GEOMETRIC MULTISCALE ANALYSIS AND APPLICATIONS TO NEUROSCIENCE IMAGING

A Dissertation Presented to

the Faculty of the Department of Mathematics

University of Houston

In Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy

By

Cihan Bilge Kayasandik

August 2017

GEOMETRIC MULTISCALE ANALYSIS AND APPLICATIONS TO NEUROSCIENCE IMAGING

Cihan Bilge Kayasandik

APPROVED:

Dr. Demetrio Labate, Committee Chair
Department of Mathematics, University of Houston

Dr. Fernanda Laezza
Department of Pharmacology and Toxicology,
University of Texas Medical Branch

Dr. Kresimir Josic
Department of Mathematics, University of Houston

Dr. Emanuel Ioannis Papadakis
Department of Mathematics, University of Houston

Dean, College of Natural Sciences and Mathematics

Acknowledgements

There are many people without whom I could not have accomplished this work. Firstly, I would like to express my sincere gratitude to my advisor Dr. Demetrio Labate for his continuous support for my Ph.D study and related research. He is a great advisor with his patience, motivation, and immense knowledge. I could not have imagined having a better advisor and mentor for my Ph.D study. Besides my advisor, I would like to thank the rest of my dissertation committee members: Dr. Fernanda Lezza, Dr. Kresimir Josic, and Dr. Emanuel Papadakis, for their insightful comments and suggestions.

There are no proper words to express my gratitude to my family. I especially want to thank my father, who has been the major architect of all my academic successes and my mother, my strongest supporter and the only one who can relieve me from all of my fears. I also need to thank my little sister Humeyra for her endless love and support. She is my friend, encourager and ally. Her humor has always alleviated the burdens of life. I love and respect her from my heart. I need to thank my uncle Zeki, my aunt Hulya and her family, who have always guided me during all of my difficult decisions, and my grandparents who amazed me with their pure love. Even though I cannot put names of all members of my family here, I want to thank each of them individually for their support and pray.

I am extremely thankful to my friends Mervis, Busra Unsal, Merve Merakli, Sila Ozen and Aysegul Karaman whom have always been my companions in spite of the distances. Almost each day I woke up with texts and calls from Merve and Busra

that brightened my day. Whenever I got stuck on academic or personal problems, Merve was my most judicious guide with her senseful advices. I want to particularly mention Busra's contributions to our chocolate chip cookie baking challenges and filming projects as a good company with her killer ambition. My memories of our college years with Mervis always made me feel good. I want to thank my housemates in Houston, Nuray Gul and Dr. Husna Karaboga, for their support especially during the last days of my study, their patience for my stress and also for proofreading of this dissertation. And special thanks goes to my friend Esra who is my housemate and fellow for more than a decade. Although I am away from my family she, by herself, makes me feel I am home. Our conversations based on music, movies and art always relieved stress of work. Our attempts for making short films, painting and writing had been best cures for tense PhD years. With her I have joyful moments even in the dullest times. I also need to thank her for spending 6 hours of proofreading of this dissertation.

Of course I need to mention Dean and Sam Winchester brothers, since they were with me all along my PhD life for five years, and were my comfort zone during my stressful times. Similarly a great thanks goes to Peter Jackson, Miyazaki, Zara, Mac Demarco, Blue Oyster Cult and The Smiths. Finally I should mention my cats Ranosh and Puf who do not care about me at all but still fill my days with joy and peace. Although they did nothing special for me I still owe them a lot.

GEOMETRIC MULTISCALE ANALYSIS AND APPLICATIONS TO NEUROSCIENCE IMAGING

An Abstract of a Dissertation

Presented to

the Faculty of the Department of Mathematics

University of Houston

In Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy

By

Cihan Bilge Kayasandik

August 2017

Abstract

This thesis is concerned with the development of quantitative methods for the analysis of neuronal images. Automated detection and segmentation of components of neurons in fluorescent images is a major goal in quantitative studies of neuronal networks, including applications of high-content-screenings where one needs to compute multiple morphological properties of neurons. Despite recent advances in image processing targeted to neurobiological applications, existing algorithms of soma detection and neurite tracing still have significant limitations which are more severe when processing fluorescence image stacks of neuronal cultures. To address such challenges, in this dissertation, we develop several novel methods and algorithms aimed at extracting quantitative information in fluorescent images of neuronal cultures or brain tissue, including methods for the automated detection of the soma and other subcellullar structures of interest, and algorithms for cell classification. Our methods rely on technique from harmonic analysis, especially wavelets and more advanced multiscale representation systems. Using these techniques, we are able to extract highly informative image characteristics with high geometric sensitivity and computational efficiency. As part of our work, we include a theoretical justification and an extensive numerical validation on microscopy imaging data provided by our collaborators in neuriscience. An extensive comparison with state-of-the-art existing methods demonstrate that our algorithms are highly competitive in terms of accuracy, reliability and computational efficiency.

Contents

List of Figures

List of Tables

CHAPTER 1

Introduction

1.1 Multiscale Directional Analysis of Images

A major challenge in image analysis is how to efficiently and accurately model images belonging to a certain class of interest. From a mathematical viewpoint, this consists in defining a useful function space or a representation method that is able to describe classes of images of interest in the applied sciences.

Multiscale methods and wavelets have been very successful in image processing during the past 20 years due to their excellent approximation properties (relatively

few coefficients are sufficient to capture the properties of interest of images), good time-frequency localization (representation coefficients carry local information) and high computational efficiency. However, it is also known that although conventional wavelets are optimally efficient in dealing with point discontinuities, they are not very efficient to deal with edges and other distributed singularities commonly found in multivariate data. This limitation is due to the fact that conventional wavelets are essentially isotropic. In order to process multivariate data, it is desirable to apply methods combining multiscale analysis and directional sensitivity, therefore they can efficiently capture anisotropic information. Several ideas were proposed in the literature to address this task, starting with rather heuristic methods such as the directional wavelet filter banks introduced by Antoine et al. [6] and the directional wavelets by Bamberger and Smith [7]. In more recent years, more sophisticated tools were introduced including the ridgelets and curvelets of Candes and Donoho [8], [9], the bandlets of Mallat and Peyre [37], the complex wavelets of Kingsbury [29], the contourlets of Do and Vetterli [13] and the class of composite wavelets and shearlets, introduced by Guo, Kutyniok and Labate and Guo [30]. These methods offer multiscale directional representations targeted to multivariate functions and a complete theoretical foundation which is useful to assess the ultimate performance of the methods. The backbone of this dissertation is the application of multiscale directional analysis methods to address problems in neuroscience image processing.

We start with some background material.

1.2 Geometric Multiscale Analysis

1.2.1 Representation systems

Rapid advances in the technology of data acquisition during the past decade have made data access easier and this calls for faster and more efficient ways to analyze data. Even in the most general scenarios, however, the first step of any data processing pipeline consists in setting a mathematical framework to model the data.

In signal analysis, data is frequently modeled as a piecewise continuous function, which has finite energy. Hence, it can be describes as a function from the space of square integrable functions, $L^2(\mathbb{R}^d)$, where d is the dimension of data. When we model a signal as a function $f \in L^2(\mathbb{R}^d)$, we basically have the representation of the data in the standard basis of $L^2(\mathbb{R}^d)$ [31].

A representation system in $L^2(\mathbb{R}^d)$ is formed by a set of functions or vectors, which is called a dictionary, and a transformation, which maps a function to its representation with respect to a given dictionary. A representation is an analysis tool which decomposes data into components reflecting the significant features of data. Note that a vector space is not limited to have one representation system, it can have many. Different dictionaries enable data to be represented in different ways in the same domain which manipulates the data in order to process it more efficiently. While in the literature the definition of representation is often very loose, it is usually preferable to assume that it is an invertible map since invertibility makes it possible to transfer data across different spaces as desired. A suitable representation can work

as a direct analysis tool, as well as it can be used as a preprocessing step to increase the efficiency of analysis. Therefore an accurate and appropriate representation of the data can be considered as the initial step of data analysis.

Perhaps the most classical representation system is a *basis* in a Hilbert space. We recall that a Hilbert space is a complete inner product space. We also recall that an inner product on a complex vector space V is a function $\langle \cdot, \cdot \rangle : V \times V \to \mathbb{C}$ satisfying:

- For any $v \neq 0 \in V$, $\langle v, v \rangle > 0$.
- For any $v, w \in V$, $\langle v, w \rangle >= \overline{\langle w, v \rangle >}$.
- For any $v, w \in V$, $c \in \mathbb{C}$, $\langle cv, w \rangle >= c\langle v, w \rangle >$.
- For any $u, v, w \in V$, $\langle u + v, w \rangle >= \langle u, w \rangle > + \langle v, w \rangle >$.

For example, the space $L^2(\mathbb{R}^d)$ is an Hilbert space with inner product given by

$$\langle f, g \rangle = \int_{L^2(\mathbb{R}^d)} f(x)\overline{g(x)}dx,$$

and the sequence space ℓ^2 is a Hilbert space with inner product is defined as

$$\langle (x_k), (y_k) \rangle = \sum_{1}^{\infty} x_k \overline{y_k}.$$

In a Hilbert space H, if a countable dictionary $\{x_n\}$ gives a unique representation for each $f \in H$ such as $f = \sum_n a_n(f)x_n$, then $\{x_n\}$ is called as *basis* [22]. In this case, the sequence $\{a_n(f)\}_n$ is called as the representation of f with respect to basis

$\{x_n\}$. As a basis representation is unique for each function, it is like a signature of the function.

A well-known basis representation system is an orthonormal basis:

Definition 1.2.1. Orthonormal basis [22]: *Let $\{x_n\}$ be a sequence in a Hilbert space H. $\{x_n\}$ is an orthonormal basis for H if every $x \in H$ can be written $x = \sum_{n=1}^{\infty} c_n x_n$ for a unique choice of scalars c_n and $\langle x_m, x_n \rangle = \delta_{mn}$ for each n. Furthermore, the reconstruction for any $f \in H$ is satisfied and*

$$f = \sum_{n \in \mathbb{N}} \langle f, x_n \rangle x_n.$$

holds.

An example of an orthonormal basis for $L^2(\mathbb{T})$, where $\mathbb{T}$ stands for the space of 2π-periodic functions, is the sequence $\{e^{2\pi i n x}\}_{n=-\infty}^{\infty}$ which is called the trigonometric basis. The representation of a function with respect to the trigonometric basis is given by the Fourier transform.

Definition 1.2.2. Discrete Fourier transform [22] : *For $f \in L^1(\mathbb{T})$ where $\mathbb{T}$ is the space of 2π-periodic functions, the Fourier transform of f is given by*

$$\hat{f}(n) = \int_0^{2\pi} f(x) e^{2\pi i x n} dx = \langle f, e^{2\pi i n} \rangle,$$

and $f(x) = \sum_{n\in\mathbb{Z}} \hat{f}(n)e^{2\pi inx}$*, for* $x \in \mathbb{R}$*. Here the representation* $\{\hat{f}(n)\}_n$ *is the unique representation of* f*.*

The sequence $\{\langle f, x_n\rangle\}_{n\in\mathbb{N}}$ is the representation of f with respect to the dictionary $\{x_n\}$. Since $\{x_n\}_n$ is an orthonormal basis, this representation is unique. Although uniqueness seems to be important it may be a deficiency in some cases. For example, if we have a unique representation of a data that means all coefficients in the representation are irreplaceable for the reconstruction. In applications from signal analysis, it is possible that some information of a signal could be lost, e.g., during transmission, and in that case it would be impossible to recover the signal from the incomplete information. Redundant dictionaries, where the representation is non-unique, are introduced to deal with this type of data loss problem. [22]

1.2.2 Redundant and sparse representations

A frame generalizes the notion of a basis in Hilbert space.

