van Pelt, A. van Ooyen, H. B. Uylings, "The need for integrating neuronal morphology databases and computational environments in exploring neuronal structure and function", *Anatomy and Embryology*, vol. 204, no. 4, pp. 255–265, 2001.

[273] R. Van Uitert & I. Bitter, "Subvoxel precise skeletons of volumetric data based on fast marching methods", *Medical Physics*, vol. 34, no. 2, pp. 627–638, 2007.

[274] L. J. van Vliet, D. Sudar, I. T. Young, "Digital fluorescence imaging using cooled CCD array cameras invisible", *JE Celis (eds); Cell Biol. Second Eddition*, vol. 3, pp. 109–120, 1998.

[275] V. Vapnik, *Statistical Learning Theory*, vol. 3, Wiley, New York, 1998.

[276] V. Vapnik, *The Nature of Statistical Learning Theory*, Springer Science and Business Media, 2013.

[277] Z. Vasilkoski & A. Stepanyants, "Detection of the optimal neuron traces in confocal microscopy images", *Journal of Neuroscience Methods*, vol. 178, no. 1, pp. 197–204, 2009.

[278] A. Vedaldi & B. Fulkerson, "VLFeat: An open and portable library of computer vision algorithms", in *Proceedings of the 18th ACM International Conference on Multimedia*, ACM, pp. 1469–1472, 2010.

[279] J.-P. Vert, K. Tsuda, B. Schölkopf, "A primer on kernel methods", *Kernel Methods in Computational Biology*, vol. 47, pp. 35–70, 2004.

[280] B.-N. Vo & W.-K. Ma, "The Gaussian mixture probability hypothesis density filter", *IEEE Transactions on Signal Processing*, vol. 54, no. 11, pp. 4091–4104, 2006.

[281] B.-N. Vo, S. Singh, A. Doucet, "Sequential Monte Carlo methods for multitarget filtering with random finite sets", *IEEE Transactions on Aerospace and Electronic Systems*, vol. 41, no. 4, pp. 1224–1245, 2005.

[282] Y. Wan, F. Long, L. Qu, H. Xiao, M. Hawrylycz, E. W. Myers, H. Peng, "BlastNeuron for automated comparison, retrieval and clustering of 3D neuron morphologies", *Neuroinformatics*, vol. 13, no. 4, pp. 487–499, 2015.

[283] Y. Wang, A. Narayanaswamy, C.-L. Tsai, B. Roysam, "A broadly applicable 3-D neuron tracing method based on open-curve snake", *Neuroinformatics*, vol. 9, no. 2-3, pp. 193–217, 2011.

[284] Y.-D. Wang, J.-K. Wu, A. A. Kassim, W. Huang, "Data-driven probability hypothesis density filter for visual tracking", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, no. 8, pp. 1085–1095, 2008.

[285] S. Wearne, A. Rodriguez, D. Ehlenberger, A. Rocher, S. Henderson, P. Hof, "New techniques for imaging, digitization and analysis of three-dimensional neural morphology on multiple scales", *Neuroscience*, vol. 136, no. 3, pp. 661–680, 2005.

[286] C. M. Weaver, P. R. Hof, S. L. Wearne, W. B. Lindquist, "Automated algorithms for multiscale morphometry of neuronal dendrites", *Neural Computation*, vol. 16, no. 7, pp. 1353–1383, 2004.

[287] H. Wickham, *ggplot2: Elegant Graphics for Data Analysis*, Springer-Verlag New York, 2016.

[288] T. M. Wood, C. A. Yates, D. A. Wilkinson, G. Rosser, "Simplified multitarget tracking using the PHD filter for microscopic video data", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 5, pp. 702–713, 2012.

[289] C. Wu, J. Schulte, K. J. Sepp, J. T. Littleton, P. Hong, "Automatic robust neurite detection and morphological analysis of neuronal cell cultures in high-content screening", *Neuroinformatics*, vol. 8, no. 2, pp. 83–100, 2010.

[290] X. Xia & S. T. Wong, "Concise review: A high-content screening approach to stem cell research and drug discovery", *Stem Cells*, vol. 30, no. 9, pp. 1800–1807, 2012.

[291] H. Xiao & H. Peng, "APP2: automatic tracing of 3D neuron morphology based on hierarchical pruning of a gray-weighted image distance-tree", *Bioinformatics*, vol. 29, no. 11, pp. 1448–1454, 2013.