Definition 1.2.3. Frames [22]: *A sequence* $\{x_n\}$ *in a Hilbert space* H *is a frame for* H *if there exist constants* $0 < A, B < \infty$ *such that*

$$A||f||^2 \leq \sum_n |\langle f, x_n\rangle|^2 \leq B||f||^2$$

holds for each $f \in H$. A and B are called as lower and upper frame bounds respectively. The largest of all possible lower frame bounds A is called the optimal lower frame bound, similarly the smallest of all possible upper frame bounds B is called the optimal upper frame bound.

The upper bound, B, ensures the stability of the frame, that is, any permutation of the frame elements will give again a frame with the ability to represent a function using a finite norm expansion. On the other hand, the nonzero lower bound, A ensures the completeness, i.e., $span\{x_n\}$ is dense in H.

If the optimal lower and upper frame bounds are equal, $A = B$, then $\{x_n\}$ is a tight frame. If further, $A = B = 1$ then $\{x_n\}$ is a Parseval frame. So, by the above definition, for a Parseval frame $\{x_n\}$ one has:

$$||f||^2 = \sum_n |\langle f, x_n \rangle|^2.$$

which makes any orthonormal basis also a Parseval frame. However, the converse is not true in general and there are Parseval frames that are not orthonormal bases.

There is no restriction on the procedure of generating frame sequences, for example a zero vector could be member of a frame or elements can be repeated. The easiest way to generate a frame is using an orthonormal basis for H with small modifications. For example, if $\{e_n\}_n$ is an orthonormal basis for H, $\{e_1, e_1 + e_2, e_2, \cdots, e_n, \cdots\}$ is a frame which is not a basis for H.

Even if there is no basis property, as we will show below a frame is associated with a basis-like representation of each $f \in H$. This representation is not unique in general, therefore it is possible to have multiple frame representations for the same element of H.

Because of the existence of upper frame bound, the representation of $f \in H$ with respect to frame sequence $\{x_n\}_n$ belongs to the $\ell^2(\mathbb{R})$ space, i.e., the space of square summable sequences. Therefore a frame defines a mapping from H to ℓ^2 by decomposing $f \in H$. This mapping is called *analysis operator* and satisfies:

$$S : H \to \ell^2$$

$$S : f \to \{\langle f, x_n \rangle\}_n.$$

The adjoint of S, S^* is called *synthesis operator*:

$$S^* : \ell^2 \to H$$

$$S^* : \{\langle f, x_n \rangle\}_n \to \sum_n \langle f, x_n \rangle x_n.$$

and for any $f \in H$. The *frame operator* is defined as

$$S^*S : H \to H$$

$$f \to \sum_n \langle f, x_n \rangle x_n.$$

The frame operator is bounded because of the definition of frame, i.e. $AI_H \leq S^*S \leq$

BI_H where I_H is the identity function in H. It is also invertible and self-adjoint. The reconstruction formula of $f \in H$ given by the frame system $\{x_n\}_n$ is

$$f = \sum_n \langle f, S^{-1}(x_n)\rangle (x_n) = \sum_n \langle f, x_n\rangle S^{-1}(x_n).$$

Here $\{S^{-1}(x_n)\}_n$ forms another frame system and called by *canonical dual frame* of $\{x_n\}$. If the frame is a Parseval frame, the *canonical dual frame* of $\{x_n\}$ is the frame itself and the reconstruction formula looks exactly like the reconstruction formula of an orthonormal basis.[22]

1.2.3 The wavelet transform

The theory of wavelets emerged in the 1990's to address some limitations of classical Fourier analysis, namely, its limitations in providing local features of signals. This is due to the fact that the basis functions of the trigonometric basis are periodic functions, with support defined over all of $\mathbb{R}$. As a result, any local change in a signal affects all of its Fourier components. To define a representation system which is able to capture the local properties of a signal, its basis functions must be local or approximately local. In the wavelet approach, this is obtained by considering building blocks made of "little waves", that is, functions that are local both in the time and frequency domain.

A wavelet system is formed by an appropriate wave function ψ with its time-scale

shifts. That is, a wavelet system on $L^2(\mathbb{R})$ is defined as:

$$\{\psi_{j,k} = D_2^{-j} T_m \psi = 2^{j/2}\psi(2^j \cdot -k) : j,k \in \mathbb{Z}\},$$

where $\psi \in L^2(\mathbb{R})$. Here D_t stands for the dilation operator and is defined as $D_t\psi = t^{-1/2}\psi(t^{-1/2}\cdot)$ and T_t stands for the translation operator which is defined as $T_t\psi = \psi(\cdot - t)$ and the corresponding wavelet transform is

$$\mathcal{W}_\psi : f \to \langle f, \psi_{j,k}\rangle \qquad j,k \in \mathbb{Z}.$$

Typically, the generator function ψ (also called the mother wavelet) is chosen to be well localized, that is, it exhibits rapid decay both in time and Fourier domain. For example, ψ can be chosen to be compactly supported with rapid decay in the Fourier domain, or band-limited (i.e., compact support in the Fourier domain) with rapid decay in the time-domain. Recall that it is not possible to construct a function that has compact support and is band limited at the same time. The construction of appropriate wavelet generators is an important topic in wavelet theory. Several properties of the wavelet system, including regularity and fast decay can be controlled by the choice of mother function. [53], [22]

1.2.4 Multiresolution analysis

The multi-resolution analysis (MRA) was introduced by Mallat as a method to generate orthonormal wavelet bases for $L^2(\mathbb{R})$. A MRA consist of a sequence of nested

closed subspaces $\{V_n\}_n$ with some special properties and a scaling function that satisfy the following properties.

Definition 1.2.4. Multiresolution Analysis [22]: *A multi resolution analysis (MRA) for $L^2(\mathbb{R})$ is a sequence $\{V_n\}_{n\in\mathbb{Z}}$ of closed subspaces of $L^2(\mathbb{R})$ such that*

1. $\cdots \subset V_{-n} \subset \cdots \subset V_{-1} \subset V_0 \subset V_1 \subset \cdots \subset L^2(\mathbb{R})$.
2. *V_{n+1} is the dilation of V_n by factor 2:*

$$V_{n+1} = \{f(2x) \colon f \in V_n\},$$

3. *$\cup_{n\in\mathbb{Z}} V_n$ is dense in $L^2(\mathbb{R})$,*
4. *$\cap_{n\in\mathbb{Z}} V_n = \{0\}$,*
5. *there exists a function $\phi \in V_0$ such that $\{T_k\phi\}_{k\in\mathbb{Z}}$ is an orthonormal basis for V_0.*

ϕ is called as scaling function for the MRA.

Here we define closed subspaces W_n to satisfy $V_{n+1} = V_n \bigoplus W_n$, where $\bigoplus$ stands for orthogonal direct sum. The sets W_n's are the *wavelet spaces*. Because of the property 3 of the MRA, $\bigoplus_{n\in\mathbb{Z}} W_n$ is dense in $L^2(\mathbb{R})$. By the property 2 of MRA, if we can find an orthonormal basis ω of W_0, then the corresponding system of scale shifts of ω will be an orthonormal basis of $L^2(\mathbb{R})$. Another useful property of MRA

is that for each MRA there exists a corresponding orthonormal wavelet system. In other words, one can build a wavelet system which is an orthonormal basis for $L^2(\mathbb{R})$ through a given sequence of subspaces$\{V_n\}_{n\in\mathbb{Z}}$ and a scaling function which satisfy the above properties. The scaling function ϕ must satisfy

$$\hat{\phi}(\xi) = m_0(\xi/2)\hat{\phi}(\xi/2),$$

where m_0 is a 2π-periodic function in $L^2([0, 2\pi])$ and it satisfies

$$|m_0(\xi)|^2 + |m_0(\xi + \pi)|^2 = 1 \text{ a.e.}$$

Then the corresponding wavelet function ψ satisfies

$$\hat{\psi}(\xi) = e^{i\xi/2}\overline{m_0(\xi/2 + \pi)}\hat{\phi}(\xi/2).$$

These observations are useful to generate the wavelet function from a given scaling function. Furthermore, one can design a wavelet system with the desired regularity and decay properties by using the appropriate scaling function.

In order to form an orthonormal basis, the mother wavelet must satisfy the *admissibility property*:

$$\int_{-\infty}^{\infty} \frac{|\hat{\psi}(\xi)|^2}{|\xi|} d\xi < \infty.$$

That implies $\hat{\psi}(0) = \int_{-\infty}^{\infty} \psi(x)dx = 0$. Therefore ψ must change its sign at the origin. [11]

We have focused so far on wavelet systems on $L^2(\mathbb{R})$. To extend the wavelet approach to the multivariate case, the simplest approach consists in generating separable wavelet systems by taking tensor products of univariate wavelet systems. For instance, in dimension $d = 2$, a separable wavelet system will be of the form

$$\{\psi_{j,k_1,k_2}(x_1, x_2) = 2^j \psi_1(2^j(x_1 - k_1))\, \psi_2(2^j(x_2 - k_2)) : j, k_1, k_2 \in \mathbb{Z}\},$$

for appropriate generators ψ_1, ψ_2. However, while such systems are used in several applications, they are not as efficient as their one-dimensional counterpart to represent signals containing discontinuities. The reason is that separable wavelets have a directional bias due to their special structure and this implies that they cannot approximate very efficiently signal containing edge discontinuities. As we mentioned above, this limitation led to the introduction of alternative systems such as curvelets and sharlets. [31]

1.2.5 The shearlet transform

A shearlet system is a multivariate wavelet-like system obtained by applying appropriate translations and dilations to a generator function. The dilation operator is designed in such a way to endow the system with anisotropy and directional selectivity.

Definition 1.2.5. Continuous Shearlet System [31]: *A continuous shearlet system associated with the generator function $\psi \in L^2(\mathbb{R}^2)$ is a collection of*

functions of the form:

$$\psi_{\alpha,s,p}(x) := \{T_p D_{A_\alpha} D_{S_s} \psi \colon \alpha > 0, s \in \mathbb{R}, p \in \mathbb{R}^2\}.$$

where T_t is the translation operator and D_t is the dilation operator as defined before. A_α is the parabolic scaling matrix, defined as $A_\alpha = \begin{pmatrix} \alpha & 0 \\ 0 & \alpha^{1/2} \end{pmatrix}$ *and S_s is the shearing matrix, defined as* $S_s = \begin{pmatrix} 1 & s \\ 0 & 1 \end{pmatrix}$

In the shearlet system, the parabolic scaling ensures that one direction is dilated quadratically with respect to the perpendicular direction, therefore the supports of the analyzing functions become increasingly more elongated at fine scales. The shearing matrix has the effect similar to rotation, effectively generating analyzing functions at multiple orientations. With respect to the rotation matrix, the shearing matrix offers the advantage of preserving the integer lattice when the shearing parameter s is restricted to the integers. This is an advantage in discrete implementations.

Discrete shearlets are obtained by sampling the scaling and shearing variables on an appropriate discrete set.

Definition 1.2.6. Discrete shearlets [31]: *For a given $\psi \in L^2(\mathbb{R}^2)$, the associated discrete shearlet system is defined by*

$$SH(\psi) = \{\psi_{j,k,l} = 2^{\frac{3}{4}j}\psi(S_k A_{2^j} \cdot -l) \colon j,k \in \mathbb{Z}, l \in \mathbb{Z}^2\}$$

and the corresponding shearlet transform is

$$f \to (SH)_\psi f(j,k,l) = \langle f, \psi_{j,k,l}\rangle, \qquad (j,k,l) \in \mathbb{Z}\times\mathbb{Z}\times\mathbb{Z}^2.$$

Similar to wavelet systems, one can generate a discrete shearlet system with special properties by choosing a proper generator function. In particular, one can find an admissible generator therefore the corresponding shearlet system is a well localized Parseval frame. Below is definition of the classical shearlet generator.

Definition 1.2.7. Classical shearlet [31]: *Let $\psi \in L^2(\mathbb{R}^2)$ be defined by*

$$\hat{\psi}(\xi) = \hat{\psi}(\xi_1,\xi_2) = \hat{\psi}_1(\xi_1)\hat{\psi}_2(\frac{\xi_2}{\xi_1}),$$

where $\hat{\psi}_1 \in L^2(\mathbb{R})$ satisfies

$$\sum_{j\in\mathbb{Z}} |\hat{\psi}_1(2^{-j}\xi)|^2 = 1 \quad \textit{for a.e. } \xi \in \mathbb{R}$$

with $\hat{\psi}_1 \in \mathcal{C}^\infty(\mathbb{R})$ and $supp\hat{\psi}_1 \subset [\frac{-1}{2},\frac{-1}{16}] \cup [\frac{-1}{2},\frac{-1}{16}]$, and $\psi_2 \in L^2(\mathbb{R})$ is a bump

function in the sense that

$$\sum_{k=-1}^{1} |\hat{\psi}_2(\xi + k)|^2 = 1 \quad \textit{for a.e. } \xi \in [-1, 1],$$

satisfying $\hat{\psi}_2 \in \mathcal{C}^\infty(\mathbb{R})$ *and* $supp\hat{\psi}_2 \subset [-1, 1]$. ψ *is called as a classical shearlet.*

A discrete shearlet system associated with the classical shearlet generator is a Parseval frame of $L^2(\mathbb{R}^2)$. Therefore $(SH)_\psi$ forms a useful analysis and synthesis tool for $L^2(\mathbb{R}^2)$. [31]

1.2.6 Cone adapted shearlet transform

Due to the asymmetry of the construction, parabolic scaling and shearing operators cause the support of the shearlet functions $\psi_{j,k}$ to become increasingly more elongated, as it is seen in Figure 1.2, but also to cover rather unequally the Fourier plane. To overcome this directional bias, there is a modified construction which consists essentially in combining two separate shearlet systems.

Definition 1.2.8. Cone-adapted shearlet transform [31]: *For* $\phi, \psi, \tilde{\psi} \in L^2(\mathbb{R}^2)$ *and* $c = (c_1, c_2) \in (\mathbb{R}_+)^2$, *the cone adapted discrete shearlet system* $\mathcal{SH}(\phi, \psi, \tilde{\psi}; c)$ *is defined by*

$$\mathcal{SH}(\phi, \psi, \tilde{\psi}; c) = \Phi(\phi; c_1) \cup \Psi(\psi; c) \cup \tilde{\Psi}(\tilde{\psi}; c),$$

where

$$\Phi(\phi; c_1) = \{\phi_m = \phi(\cdot - c_1 m) \colon m \in \mathbb{Z}^2\},$$

$$\Psi(\psi; c) = \{\psi_{j,k,l} = 2^{\frac{3}{4}j}\psi(S_k A_{2^j} \cdot - M_c l) \colon j \geq 0, |k| \leq \lceil 2^{j/2} \rceil, l \in \mathbb{Z}^2\},$$

$$\tilde{\Psi}(\psi; c) = \{\tilde{\psi}_{j,k,l} = 2^{\frac{3}{4}j}\psi(S_k \tilde{A}_{2^j} \cdot - \tilde{M}_c l) \colon j \geq 0, |k| \leq \lceil 2^{j/2} \rceil, l \in \mathbb{Z}^2\},$$

with

$$M_c = \begin{pmatrix} c_1 & 0 \\ 0 & c_2 \end{pmatrix} \quad \textit{and} \quad \tilde{M}_c = \begin{pmatrix} c_2 & 0 \\ 0 & c_1 \end{pmatrix} \quad \textit{and} \quad \tilde{A}_\alpha = diag(\alpha^{1/2}, \alpha).$$

If $c = (1, 1)$, *the parameter* c *is omitted in the formulas above.*

As seen in the definition $\mathcal{SH}$ is combination of three generating functions which are responsible of different subregions in frequency domain. The corresponding subregions for each of these generating functions are illustrated in Figure 1.1

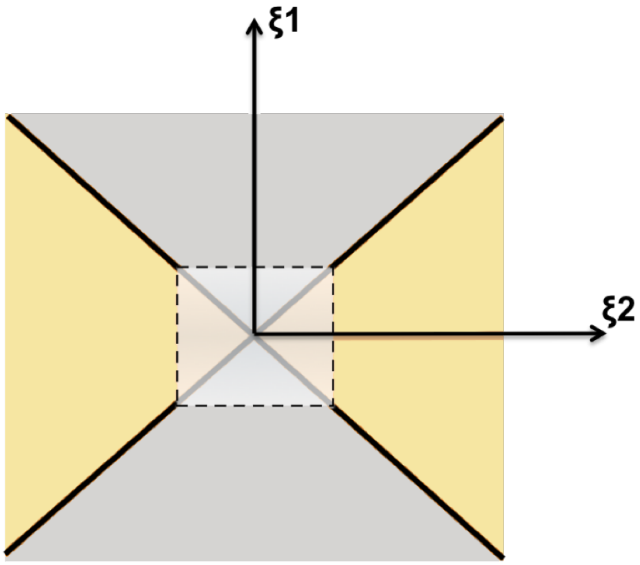

Figure 1.1: Cone adapted shearlet transform. Cone adapted shearlet transform seperates frequency domain into five subregions.The shearlet scaling function ϕ is associated with central white region, shearlet generator function ψ is associated with horizontal orange conic regions, $\tilde{\psi}$ is associated with vertical grey conic regions.

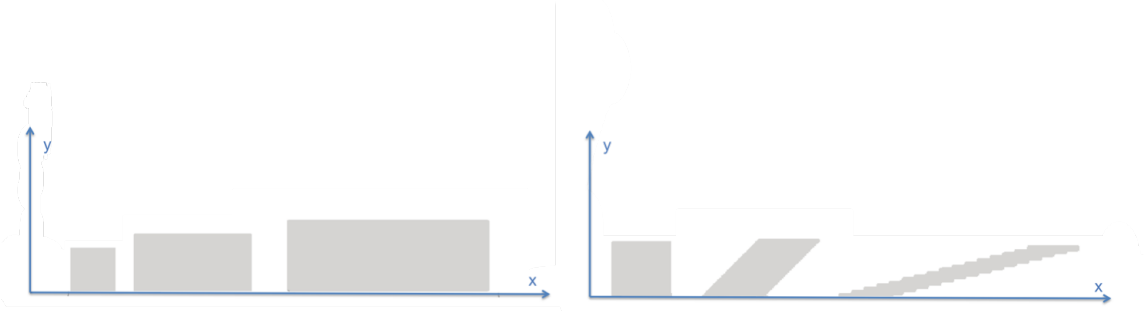

Figure 1.2: Effect of parabolic scaling and shearing on square region. *left:* Parabolic scaling causes the region evolve along one axis more than the other one. Therefore regions loses its characteristic and become more elongated. *right:* Shearing operator shifts the upper vertices of square along a line parallel to x-axis. This causes very thin regions in large shearing factors.

1.3 Neuroscience Imaging

Neurons are the main functional units of the human brain and each one includes three main components: a soma, several dendrites and one axon as shown in Figure 1.5.

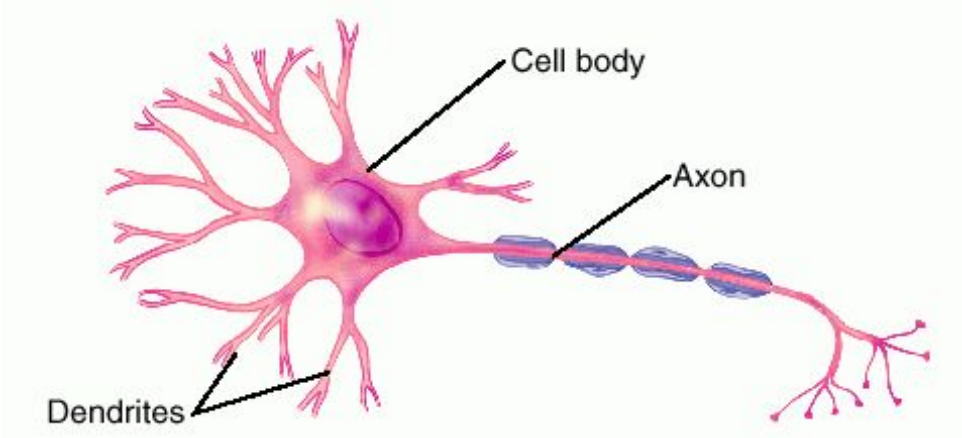

Figure 1.3: A scheme of neuron. A simple neuron has three main components. (1) soma (cell body): The main body of the cell, (2) dendrite: the vessel like structures which receives the input, (3) axon: vessel like structure to send signals. [**?**]

Neurons exhibit a highly complex and diverse morphology which has led neurobiologists to investigate the connection between neuronal structure and neuronal function. While early neuroanatomical studies were hand-drafted, advances in technologies and accessibility to larger amounts of data have led to the development of more sophisticated methods of neuroimage analysis. In more recent years, the impact of ideas from mathematics and computer science has grown significantly, spurred by the need to develop accurate and automated methods that can be applied to process large amounts of data with high efficiency. High-content screenings (HCS), for instance, require the identification and extraction of multiple morphological features of neurons, such as soma shape and volume, neurite length and branching properties

from large data sets. Such complex information, usually compiled from multi-channel fluorescence images, requires automated processing methods to handle large batches of data and establish a confident statistical basis for a reliable prediction model. With the rapid and widespread rise in the use of HCS in basic science settings, automated detection of morphological characteristics of neurons from fluorescent images has become an area of very active research. Despite the advances in this area, however, there is still a lack of quantitative methods targeted to the needs of fluorescent microscopy which is caused by challenges in the neural images analysis.

Fluorescent microscopy is an imaging modality which has gained increasingly more relevance in neurobiology. Thanks to the application of a wide array of very selective fluorescent dyes, this method enables the visualization of subcellular compartments of a neuron with remarkable accuracy. As a result, fluorescent microscopy has been instrumental to several important discoveries in the field of neuroscience [28].

However, processing images acquired through fluorescent microscopy comes with some very specific challenges, due to the uneven distribution of the signal, the small range of contrast levels and the noise introduced by the stochastic nature of the image acquisition process. For such reasons, conventional image processing algorithms may perform poorly when applied to fluorescent images. Furthermore, the objects of interest in neuroscience imaging are often topologically complex and contain structures at multiple resolution level making the processing of such images very challenging. For instance, the segmentation of dendritic arbors from fluorescent images of neurons is very complex and generic image segmentation packages are unable

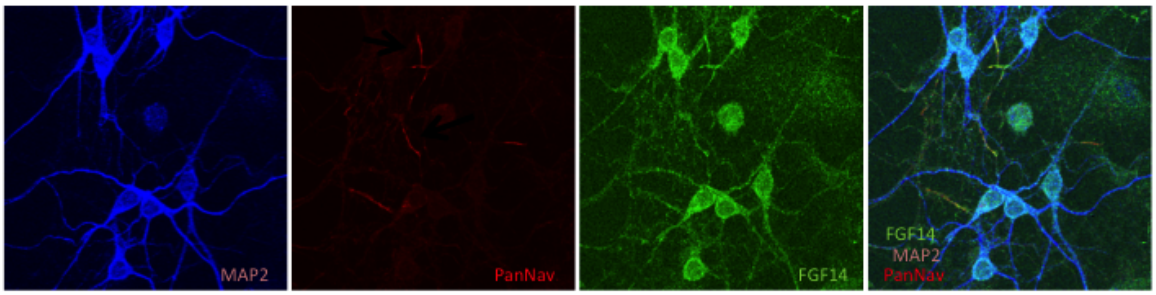

Figure 1.4: Confocal stacks of fluorescent images. Confocal stacks of neuronal culture stained for proteins; MAP2(blue channel), PanNav (Red Channel), FGF14 (Green Channel). The image was provided by Dr. Laezza at the University of Texas Medical Branch. Image size = 512×512 pixels (1 pixel = $0.28 \times 0.28 \mu m$).

to provide accurate segmentation in general. As we will show in this Dissertation, even a problem apparently as simple as the automated separation of somas from neurites in a fluorescent image of a neuronal network requires some special care to be solved reliably. Consider for instance the image in Fig. 1.4. Methods attempting to separate the soma based on intensity thresholding are ineffective since high intensity values are also found outside the soma. Conventional morphological operators also perform poorly due to the irregularity and variation in size of the somas. The method we developed relies on introducing a measure of local isotropy which is measured using a collection of multiscale directional filters. As we will show, this method is theoretically justfied and performs very reliably.