[292] J. Xie, T. Zhao, T. Lee, E. Myers, H. Peng, "Automatic neuron tracing in volumetric microscopy images with anisotropic path searching", in *International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, Springer, pp. 472–479, 2010.

[293] G. Xiong, X. Zhou, A. Degterev, L. Ji, S. T. Wong, "Automated neurite labeling and analysis in fluorescence microscopy images", *Cytometry Part A*, vol. 69, no. 6, pp. 494–505, 2006.

[294] C. Yan, A. Li, B. Zhang, W. Ding, Q. Luo, H. Gong, "Automated and accurate detection of soma location and surface morphology in large-scale 3D neuron images", *PloS One*, vol. 8, no. 4, p. e62579, 2013.

[295] J. Yang, P. T. Gonzalez-Bellido, H. Peng, "A distance-field based automatic neuron tracing method", *BMC Bioinformatics*, vol. 14, no. 1, p. 93, 2013.

[296] J. Yang, K. Yu, Y. Gong, T. Huang, "Linear spatial pyramid matching using sparse coding for image classification", in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, pp. 1794–1801, 2009.

[297] J.-C. Yoo & T. H. Han, "Fast normalized cross-correlation", *Circuits, Systems and Signal Processing*, vol. 28, no. 6, p. 819, 2009.

[298] D. Yu, F. Yang, C. Yang, C. Leng, J. Cao, Y. Wang, J. Tian, "Fast rotation-free feature-based image registration using improved N-SIFT and GMM-based parallel optimization", *IEEE Transactions on Biomedical Engineering*, vol. 63, no. 8, pp. 1653–1664, 2016.

[299] W. Yu, K. Daniilidia, G. Sommer, "Rotated wedge averaging method for junction characterization", in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, pp. 390–395, 1998.

[300] X. Yuan, J. T. Trachtenberg, S. M. Potter, B. Roysam, "MDL constrained 3-D grayscale skeletonization algorithm for automated extraction of dendrites and spines from fluorescence confocal images", *Neuroinformatics*, vol. 7, no. 4, p. 213, 2009.

[301] L. A. Zadeh, "Fuzzy logic and approximate reasoning", *Synthese*, vol. 30, no. 3-4, pp. 407–428, 1975.

[302] T. Zajic & R. P. Mahler, "Particle-systems implementation of the PHD multitarget-tracking filter", in *Signal Processing, Sensor Fusion, and Target Recognition XII*, vol. 5096, International Society for Optics and Photonics, pp. 291–300, 2003.

[303] B. Zhang, J. Zerubia, J.-C. Olivo-Marin, "Gaussian approximations of fluorescence microscope point-spread function models", *Applied Optics*, vol. 46, no. 10, pp. 1819–1829, 2007.

[304] R. Zhang, W. Zhou, Y. Li, S. Yu, Y. Xie, "Nonrigid registration of lung CT images based on tissue features", *Computational and Mathematical Methods in Medicine*, vol. 2013, 2013. Article ID 834192, 7 pages.

[305] Y. Zhang, X. Zhou, A. Degterev, M. Lipinski, D. Adjeroh, J. Yuan, S. T. Wong, "Automated neurite extraction using dynamic programming for high-throughput screening of neuron-based assays", *NeuroImage*, vol. 35, no. 4, pp. 1502–1515, 2007.

[306] Y. Zhang, X. Zhou, A. Degterev, M. Lipinski, D. Adjeroh, J. Yuan, S. T. Wong, "A novel tracing algorithm for high throughput imaging: Screening of neuron-based assays", *Journal of Neuroscience Methods*, vol. 160, no. 1, pp. 149–162, 2007.

[307] T. Zhao, J. Xie, F. Amat, N. Clack, P. Ahammad, H. Peng, F. Long, E. Myers, "Automated reconstruction of neuronal morphology based on local geometrical and global structural models", *Neuroinformatics*, vol. 9, no. 2-3, pp. 247–261, 2011.

[308] J. Zhou, S. Chang, D. Metaxas, L. Axel, "Vascular structure segmentation and bifurcation detection", in *IEEE International Symposium on Biomedical Imaging: From Nano to Macro (ISBI)*, IEEE, pp. 872–875, 2007.

[309] Z. Zhou, X. Liu, B. Long, H. Peng, "TReMAP: automatic 3D neuron reconstruction based on tracing, reverse mapping and assembling of 2D projections", *Neuroinformatics*, vol. 14, no. 1, pp. 41–50, 2016.