1.4 Biological Analytics

This doctoral research is motivated by the need to develop innovative quantitative methods targeted to the analysis of neuronal images. While a very significant efforts has been made in this direction, for example, in the development of methods for denoising and segmentation of neuronal images, several challenges remain. In particular, methods for the identification of subcellular compartments of neurons and extraction of morphological features still require a large component of manual intervention. The work in this thesis aims to show that emerging methods from harmonic analysis, including the new class of directional multiscale representations emerged during the past decade, have the potential to facilitate the development of a new generation of more accurate and computationally efficient algorithms for the analysis of fluorescent images of neurons.

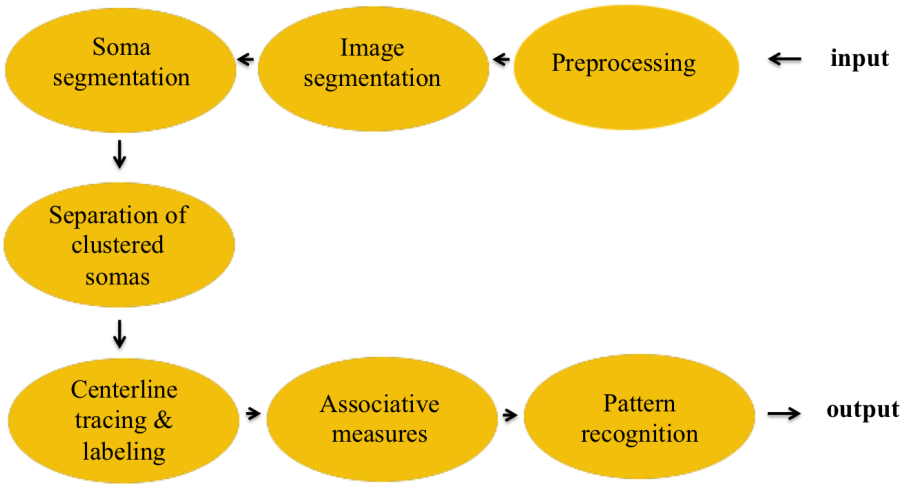

Figure 1.5: Pipeline of bio image analysis. Input is fluorescent multi channel neural image. *Preprocessing:* Neural culture data is prepared for further analysis firstly by converting the z-stack to 2d image through maximum projection. If data is already 2d or a volume, this step is skipped. After the data is present, shearlet based denoising is applied. *Segmentation:* The cell is separated from background with SVM based classification method. In order to generate classification features, orientable laplacian and shearlet filters are employed. *Soma segmentation:* Soma regions are detected with Directional ratio, then segmented with help of Fast Marching method. *Separation of contiguous somas:* If there is any contiguous soma, it is detected automatically and separated. *Centerline tracing:* starting from the segmented somas, neurites are detected and traced along the centerline. After the tracing step is completed, by comparing intensities along centerline on different color channels, neurites are labeled as axon of dendrite. *Associative measures:* Along the centerline corresponding intensity features are collected. since the background is an artifact, it is eliminated automatically. *Interpretation of features:* For each of the somas, which has axons, intensity vectors are collected form two dendrites and one axon. These features are used to cluster corresponding cells with the help of pattern recognition techniques. Output is classified cells.

As part of this doctoral research, we investigated the following specific and highly relevant problems.

1.4.1 Automated detection of somas in fluorescent images of neuronal networks

We introduce an innovative algorithm for the detection and extraction of somas in fluorescent images of neuronal cultures or brain tissue where somas and other structures exist in the same fluorescent channel. Our method relies on a new geometrical descriptor called Directional Ratio and a collection of multiscale orientable filters to quantify the level of local isotropy in an image. To optimize the application of this approach, we introduce a new construction of multiscale anisotropic filters that is implemented by separable convolution. In combination with the application of the Fast Marching method, our algorithm is able to reliably detect somas in 2 and 3 dimensions, accurately segment them and separate contiguous ones.

These methods will be discussed in detail in Section 2.

1.4.2 Automated extraction of neuronal trees

Automatic extraction of the tree structure of neurons in fluorescent imaging can be particularly challenging due to the topological complexity of the data and the irregularities of fluorescent signal intensity that may cause thin neurites to appear broken and neighbouring ones to merge. Difficulties are accentuated in fluorescent images of neuronal cultures where neurons develop mostly horizontally and neurites from different neurons may physically overlap. In order to overcome these challenges, we introduce a method base on front propagation starting from each soma location which is utilized with a well designed decision rule at the intersection of branches.

Numerical experiments show that our method is highly reliable and accurately extracts trees associated with individual neurons in a network where neurites from different cells may overlap.

The neuronal trees obtained from our method provide a spatial reference system to compute the local fluorescent intensity signal along each neurite. Using multispectral fluorescent images, we can automatically compute local fluorescent intensity profiles on cells exposed to different types of perturbations. Hence generating measurements that can be used for problems of high content screening.

This algorithm will be discussed in detail in Section 3.

1.4.3 Neuronal classification

As part of a collaboration with a team of neuroscientists from UTMB, we have investigated what perturbations associated with a number of kinase inhibitors are explanatory variables of changes in cell network architecture and expression levels of macromolecular complexes of the axonal initial segment (AIS). This work is motivated by problems in drug discovery since many brain disorders can be traced back to molecular events happening at the single cell level. Change in the molecular composition of the AIS has received particular attention in recent years.

In order to correlate the kinase inhibitors to composition of AIS proteins, we applied our methods described above to measure expression levels of macromolecular components through of the AIS in neurons exposed to different perturbations. The major problem to address is the variability in the measurements due to cellular noise

and noise caused by acquisition method. In order to eliminate irrelevant information from the measurements we applied a shift-invatiant wavelet transform to generate the features for our classifier. For the classification, we applied a Support Vector Machine (SVM) classifier.

This algorithm will be discussed in detail in Section 4.

CHAPTER 2

Soma Detection and Segmentation

[1]

Accurate identification of soma location and morphology in neuronal images is a critical task in many neuroscience studies. It is known that neuronal characteristics including soma volume and surface morphology are fundamental features for discriminating different types of neurons [50]. Detecting somas also plays a major role in extracting the connectivity and graph structure of a network of neurons as each soma location is identified with the root of a (directed) graph associated with the corresponding cell [3, 4].

[1] Materials from [27] are used in this chapter.

In applications of high-content screening (HCS), multiple morphological properties of neurons need to be identified including soma location and shape characteristics. This information is usually collected from large sets of multi-channel fluorescence images therefore automated processing methods are often necessary to handle such large data. However the automated analysis of confocal images of neuronal cultures presents a number of challenges. In cultures, somas are usually visible in the channel marked by the Microtubules Associated Protein 2 (MAP2) antibody staining or by a nucleus marker (e.g., DAPI or TROPO-3), neither of which is soma-selective. Hence further processing is needed to identify somas. An additional difficulty is that such confocal images typically consist of stacks containing 10-25 images. As a result, only 10-25 pixels are available along the z-direction as compared with the x and y directions which contain many more pixels (typically 512 or more). Due to this large difference in pixels number, it may be inefficient or even impossible to process such data using conventional 3D filters.

Automated or semi-automated methods for soma and cell detection found in standard image analysis packages frequently rely on binary masks generated by contrast enhancement and image intensity thresholding [43, 52]. While these methods can be very effective in phase-contrast microscopy [52, 18], they are often unreliable when applied to the analysis fluorescent images since high intensity values are commonly found outside somas. A number of alternative methods proposed in recent years to deal with images where somas and other structures exist in the same fluorescence channel. These methods combine smoothing filters to regularize fluorescence intensity and classical morphological operators (e.g., morphological opening) to separate

somas from connecting neurites [3, 21, 51, 46].

The main drawback of these methods is that they are very sensitive to the parameters of the algorithm therefore they typically require a significant manual intervention to perform efficiently. In addition, many such methods have proven to be impractical or inefficient in 3D setting, even though some recent studies have shown a clever way to process 3D information by projecting the original image stack onto its the three orthogonal planes [54].

To overcome the limitations of existing methods especially in the context of neuronal cultures, an innovative approach was recently introduced by Labate et al. that relies on a novel multiscale descriptor called Directional ratio to separate somas from dendrites [42]. By quantifying the degree of local isotropy in images, this method was shown to detect somas very accurately and reliably in 2D images. However, it is computationally intensive as it requires the computation of multiple directional filters at various scales and the application of the level set method. In addition, since more directions and larger filters are needed in 3D, direct application of this method in 3D is impractical. One major aim of this dissertation is to introduce and demonstrate a major improvement of this method which relies on two novel ideas: (i) the application of a new class of multiscale directional filters which can be implemented by separable convolution to detect soma locations very efficiently; (ii) the application of the Fast Marching method to extract the soma regions and separate contiguous somas. Through these ideas we are able to speed up the extraction of somas in a 2D image by over 20 times with respect to the results in [42] while keeping the same excellent level of accuracy and reliability.

Remarkably, our method can detect somas more efficiently than algorithms based on conventional morphological operators, which were known to be very fast, with much higher levels of accuracy. Another contribution of this work is the algorithmic implementation of our new approach based on Directional Ratio to 3D setting. The performance of our 2D and 3D algorithms is extensively validated on multiple confocal images of neuronal cultures and brain tissue, and successfully compared against state-of-the-art methods from the literature.

2.1 Materials and Methods

We consider the problem of detecting soma locations in fluorescent image stacks of neuronal cultures or neuronal tissue both in 2D and 3D settings.

Our algorithm for soma detection and extraction – whether in 2D or 3D – consists of the following steps. *Preprocessing:* It is designed to remove noise and improve image quality. *Segmentation:* It separates neurons from background. *Soma detection:* It is designed to find somas. *Soma extraction:* It is designed to identify the entire soma regions and split somas that are clustered together.

In the following, we describe the methods we developed and applied to address each processing step, with most emphasis on the last two steps which contain our main original contributions.

2.1.1 Preprocessing

Preprocessing is designed to take full advantage of the capabilities of instrumentation by reducing sources of image degradation such as blurring and noise.

A common denoising routine found in biomedical applications is *Gaussian smoothing* [20], which consists in convolving an image with a Gaussian function $g(x) = \frac{1}{2\pi\sigma^2}\exp\left(-\frac{\|x\|^2}{2\sigma^2}\right)$, where σ^2 is a variance parameter. This operation has the effect of smoothing the image with a smoothing level controlled by σ^2. While this method is simple to implement and computationally efficient, it has the undesirable effect of blurring edges with the consequent loss of spatial information.

To avoid the above limitation, we adopted instead *wavelet shrinkage* which consists in: (i) transforming the image using the wavelet transform; (ii) filtering the resulting wavelet coefficients using a shrinkage function whose parameters are automatically determined from the data; (iii) applying the inverse wavelet transform to the shrinked coefficients to obtain a restored image. Wavelet shrinkage denoising was theoretically proven to be optimal with respect to the mean-squared error for estimating piecewise smooth signal corrupted by additive Gaussian noise [16, 17] and was shown to perform very competitively with fluorescence microscopy imaging data [32, 36, 57, 42].