[310] Z. Zhou, S. Sorensen, H. Zeng, M. Hawrylycz, H. Peng, "Adaptive image enhancement for tracing 3D morphologies of neurons and brain vasculatures", *Neuroinformatics*, vol. 13, no. 2, pp. 153–166, 2015.

[311] Z. Zhou, S. A. Sorensen, H. Peng, "Neuron crawler: an automatic tracing algorithm for very large neuron images", in *IEEE International Symposium on Biomedical Imaging: From Nano to Macro (ISBI)*, IEEE, pp. 870–874, 2015.

Samenvatting

Neuronen behoren tot de belangrijkste elementen van het zenuwstelsel. Fascinatie voor deze cellen gaat minstens terug tot het baanbrekende werk van Ramón y Cajal, nu meer dan een eeuw geleden. Gewapend met een microscoop en gebruikmakend van zilverkleuring, niet lang daarvoor ontdekt door Golgi, bestudeerde hij hersenweefsels uit een groot aantal gebieden van de hersenen. Zijn bevindingen leidden tot het opstellen van de neuronentheorie, die stelt dat het zenuwstelsel, net als alle andere organen in het lichaam, is opgebouwd uit afzonderlijke cellen. Golgi en Ramón y Cajal deelden in 1906 de Nobelprijs voor Geneeskunde. Daarop volgend grondig onderzoek naar neuronen onthulde dat deze cellen de bijzondere eigenschap hebben dat ze signalen kunnen ontvangen en doorgeven. Daarmee regelen ze een groot aantal lichaamsfuncties. Ook werd duidelijk dat neuronen in de verschillende delen van de hersenen verschillende functies hebben. Afhankelijk van hun specifieke rol in het zenuwstelsel kunnen neuronen nogal variren in hun morfologische eigenschappen.

Morfologische analyse van de verschillende soorten neuronen is daarom vaak een belangrijk onderdeel in het onderzoek naar hun functie. Neuronen kunnen tegenwoordig in groot detail en digitaal worden afgebeeld door middel van moderne lichtmicroscopen. Maar om de eigenschappen van een gegeven neuron daadwerkelijk te kunnen kwantificeren is een explicietere representatie van zijn morfologie nodig dan een microscoopbeeld. Het afleiden van een grafische representatie van een neuron uit zijn microscoopbeeld, in de vorm van een boomstructuur bestaande uit knooppunten en vertakkingen, wordt doorgaans aangeduid als digitale reconstructie, en is het hoofdthema van dit proefschrift. Veel neurowetenschappelijke studies zijn afhankelijk van een nauwkeurige beschrijving van de morfologie van neuronen in de vorm van digitale reconstructies. Daarmee is digitale reconstructie een belangrijk technisch probleem in neurowetenschappelijk onderzoek.

Dit proefschrift presenteert nieuwe computationele methoden voor de automatische analyse van neuronen. De hoofdproblemen waarvoor oplossingen worden gepresenteerd zijn de detectie en de reconstructie van neuronen in digitale fluorescentiemicroscopiebeelden. Een van de belangrijkste vernieuwingen die worden voorgesteld ten opzichte van bestaande reconstructiemethoden is het gebruik van probabilistische filtertechnieken. Na een algemene inleiding in het eerste hoofdstuk, beschrijven de daarop volgende hoofdstukken originele oplossingen voor de automatische detectie van knooppunten en eindpunten van neuronen in hun afbeeldingen, het traceren van alle vertakkingen van de neuronen in de beelden, en het vinden van neuronen in lageresolutiebeelden uit screeningstudies. De rest van deze samenvatting geeft kort de

inhoud van de hoofdstukken weer.

Het tweede hoofdstuk presenteert een nieuwe methode voor de automatische detectie van die punten in beelden van neuronen die van cruciaal belang zijn voor de correcte topologische representatie van de neuronen. Het gaat hierbij vooral om de knooppunten en de eindpunten van alle vertakkingen. Een knooppunt is een punt waar drie segmenten van de boomstructuur bij elkaar komen, en een eindpunt is een punt waar een vertakking van de boomstructuur eindigt. De voorgestelde detectiemethode maakt gebruik van richtingsfilters, waarmee in elk punt van een beeld wordt bepaald in hoeverre en in welke richting(en) er lijnachtige structuren door het punt lopen. De gevonden informatie hierover wordt uitgedrukt in taalkundige termen die vervolgens verwerkt worden door middel van zogeheten vage logica. Daarmee wordt elk punt met behulp van een stelsel van regels geclassificeerd als zijnde irrelevant of een specifiek type cruciaal punt uit de boomstructuur van het neuron. Voor dit doel wordt een nieuw stelsel van regels en klassen voorgesteld. De kracht van de gekozen aanpak is dat voor elk punt berekend wordt in welke mate het behoort tot elk van de beschouwde klassen. Daardoor wordt rekening gehouden met de onzekerheid in de beeldinformatie en het filterproces.

In het derde hoofdstuk wordt het theorema van Bayes benut om te komen tot een nieuwe automatische methode voor het traceren van de middellijn van neuronale vertakkingen in microscoopbeelden. In tegenstelling tot bestaande methoden is de voorgestelde methode probabilistisch van aard en in staat om tegelijkertijd een onbeperkt aantal vertakkingen te traceren. Voor de implementatie van de methode wordt gebruik gemaakt van sequentile Monte Carlo filtering. Een dergelijke aanpak wordt ook wel gebruikt in andere toepassingen, voor het volgen van bewegende objecten over de tijd in filmopnames. In dit hoofdstuk wordt het idee echter aangewend voor het volgen van objecten in de ruimte in statische opnames. Om dit mogelijk te maken worden nieuwe wiskundige modellen voorgesteld voor het filterproces, waarin bestaande kennis is opgenomen over de vorm van neuronale vertakkingen en de manier waarop ze worden afgebeeld door een microscoop. De probabilistische aard van de methode maakt dat herhaalde toepassing op hetzelfde beeld net iets andere resultaten oplevert. Op deze manier kan meer statistisch bewijsmateriaal worden vergaard over de vorm van de vertakkingen dan dat deterministische methoden kunnen leveren. De gepresenteerde experimentele resultaten bevestigen inderdaad dat de methode nauwkeuriger is.

Het vierde hoofdstuk tilt het idee van probabilistische tracing nog een stap verder en presenteert een nieuwe methode voor automatische volledige reconstructie van neuronen uit microscoopbeelden. Deze methode vindt niet alleen de middellijn van individuele vertakkingen, maar maakt ook een schatting van de lokale diameter op elk punt van de vertakkingen, en voegt alle gevonden segmenten samen tot een datastructuur die de berekening van allerlei morfologische eigenschappen mogelijk maakt. De methode begint met het identificeren van die gebieden in een beeld die hoogstwaarschijnlijk neuronale vertakkingen bevatten. Voor dit doel wordt een bestaande buisfiltermethode gebruikt. Uit de gevonden gebieden worden vervolgens startpunten geselecteerd voor het traceren van de vertakkingen. Net als in het voorgaande hoofdstuk wordt hiervoor een probabilistische aanpak gebruikt op basis van sequentile Monte Carlo filtering. Ook hier levert herhaalde toepassing meer informatie over

de vertakkingen en leidt tot betere resultaten. Tevens zij opgemerkt dat de herhalingen onafhankelijk zijn van elkaar en zich dus uitstekend lenen voor parallelle implementatie. Om een volledige reconstructie te verkrijgen worden de resultaten van de verschillende herhalingen verfijnd en samengevoegd. Hiervoor worden nieuwe iteratieve algoritmes voorgesteld. Een eerste versie van de methode is opgenomen in een internationale vergelijkingsstudie genaamd BigNeuron, waar het als een van de betere methoden uit de bus kwam. In dit hoofdstuk wordt een sterk verbeterde versie gepresenteerd.

Tenslotte wordt in het vijfde hoofdstuk een haalbaarheidsstudie gepresenteerd van het detecteren van gebieden in lageresolutiebeelden van celculturen die neuronen bevatten. Deze taak is doorgaans de eerste stap in screeningstudies naar de aantasting van neuronen door neurodegeneratieve ziektes en het effect van medicijnen. De gevonden gebieden worden vervolgens afgebeeld op hoge resolutie, waarna de neuronen kunnen worden gereconstrueerd met behulp van de in de vorige hoofdstukken beschreven methoden. De detectie in de lageresolutiebeelden wordt bemoeilijkt door de afwezigheid van details, het feit dat de neuronen vaak niet volledig zijn afgebeeld, de aanwezigheid van vergelijkbare cellen zoals astrocyten, en beeldvormingsartefacten zoals (veel) ruis. Daarom is in deze studie gekozen voor het gebruik van zelflerende methoden op basis van beeldkenmerken berekend door een zeer groot aantal bestaande filtertechnieken. In de gepresenteerde experimenten worden de prestaties van vier soorten traditionele zelflerende methoden vergeleken. Ook wordt een proefexperiment beschreven met een tegenwoordig zeer populaire aanpak op basis van kunstmatige, dieplerende neurale netwerken. De conclusie is echter dat op deze beperkte data de traditionele methoden beter presteren.