2.1.2 Segmentation

A widely used image segmentation strategy is *intensity thresholding*, which consists in setting all pixels whose intensity value is below a certain threshold to 0, and setting those pixels above or equal the threshold to 1. To automatically set a threshold, one classical approach is Otsu's method [20], which assumes that the intensity distribution is bimodal and calculates the optimum threshold separating the two classes.

The main setback of intensity thresholding is that it considers only the pixel intensity and ignores any relationships between pixels, with the result that pixels identified in the region of interest may fail to be contiguous. This performance issue is more severe as the noise level increases.

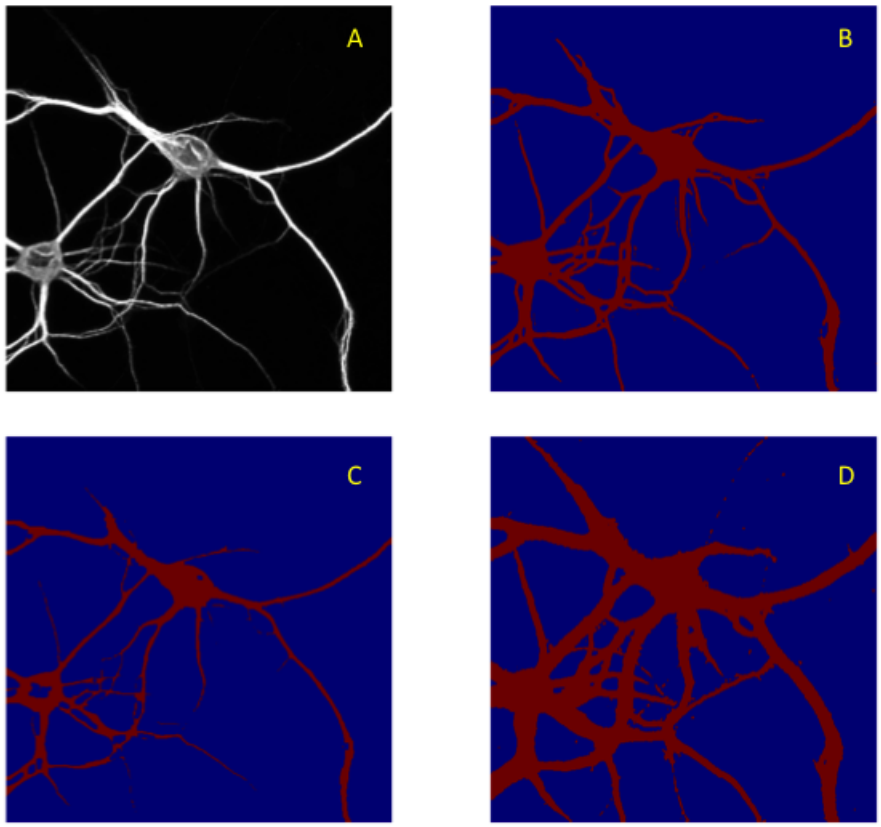

Figure 2.1: Comparison of segmentation routines. Image size = 512×512 pixels (1 pixel = $0.28 \times 0.28 \mu m$). (A): Denoised image of a neuronal culture. (B-D): Image segmentation results using our SVM-based method. Red pixels are points belong to cell, blue pixels are points belong to background. (B); intensity thresholding, Otsu's method (C); intensity thresholding, threshold based on median of the image (D).

To ensure a more faithful preservation the geometry of data, we adopted a segmentation strategy based on Support Vector Machines (SVM) originally introduced by one of the authors [23, 25] and whose main novelty is the use of features generated by a set of multiscale Laplacian and multiscale directional filters designed to capture tubular structures in neuronal images. As for many algorithms of this type, the proper classification stage of the algorithm is preceded by a training stage of

the classifier which may be computationally intensive. However, training needs to be run only once as long as the segmentation algorithm is applied to images of the same type (e.g., same cell type and image acquisition setting). The entire procedure is fully automated. Fig. 2.1 illustrates the segmentation of a 2D image of a neuronal culture using different strategies. The result in the figure suggests that the SVM approach is more effective at capturing the true data structure.

2.1.3 Soma detection

Our method for soma detection relies on the Directional Ratio, an approach recently introduced in [33, 41] to quantify the degree of local isotropy in an image and shown to be very effective for separating somas from neurites in fluorescent images of neurons [42].

2.1.4 Directional Ratio

Given a collection of multiscale orientable filters $\{\phi_{j,\ell}\}$, where the indices j, ℓ are associated with a range of scales and orientations, respectively, the *Directional Ratio* of an image f at the j-th scale and at location p is the quantity

$$D_j f(p) = \frac{\min_\ell\{|f * \phi_{j,\ell}(p)|\}}{\max_\ell\{|f * \phi_{j,\ell}(p)|\}}\,. \tag{2.1}$$

For instance, in dimension 2, the simplest choice of filters $\phi_{j,\ell}$ are the functions

$$\phi_{j,\ell}(x) = \chi_{S_{j,\ell}}(x),$$

where χ_A is the indicator function of A and the sets $S_{j,\ell}$ are the scaled and rotated rectangles $S_{j,\ell} = 2^j R_{\theta_\ell} S$, where $R_\theta = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix}$ and S is a fixed rectangle. Clearly, the same idea applies in 3D setting if one replaces 2D rectangles with 3D rectangles and uses 3D rotations.

The Directional Ratio ranges between 0 and 1, and it measures the degree of directional coherence of an image f at given scale and location. It is proved in [33, 41] that, if f is an image containing blob-like and vessel-like structures, then, for an appropriate range of scales controlled by j, there exists a threshold T significantly less than 1 such that the Directional Ratio does not exceed T when p is located inside a vessel-like structure. In contrast to that, when p is located strictly inside a blob-like structures then the Directional Ratio is close to 1 (See Figure 2.2). Note that Directional Ratio is not guaranteed to be close to 1 near the boundary of a blob-like structure. Nevertheless it was shown that one can reliably detect somas in a segmented image of a neuron f by computing the Directional Ratio $D_j f(p)$ at an appropriate scale controlled by j and discarding those points p for which $D_j f(p) < T$. Specifically, j is chosen therefore the filter length is larger than the radius of the neurites and close to the radius of the somas.

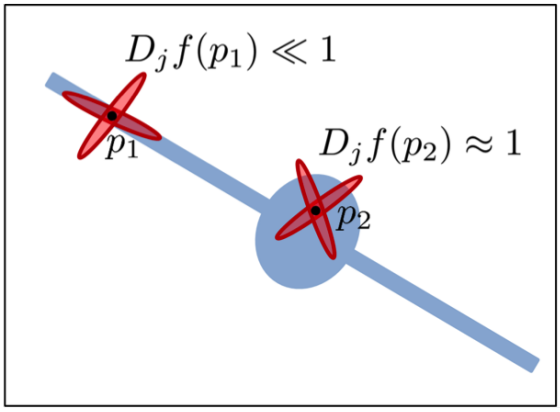

Figure 2.2: Directional Ratio. Due to the different behavior of orientable filters, for an appropriate range of scales, the Directional Ratio is much smaller than 1 at p_1 inside a vessel, and it is about 1 at p_2, inside a blob structure.

The practical implementation of equation (2.1) requires computing multiple filtered images $f * \phi_{j,\ell}$ for various orientations ℓ (at a fixed scale j). For a 2D image f of size $N \times N$ the computation of each filtering step $f * \phi_{j,\ell}$ using FFT to implement convolution requires $O(N^2 \log N)$ operations. Using L orientations, this brings the total number of operations to $O(LN^2 \log N)$. Applying the same reasoning in 3D, with the same density of orientations, the computational cost would be $O(L^2 N^3 \log N)$ operations. This shows that the computational cost of directly implementing equation (2.1), as it was carried out in [42], is already significant in 2D and would be unacceptable in 3D setting. A test on a $512 \times 512 \times 512$ image stack run on a standard laptop (2.4GHz processor) with a Matlab code requires about 103 hours to compute the Directional Ratio with $L^2 = 40$ directional filters. Since the main driver of the cost is explicit convolutions, we focused on reducing the cost of that process. To address this problem we considered two approaches: (1) adopting separable convolution instead of usual one, (2) using computationally more efficient

filters instead of rectangular one.

2.1.5 Separable convolutions

A matrix is *separable* if it can be expressed as the product of a column and a row vector [2]. It is a known property that if a 2D matrix filter is separable, then a 2D convolution can be implemented more efficiently.

We recall that 2D discrete convolution of an image f and a 2D filter h is given by

$$y[m,n] = f[m,n] * h[m,n] = \sum_j \sum_i h[i,j] f[m-i, n-j] = \langle h, T_{(m,n)} \tilde{f} \rangle.$$

where $\tilde{f} = f(-x)$. If the filter is separable, then $h[m,n] = h_1[m] h_2^t[n]$ where $h_1[m]$, $h_2[n]$ are appropriate column vectors. Hence a direct computation shows that

$$y[m,n] = \sum_j h_2[j] \sum_i h_1[i] f[m-i, n-j] = \langle h_2, \langle h_1, T_{(m,n)} \tilde{f} \rangle \rangle,$$

that is, 2D convolution is computed by performing two 1D convolutions, first along the rows and then along the columns.

Hence according to the previously given image and filter sizes computational cost will reduce to $O(L2N \log N)$ from $O(LN^2 \log N)$.

[2] This condition is verified if the matrix has rank 1.

Originally rotated rectangular filters are used in the Directional Ratio computation which are not separable unless rotation angle is either 0 or $\pi/2$. Thereby separable convolution is not applicable to our case without modification. Luckily we can still apply separable convolution if we rotate image instead of rotating filters. Filtering the image with rotated filters is same with filtering the rotated image with axis aligned rectangular filters and rotating the filtered image back. Therefore for real valued image f and h define $\tilde{h}(x) = h(-x)$:

$$\begin{aligned}
(f * R_{-\theta}h)(x) = \langle f, T_x R_{-\theta}\tilde{f}\rangle &= \int f(y) R_{-\theta}\tilde{h}(y-x)dy \\
&= \int f(y) R_{-\theta}h(x-y)dy \quad \text{(substitute } y = R_{-\theta}y\text{)} \\
&= \int f(R_{-\theta}y) R_{-\theta}h(x - R_{-\theta}y)dy \\
&= \int f(R_{-\theta}y) h(R_{\theta}(x - R_{-\theta}y))dy \\
&= \int f(R_{-\theta}y) h(R_{\theta}x - y))dy \\
&= \int R_{\theta}f(y) h(R_{\theta}x - y)dy \\
&= \int R_{\theta}f(y) \tilde{h}(y - R_{\theta}x)dy \\
&= \langle R_{\theta}f, T_{R_{\theta}x}\tilde{h}\rangle = (R_{\theta}f * h)(R_{\theta}x)
\end{aligned} \tag{2.2}$$

where R is the rotation matrix. Through this idea separable convolution can be applied to the directional rectangular filters. Adopting shearing instead of rotation is wiser since it preserves the integer grid. Equation (2.2) holds for shearing matrix S too since $\det S = 1$.

2.1.6 Directional Ratio with sheared filters

Shearing is a transformation which keeps the area and shifts the geometric object along a line. Since upper and lower triangular matrices are used as a shearing matrix, shearing keeps one coordinate of points same. That is the reason why computational effort of shearing is less than rotation.

Horizontal shearing is shifting parallel to x-axis and the matrix form is:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} 1 & \lambda \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix},$$

vertical shearing is shifting parallel to y-axis and the matrix form is:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ \lambda & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}.$$

where $\lambda = \tan\theta$ and θ is the shearing angle.