Acknowledgement

The beauty of reaching a goal, besides exciting discoveries, is likewise based on the fact that a journey towards the goal often involves overcoming various hurdles. Here, I would like to acknowledge people that had helped along that journey and made reaching the goal possible.

I am very grateful to my co-promotor Erik Meijering and promotor Prof. Wiro Niessen for giving me the opportunity to do research job and for having enough patience and trust. The experience gained at Biomedical Imaging Group Rotterdam allowed discovering the world better and shaped up my views on work and life. I equally owe my supervisor a debt of gratitude for all the dedication in crafting the publications and beautiful writing suggestions that brought the work to a higher standard.

The Netherlands Organization for Scientific Research (NWO) deserves appreciation for funding the project together with all the reviewers for their constructive feedback.

I owe special gratitude to the colleagues and collaborators Ihor, Gadea, Carlos and Niels that contributed building this work up. Many thanks to Petra and Desiree for the administrative support and to all past and present members of Medical Informatics and Radiology Department.

From the bottom of my heart, I wish to thank my dear friends, one very special cousin, my parents Milija and Vera, my sister Danica and my girlfriend Letizia for all the genuine kindness and support one could wish for.

Miroslav Radojević
Rotterdam, October 2018

Publications

Publications in International Journals

- **M. Radojević**, I. Smal, E. Meijering, "Fuzzy-logic based detection and characterization of junctions and terminations in fluorescence microscopy images of neurons", *Neuroinformatics*, vol. 14, no. 2, pp. 201-219, 2016
- **M. Radojević**, E. Meijering, "Automated neuron tracing using probability hypothesis density filtering", *Bioinformatics*, vol. 33, no. 7, pp. 1073-1080, 2017
- V. Ulman, M. Maška, K. E. G. Magnusson, O. Ronneberger, C. Haubold, N. Harder, P. Matula, P. Matula, D. Svoboda, **M. Radojević**, I. Smal, K. Rohr, J. Jaldén, H. M. Blau, O. Dzyubachyk, B. Lelieveldt, P. Xiao, Y. Li, S. Y. Cho, A. C. Dufour, J. C. Olivo-Marin, C. C. Reyes-Aldasoro, J. A. Solis-Lemus, R. Bensch, T. Brox, J. Stegmaier, R. Mikut, S. Wolf, F. A. Hamprecht, T. Esteves, P. Quelhas, Ö. Demirel, L. Malmström, F. Jug, P. Tomancak, E. Meijering, A. Muñoz-Barrutia, M. Kozubek, C. Ortiz-de-Solorzano "An objective comparison of cell-tracking algorithms", *Nature Methods*, vol. 14, no. 12, pp. 1141-1152, 2017
- **M. Radojević**, E. Meijering, "Automated neuron reconstruction from 3D fluorescence microscopy images using sequential Monte Carlo estimation", *Neuroinformatics*, *in press*, 2018
- G. Mata, **M. Radojević**, C. Fernandez-Lozano, I. Smal, N. Werij, M. Morales, E. Meijering, J. Rubio, "Automated neuron detection in high-content fluorescence microscopy images using machine learning", *Neuroinformatics*, *in press*, 2018

Publications in International Conference Proceedings

- L. Šajn, **M. Radojević**, T. Dobravec, "3D volume localization using miniatures", in *IEEE International Convention on Information and Communication Technology, Electronics and Microelectronics — MIPRO 2014* (37th international conference, held in Opatija, Croatia, May 26–30, 2014), IEEE, Piscataway, NJ, pp. 411–416, 2014
- **M. Radojević**, I. Smal, W. Niessen, E. Meijering, "Fuzzy logic based detection of neuron bifurcations in microscopy images", in *IEEE International Symposium*

on Biomedical Imaging: From Nano to Macro — ISBI 2014 (11th international conference, held in Beijing, China, April 29–May 2, 2014), G. Wang and B. He (eds.), IEEE, Piscataway, NJ, pp. 1307–1310, 2014

- **M. Radojević**, I. Smal, E. Meijering, "Automated neuron morphology reconstruction using fuzzy-logic detection and Bayesian tracing algorithms", in *IEEE International Symposium on Biomedical Imaging: From Nano to Macro — ISBI 2015* (12th international conference, held in New York, NY, USA, April 16–19, 2015), E. Angelini and J. Kovačević (eds.), IEEE, Piscataway, NJ, pp. 885–888, 2015

- G. Mata, **M. Radojević**, I. Smal, M. Morales, E. Meijering, J. Rubio, "Automatic detection of neurons in high-content microscope images using machine learning approaches", in *IEEE International Symposium on Biomedical Imaging: From Nano to Macro — ISBI 2016* (13th international conference, held in Prague, Czech Republic, April 13–16, 2016), J. Kybic and M. Šonka (eds.), IEEE, Piscataway, NJ, pp. 330–333, 2016

- **M. Radojević**, E. Meijering, "Neuron reconstruction from fluorescence microscopy images using sequential Monte Carlo estimation", in *IEEE International Symposium on Biomedical Imaging: From Nano to Macro — ISBI 2017* (14th international conference, held in Melbourne, VIC, Australia, April 18–21, 2017), G. Egan and O. Salvado (eds.), IEEE, Piscataway, NJ, pp. 36–39, 2017

PhD Portfolio

Research Skills:

- B.Sc. degree in Electrical Engineering, University of Belgrade (Serbia), 2008
- M.Sc. degree in Computer Vision and Robotics (ViBot), by a consortium of the Heriot-Watt University (United Kingdom), Université de Bourgogne (France) and Universitat de Girona (Spain), 2011

In-Depth Courses:

- Knowledge driven Image Segmentation, ASCI, LUMC Leiden, 2012
- Advanced Pattern Recognition, ASCI, TUD, Delft, 2012
- Computer Vision by Learning, UvA, Amsterdam, 2014
- Course on R, EMC, Rotterdam, 2013

International Conference Presentations:

- INCF Neuroinformatics Congress — Neuroinformatics 2014, Leiden, the Netherlands, August 25-27, 2014
- IEEE International Symposium on Biomedical Imaging: From Nano to Macro — ISBI 2015, New York, USA, April 16-19, 2015
- IEEE International Symposium on Biomedical Imaging: From Nano to Macro — ISBI 2017, Melbourne, Australia, April 18-21, 2017

Summer schools and workshops:

- Summer School on Image Processing, TU Wien, Vienna, Austria, 2012
- 6th CIMST Interdisciplinary Summer School on Biomedical Imaging, ETH, Zurich, Switzerland, 2012

- Summer school on Finite Set Statistics, HWU, Edinburgh, United Kingdom, 2013
- ICT with industry, Lorentz Centrum, Leiden, 2013

Other:

- Referee activities for international scientific journals (IEEE Transactions on Medical Imaging, Bioinformatics) and international conferences (Bioimaging-Biostec). Seminar talks at Biomedical Imaging Group Rotterdam.

Curriculum Vitae

Miroslav Radojević was born in Užice, Serbia, on January 7, 1984. He received a B.Sc. degree in Electrical Engineering from the University of Belgrade - School of Electrical Engineering, Serbia, in 2008.

From 2008 to 2009, he was a Development Engineer with ABS Electro. During that period he developed protection algorithms for digital protective relays.

In 2009 he was awarded Erasmus Mundus scholarship by the European Commission and from 2009 to 2011 took part in the international master programme in Computer Vision and Robotics (ViBot) offered by a consortium of the Heriot-Watt University (United Kingdom), Universitè de Bourgogne (France) and Universitat de Girona (Spain). In 2011 he received a M.Sc. degree graduating on the underwater robotics project "Underwater Vehicle Localization using Extended Kalman Filter". During that period he participated in 2011 edition of the SAUC-E Student AUV Challenge Europe.

From Dec. 2011 to Feb. 2016 he was a Ph.D. student at the Departments of Medical Informatics and Radiology of the Erasmus University Rotterdam, the Netherlands. His research topic was automated reconstruction of neuron cells in microscopy images. The results are described in this thesis.

Since 2017, he has been a software engineer with Becton, Dickinson and Company (BD), working at the research and development department within BD Life Sciences.