On the discrete case the deformation on the sheared image would be large if the shearing angle is larger than $\pi/4$. Therefore we have taken the adaptation from *cone-adapted continuous shearlet transform* [30] and then used vertical shearing when the shearing angle is less than $\pi/4$ or larger than $3\pi/4$ and used horizontal shearing otherwise. By this method we minimize the error caused by shearing.

Through that approach we save from computational cost as a consequence of separable convolution but applying shearing operator to input twice for each oriented

filter is computationally very expensive. Therefore the overall gain from computational effort is not noteworthy. An alternative way to reduce cost would be filtering image with anisotropic Gaussian function since it has both separable convolution and recursive filtering implementation.

2.1.7 Anisotropic Gaussian filters

Several constructions of orientable anisotropic filters were proposed in the literature such as the rotated Gaussians by Perona [45]. For our implementation of the Directional Ratio (2.1) we will employ a version of such filters that can be computed very efficiently.

In 2 dimensions, an anisotropic Gaussian function is obtained by scaling a 2D Gaussian using different factors σ_x and σ_y (say, $\sigma_x = 10\sigma_y$) in the x and y directions, respectively:

$$g_0(x, y; \sigma_x, \sigma_y) = \frac{1}{2\pi\sigma_x\sigma_y} \exp\left(-\frac{1}{2}\left(\frac{x^2}{2\sigma_x^2} + \frac{y^2}{2\sigma_y^2}\right)\right).$$

By rotating the coordinate axes by an angle θ, one obtains oriented anisotropic Gaussian functions:

$$g_\theta(x, y; \sigma_x, \sigma_y) = \frac{1}{2\pi\sigma_x\sigma_y} \exp\left(-\frac{1}{2}\left(\frac{(x\cos\theta + y\sin\theta)^2}{\sigma_x^2} + \frac{(-x\sin\theta + y\cos\theta)^2}{\sigma_y^2}\right)\right).$$

Geusebroek et al. [19] introduced a very efficient method to separate the convolution

with the anisotropic Gaussian $g_\theta(x, y; \sigma_x, \sigma_y)$ as the composition of a 1D convolution with a Gaussian filter in the x direction followed by another 1D convolution with a Gaussian filter in a non-orthogonal direction, that is

$$g_\theta(x, y; \sigma_x, \sigma_y) = \frac{1}{2\pi\sigma_x\sigma_\phi} \exp\left(-\frac{1}{2}\frac{x^2}{\sigma_x^2}\right) * \exp\left(-\frac{1}{2}\frac{t^2}{\sigma_\phi^2}\right),$$

where $t = x\cos\phi + y\sin\phi$ and ϕ is an appropriate functions of θ. An illustration of this decomposition is shown in Fig. 2.3. Using a recursive approximation to implement 1D Gaussian convolutions, this method yields an implementation that is very accurate and faster than a FFT-based 2D convolution, as it requires only $O(1)$ multiplications per pixel. We adopted this implementation to compute our Directional Ratio. In this case, the scale of the filters is controlled by σ_x and σ_y.

The same idea extends to the n-dimensional case, as shown by Lampert and Wirjadi [34]. In particular, similar to 2D case we can represent any 3-dimensional rotated anisotropic Gaussian filters as the composition of three 1-dimensional Gaussian filters aligned along non-orthogonal directions. Again these filters have fast implemention using separable convolution.

Adapting the above idea to the n-dimensional case, Lampert and Wirjadi [34] introduced a method to decompose n-dimensional rotated anisotropic Gaussian filters into n 1D Gaussian filters aligned along non-orthogonal directions.

An n dimensional anisotropic Gaussian filter with mean 0 and covariance matrix Σ has the form

$$g(x) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}x^t\Sigma^{-1}x\right)$$

where $|\Sigma|$ is the determinant of Σ. The following is proved in [34].

> For any decomposition $\Sigma = VDV^t$ of the covariance matrix Σ into square matrices D and V where D is diagonal and positive and V is a upper triangular matrix with determinant 1, there is a separation of the n-dimensional Gaussian into 1D Gaussians, where the separation directions are given by the column vectors of V [34].

That is, one can write

$$g(x) == \frac{1}{\sqrt{2\pi}d_1}\exp(-\frac{v_1^2}{2d_1^2})\ldots\frac{1}{\sqrt{2\pi}d_n}\exp(-\frac{v_n^2}{2d_n^2})$$

where $d_1,\ldots,d_n$ are diagonal entries of the matrix D and $v_1,\ldots,v_n$ are the nonzero entries of upper triangular matrix V, which give the direction of line convolutions.

Therefore, covariance decomposition factors directly apply to input. Since the matrix V is upper triangular, it can be considered as a shearing matrix. Hence decomposition $\Sigma = VDV'$ geometrically means shearing input with matrix V^t, applying axis aligned gaussian functions whose densities are diagonals of matrix D, then shearing in the opposite direction with V. This is similar with our idea that we attempted to use in shear filters section. Unlike sheared filters this new method will be computationally affordable because of nice properties of gaussian function filtering implementation.

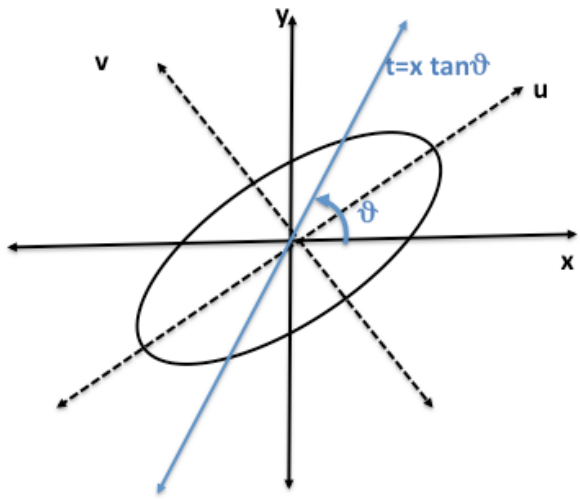

Figure 2.3: Nonorthogonal separation of anisotropic Gaussian in 2D. rotated anisotropic Gaussian filter can be decomposed along its main axes u and v, but it is not useful for implementation. The decomposition into x-axis and t-line is more useful and requires less computational cost.

2.2 Automatic Scale Selection based on Scale-Space Theory

In order to determine the optimal filter size we adapted the idea of scale space theory. The idea bases on finding the scale of the local isotropy of the points. The definition of local isotropy is as follows:

Definition 2.2.1. *Let $A \subset R^2$. If $x \in A$ we say that the set A is locally isotropic at the location x and at scale $s > 0$ if $B(x, s/2) \subset A$. [40]*

Since our soma detection algorithm is designed to use radius of the largest soma

contained in the image, we designed the automated scale selection algorithm to predict this value. Since the soma regions are not perfectly blob, our method predicts the radius of the largest disk that is completely covered by the soma. Therefore the optimal scale value to use for our soma detection method is:

$$\sup\{s \in \mathbb{Z} \colon B(x, s/2) \subset I, x \in I\},$$

where I is the image. Thus basically the largest scale of local isotropy of points in image, is the optimal scale for our soma detection method. In order to measure the local isotropy level of a point $x \in I$, I is filtered with orientable directional filters with various scales and at various orientations centered at $x \in I$. If x is locally isotropic at scale s then for any $s' \leq s$ filtering response is 1 for all oriented normalized filters with scale s'. For scale $s' \geq s$ the filtering response starts decreasing which shows that x is locally isotropic with scale s.

The algorithm calculates the largest scale of local isotropy for each point of input. Then the maximum of all is chosen as estimated optimal scale which gives the radius of largest disk contained in cell. In application we used only two orthogonal directions for filtering, with directional angles $\pi/4$ and $-\pi/4$, since our numerical experiments showed that filtering at many directions do not improve the result significantly.

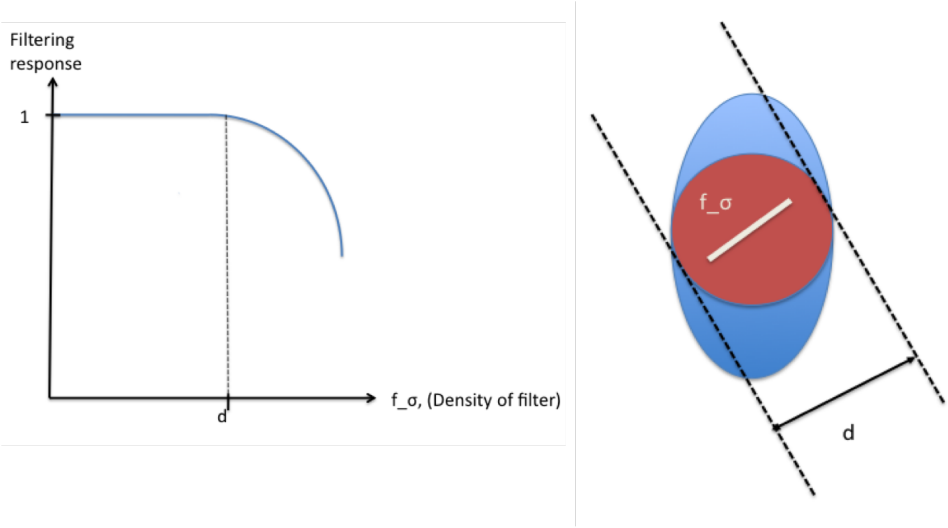

Figure 2.4: Scale estimation. Figure illustrates the automatic scale selection algorithm. On the right there is a ellipse-like (blue) object and algorithm estimates the diameter(in terms of density of Gaussian function) of the largest blob (red) contained in. The algorithm is inspired by the filter response of the centre (point p) of blobs according to different scales. The filter response of the filter at point p which is the center of the blob is given in the left plot. When the filter (white rectangle at the center of blob) density, f_σ, reaches value d, filtering response starts decreasing. d is the diameter of the given blob in terms of density of Gaussian function.

In order to see the performance of algorithm we examined 20 images to segment soma regions with both automatically selected scale and user determined scale. Our numerical tests show that automated scale selection does not change overall performance of the method. However it makes out method fully automated with an insignificant cost, which is 0.67 sec. for 512x512 images. Performance result for our method with automatically selected scale is given in Appendix Table 5.1 and Table 5.2. The average result of accuracy of our method is given below in Table 2.1, accuracy of the morphological methods is given in Table 2.2.

Table 2.1: Performance results of Anigauss when scale selected automatically

	TPR	FPR	DC	Auto Scale Selection Cost	Detection rate
AVG	0.90	0.29	0.84	0.67 sec	71 TP, 0 FN, 0 FP

Table 2.2: Performance results of MORPH2 when scale selected automatically

	TPR	FPR	DC	Auto Scale Selection Cost	Detection rate
AVG	0.86	1.16	0.69	0.67 sec	65 TP, 6 FN, 20 FP

2.3 Soma Extraction

As we observed above, the Directional Ratio of the segmented image of a neuron is expected to be close to 1 inside a soma but its value may be much lower near its boundary. Hence, by thresholding the Directional Ratio, we will only find a region strictly inside the soma (see Fig. 2.5). To identify the entire soma including the region near its boundary, we need to grow the initial soma region. In our studies we develop an approach based on the Fast Marching method.

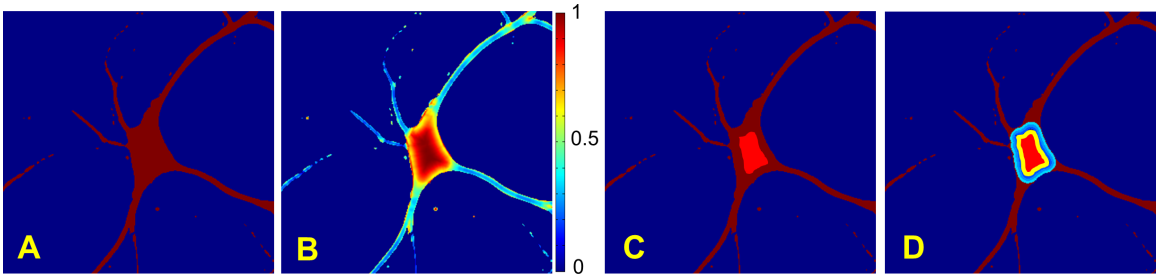

Figure 2.5: Soma extraction. Image size $= 512 \times 512$ pixels (1 pixel $= 0.28 \times 0.28 \mu m$). Starting from the segmented image of a neuron (A), where red points belong to cell and blue points belong to background, we compute the Directional Ratio in (B); whose values are real numbers range from 0 (blue) to 1 (red). By thresholding those pixels in (B) was value in above 0.85, we detect an initial soma region in (C). We apply the Fast Marching algorithm to evolve the boundary of the initial soma region in (C) until it finds the entire soma region in (D). The colors show the evolution of boundary at different time steps.

2.3.1 Level Set and Fast Marching methods

The *Level Set* and *Fast Marching* methods are variational approaches introduced to track evolution of curves and shapes without having to parametrize these objects [39, 47]. In [42], the Level Set approach was used with the boundary curve of the region found by the Directional Ratio inside the soma as the initialization curve Γ of the level set evolution equation. Even though this method provides excellent results, it is computationally intensive and its direct extension to 3D setting is impractical.

We apply here the Fast Marching method as an alternative faster approach for tracking a moving boundary, which is designed for problems in which the speed function never changes sign, therefore the front is always moving forward or backward. This assumption allows one to convert the evolution problem to a stationary formulation, which has much faster implementation (about $N \log N$ operations for an N size grid) than a Level Set method, even though the latter one is more flexible.

The Fast Marching method builds the evolving curve Γ by computing the arrival time $T(p)$, as the time when the curve crosses a location p, with evolution speed F given by the user. The equations which determine the expanding front is:

$$|\nabla T|F = 1 \quad T = 0 \quad on \quad \Gamma$$

$$Front = \Gamma(t) = \{(x, y)|T(x, y) = t\} \quad F > 0$$

The selection of the speed F is the critical factor in the application of this method. In our situation, taking again the boundary curve of the initial soma region as the initialization curve, we want the speed F of the evolving curve to decrease as it approaches the boundary of the soma and finally stop at the boundary. This suggests that the Directional Ratio of the segmented image could be a good candidate for the velocity map as its values are larger inside the soma and decrease at the boundary. However, the Directional Ratio does not vanish near the boundary of the soma or at the connected neurites, therefore the curve Γ would continue to evolve outside the soma and inside the neurites. To deal with this problem, we use the following strategy. To generate a sharper decrease away from the interior of the soma, we set $F(p)$ equal to the modified Directional Ratio

$$\frac{\min_\ell\{|f * \phi_{j,\ell}(p)|^3\}}{\max_\ell\{|f * \phi_{j,\ell}(p)|\}}.$$

Due to the power introduced in the numerator, this quantity decreases faster than the Directional Ratio away from the soma which is illustration is given in Figure

2.6. Additionally, this function is thresholded by setting very small values (below 0.00001 in our experiments) to 0 in order to ensure that the evolution will stop when Γ reaches the soma's boundary.

For our numerical implementation of the Fast Marching method, we adapted the Fast Marching Matlab toolbox by G. Peyré which is based on [12, 47].

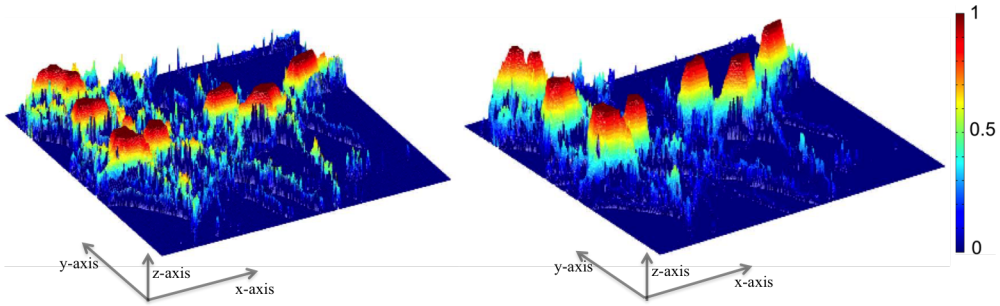

Figure 2.6: Optimal Speed function for Fast Marching Approach. Optimal speed map is shown as a 3d image. Intensity values (which are shown in z-axis) are real numbers between 0 (blue) and 1 (red). *Left*: Directional Ratio of input, *Right*: Modified Directional Ratio of input which gives a sharper decrease in Directional Ratio values close to boundary of soma. Hence it is a better speed function for Fast Marching method

2.3.2 Separation of clustered somas

Our method to extract somas may detect multiple contiguous somas as a single one. To address this issue, we use the following approach developed by one of the authors in [42]. After applying our method based on Directional Ratio at the default scale, we check the extracted soma area. If this area differs from the expected area more than three times of the estimated standard deviation, then we conclude that it contains more than one soma. Next, we compute again the Directional Ratio at a coarser scale, that is, using twice as long directional filters; the application of a

threshold on the Directional Ratio will now produce a smaller initial set inside each true soma region. Finally, we apply again the Fast Marching routine using the new boundary curves for each initial set. This method is run automatically and, as shown by numerical tests in the Results section, reliably separates contiguous somas.

2.4 Specimen Preparation and Imaging

Images used in our studies are primary hippocampal neuronal cultures that were prepared in Dr. Laezza's Laboratory at the Department of Pharmacology & Toxicology of the University of Texas Medical Branch.

Confocal images were acquired with a Zeiss LSM-510 Meta confocal microscope with either a 63X or a 40X oil immersion objective (1.4 NA). Multi-track acquisition was done with excitation lines at 488 nm for Alexa 488, 543 nm for Alexa 568 and 633 nm for Alexa 647. Respective emission filters were band-pass 505-530 nm, band-pass 560-615 nm and low-pass 650. Stacks were collected at z-steps of 1 μm with a frame size of 512 $\times$ 512, pixel time of 2.51 μs, pixel size 0.28 $\times$ 0.28 μm (63X objective) or 0.44 $\times$ 0.44 μm (40X objective) and a 4-frame Kallman averaging.

Banker's style hippocampal neuron cultures were prepared from embryonic day 18 (E18) rat embryos as described in [49]. Briefly, following trituration through a Pasteur pipette, neurons were plated at low density (105 $\times$ 105 cells/dish) on poly-L-lysine-coated coverslips in 60 mm culture dishes in MEM supplemented with 10% horse serum. After 24 h, coverslips (containing neurons) were inverted and placed over a glial feeder layer in serum-free MEM with 0.1% ovalbumin and 1 mM pyruvate

(N2.1 media; Invitrogen, Carlsbad, CA) separated by approx. 1 mm wax dot spacers. To prevent the overgrowth of the glia, cultures were treated with cytosine arabinoside at day 3 in vitro (DIV).

Hippocampal neurons (DIV14) were fixed in fresh 4% paraformaldehyde and 4% sucrose in phosphate-buffered saline (PBS) for 15 min. Following permeabilization with 0.25% Triton X-100 and blocking with 10% BSA for 30 min at 37 °C, neurons were incubated overnight at room temperature with the following primary antibodies: mouse anti-FGF14 (monoclonal 1:100; Sigma Aldrich, St Louis, MO), rabbit anti-PanNav (1:100; Sigma, St Louis, MO) and chicken anti-MAP2 (polyclonal 1:25000; Covance, Princeton, NJ) diluted in PBS containing 3% BSA. Neurons were then washed 3 times in PBS and incubated for 45 min at 37 °C with appropriate secondary antibodies as described for brain tissue staining. Coverslips were then washed 6 times with PBS and mounted on glass slides with Prolong Gold anti-fade reagent.

All the animal procedures were performed in accordance to the University of Texas Medical Branch at Galveston IACUC approved protocols.

2.5 Results and Discussion

In this section, we illustrate the application of our improved soma detection and extraction algorithm on multiple fluorescent images of neurons both in 2D and 3D settings. Imaging data, as indicated above, were provided by Dr. Laezza at the Department of Pharmacology & Toxicology of the University of Texas Medical Branch.

Our numerical code is written in Matlab and includes the routine for the computation of anisotropic Gaussian filters and Fast Marching propagation described above. All numerical experiments were run using a MacBook with Intel Core i5 2.4GHz processor and 16 GB RAM. Data and open source code used to generate our results are publicly available at

`https://github.com/cihanbilge/SomaExtraction`.

2.5.1 Performance metric

To assess the performance of our algorithm on soma extraction, we adopt the following standard statistical measures of the performance of a binary classification test [5]. The *True Positive Rate TPR* (or *Sensitivity*) measures the proportion of correctly identified soma pixels with respect to the total number of true soma pixels, which are manually identified by a domain-expert (without knowledge of the algorithm results). Denoting by TP (= true positive) the number of correctly identified soma pixels and by FN (= false negative) the number of true soma pixels incorrectly rejected, we define:

$$TPR = \frac{TP}{TP + FN}.$$

The *False Positive Rate FPR* (this is the complement of the *Specificity*) measures the proportion of pixels incorrectly identified as soma pixels with respect to the total number of true soma pixels. That is, denoting by FP (= false positive) the pixels

incorrectly selected as soma pixels, we define

$$FPR = \frac{FP}{TP + FN}.$$

This rate is a penalty akin to wrong soma pixel detections. When our FPR is compared with the traditional definition $(FP)/(TN + FP)$, one notices that this last expression would be very close to zero in our neuronal images since false soma detections are much less than the number of background pixels, because of the low neuronal density in our images. Hence, we adopted a new modified definition which describes false soma detections as a percentage of soma volume measured in pixels. Finally, the *Dice Coefficient* DC is used to compare the similarity between two samples or measures and is given by

$$DC = \frac{2\,TP}{2\,TP + FN + FP}.$$

Note that the denominator $2\,TP + FN + FP = TP + FP + FN + TP$ is the sum of the detected pixels and the true soma pixels. DC can be considered as a measure of the overall effectiveness of the soma extraction algorithm.

2.5.2 2D soma analysis

Due to the difficulty of processing an image stack in 3D resolution, in several studies stacks are converted into 2D images by computing projections along the axis perpendicular to the image plane (the z axis). The *maximum intensity projection* (MIP)

for instance maps an image stack into a 2D image where each pixel contains the maximum value over all images in the stack at that pixel location.

For our 2-dimensional tests, we considered 20 MIP images obtained from 20 standard field-of-view confocal image stacks of low-density neuronal cultures, as commonly used in phenotypic screenings of analytes for drug-discovery or biomarker identification (cf. [14, 38, 48, 49]). Each image stack comprises between 10 and 25 images and contains between 1 and 10 neurons, for a total of 71 somas in the 20 data sets we considered. According to the processing pipeline described in the Materials and Methods section, images were first preprocessed and segmented; the Directional Ratio was computed; an initial region was obtained by thresholding the Directional Ratio with threshold = 0.85; finally, by applying the Fast Marching algorithm to the initial region, somas were extracted and contiguous somas separated, if needed.

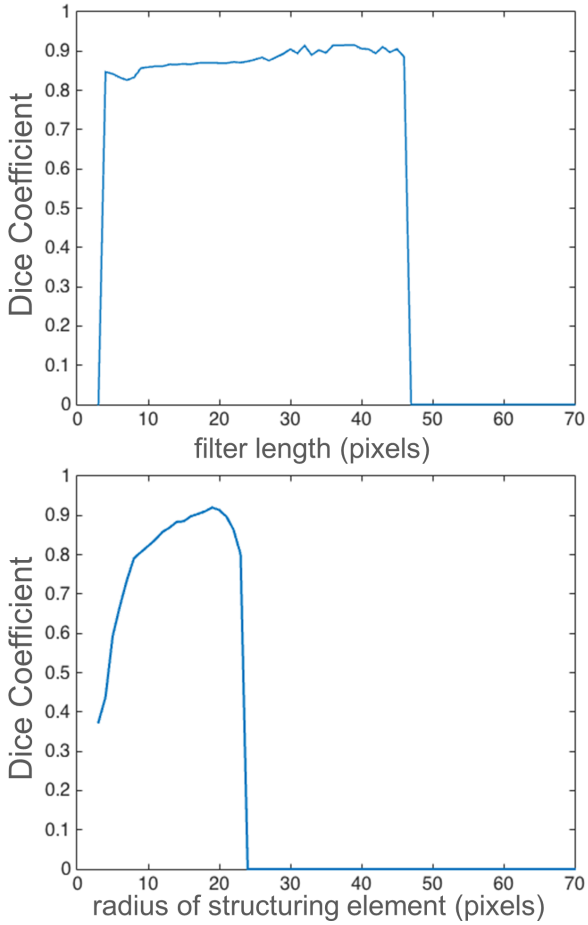

Figure 2.7: Sensitivity on scale parameter. Plots illustrate the soma extraction performance, according to the Dice Coefficient, as a function of the filter length for our algorithm (above) and as a function of the radius of the structuring element for an algorithm using the morphological opening operator (below).

For the implementation of the Directional Ratio, we considered the two types of directional filters described above, i.e., rectangular and anisotropic Gaussian filters. As this method requires to compute the filtered image for multiple orientations at an appropriate scale, we need to select the scale parameter and the number of orientations.

According to the theory, the scale parameter must be such that filter length is close to the radius of the somas, say 85% of it. Our data include images acquired at magnification 40X and 63X, and on such data the expected radius of a soma is about 32 and 48 pixels, respectively. Hence, for our rectangular filters we set filter length equal to 27 and 40 pixels, respectively. Similarly, for the anisotropic Gaussian filters, we set σ_x equal to 9 and 13 pixels, respectively (here we assume that the length of the filter is approximately $3\sigma_x$). Fig. 2.7 shows that the performance of our algorithm is clearly dependent on the selection of the scale parameter but not too sensitive to its value. Even though, for the image considered here, our method performs better when filter length is near 40 pixels, yet the algorithm performance is overall very consistent in the range 5-45 pixels. By contrast, the figure shows that a method based on conventional morphological operators is typically much more sensitive to the scale parameter.

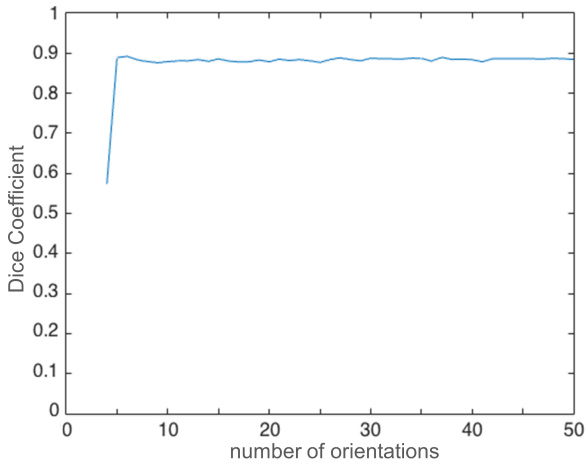

Figure 2.8: Sensitivity on the number of orientations. The plot illustrates the sensitivity of our algorithm performance, according to the Dice Coefficient, as a function of the number of orientations of the directional filters.

For the selection of the number of orientations, it is expected that the algorithm performance would improve (or at least would not worsen) by increasing the number of orientations since the Directional Ratio would become potentially more able to detect changes in geometry. On the other hand, computing time increases with the number of orientations as more filtered images are being computed; thus, we wish to keep this number relatively low. The analysis of the sensitivity of the algorithm as a function of this parameter, as illustrated in Fig 2.8, shows that the performance of the algorithm stabilizes very rapidly when the number of orientations increases and there is essentially no performance improvement choosing more than 7 or 8 orientations. Therefore, in all our experiments we selected 10 uniformly spaced orientations.

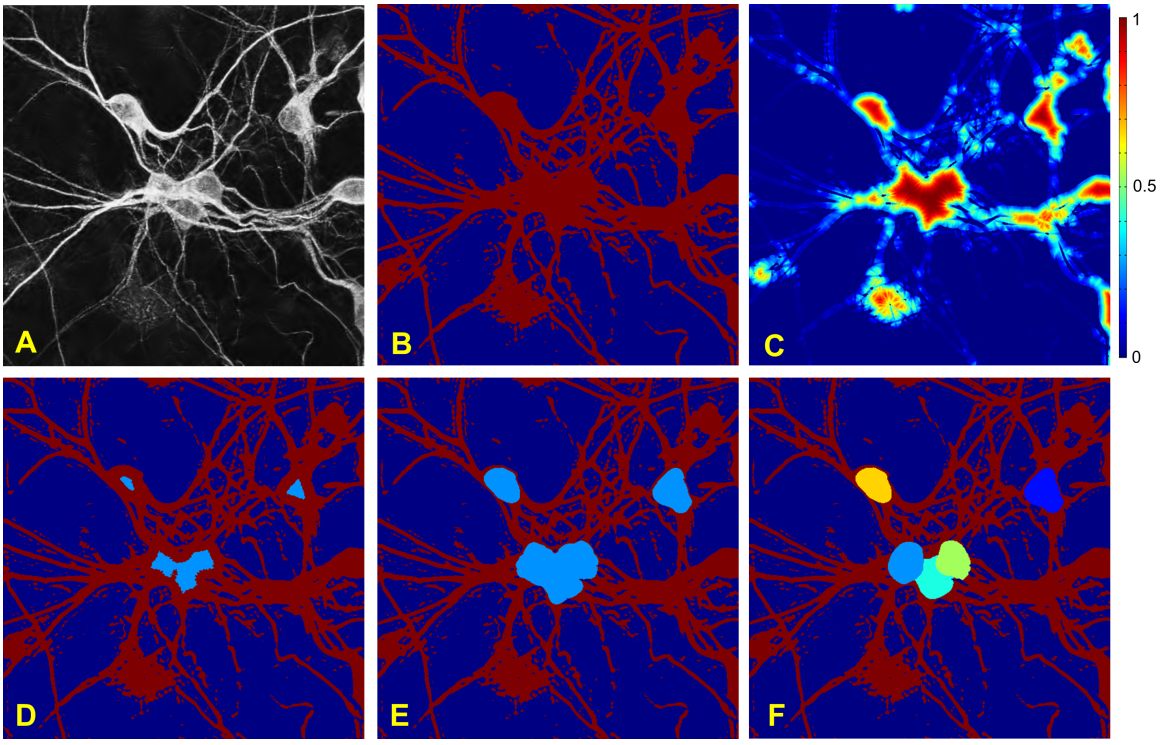

Figure 2.9: 2D soma detection and extraction. (**A**) Denoised image obtained using a shearlet-based routine on the MIP of the image stack. Image size = 512 × 512 pixels (1 pixel = 0.28 × 0.28μm). (**B**) Segmented binary image from SVM based segmentation. Segmented cell is shown in red while the background is shown as blue. (**C**) Directional Ratio plot; values are real numbers range between 1, in red color (most isotropic regions), and 0, in blue color (least isotropic regions); the Directional Ratio is only computed inside the segmented region, i.e., the red region in Panel B. Anisotropic Gaussian filters are used for directional filtering. (**D**) Detection of initial soma region obtained by thresholding values below 0.85 in Panel C. (**E**) Soma segmentation obtained by applying the Fast Marching method with the initialization curve determined by the boundary of the initial soma region in Panel D. (**F**) Separation of contiguous somas.

Figure 2.9 illustrates the application of our algorithm using anisotropic Gaussian filters (with default parameters $\sigma_x = 9$ and 10 orientations) on a representative MIP image of a neuronal culture of size 512×512 pixels containing five somas. The figure shows that the algorithm correctly detects somas and separates contiguous ones.

To benchmark the performance of our algorithm, we applied it to our 20 images and compared it against multiple state-of-the-art algorithms. For the comparison, we considered: the method based on Directional Ratio and level set by Ozcan et al. [42], henceforth denoted as DR-Lev; the algorithms based on conventional morphological operator by Vallotton et al. [51] and by Schmitz et al. [46], henceforth denoted as Morph1 and Morph2, respectively. For our algorithm, we consider two variants where the directional filters are rectangular or anisotropic Gaussian functions; henceforth we refer to these two versions of the algorithm as DR-Rec and DR-Gau, respectively. For Morph1 and Morph2, we need to set the value of the radius of the structuring element associated with the morphological opening operator. Based on the indication from the original papers, for two types of images considered in our tests, we set this value to 15 and 23 pixels, respectively. For DR-Rec, DR-Gau, DR-Lev, we used default scale parameter and number of orientations, as described above.

The performance of our algorithm and its baseline comparison with the various competing methods is reported in Table 2.3. The table lists the soma detection rate in terms of True Positive (TP), False Positive (FP) and False Negative(FN), the performance metrics FPR, TPR and DC for soma extraction, and the computing times for soma detection and extraction. The table shows that methods based on Directional Ratio provide overall very competitive performance for both soma detection and extraction. They have perfect or excellent detection rate and best extraction performance. By contrast, the soma detection and extraction performance of Morph1 and Morph2 is significantly lower; Morph2 performs better than Morph1. Concerning computing time, DR-Gau has the fastest computing time for soma detection, thanks

to the implementation of directional filters by separable convolution; computing time is about 4 times longer for all other methods, including DR-Rec, where filtering is implemented by regular 2D convolution. As Morph1 and Morph2 have no separate routines for detection and extraction, the reported computing times are the same. These methods exhibit the fastest computing time for extraction. Among methods based on Directional Ratio, DR-Gau is the fastest one, whereas Dr-Lev is about 40 times slower. As argued in the Materials and Methods section, this difference is due mostly to the significant difference in computational cost between the Level Set and Fast Marching routines. Results in the table show that, despite the huge difference in computing time, the use of either one of the two routines has negligible impact on soma extraction performance hence it is a major advantage to use the Fast Marching method in this algorithm.

Table 2.3: Performance analysis of different soma detection and extraction algorithms using 20 images containing 71 somas in total.

	DR-Gau	DR-Rec	Morph1	Morph2	DR-Lev
Detection rate (TP, FP, FN)	71, 0, 0	71, 0, 0	54, 6, 17	69, 16, 2	70, 0, 1
TPR (extraction)	0.95	0.99	0.83	0.95	0.95
FPR (extraction)	0.28	0.40	1.72	1.15	0.23
DC (extraction)	0.86	0.84	0.52	0.77	0.87
Comp. time: detect	0.21 s	0.86 s	0.90 s	0.90 s	0.86 s
Comp. time: detect+extract	4.01 s	4.45 s	0.90 s	0.90 s	157.76 s

The performance of each algorithm presented above depends on the combined performances of their segmentation and soma extraction subroutines. As observed in the Materials and Methods section, intensity thresholding is not expected to perform as effectively as our SVM-based segmentation routine but the former method is

significantly faster than the other one. Similarly, conventional morphological operators as those used in Morph2 to extract somas are faster than our method combining Directional Ratio and Fast Marching routine. To better illustrate the impact of each subroutine, we report in Table 2.4 the values of detection rate, DC and computing time obtained from various combinations of segmentation (SVM = SVM-based segmentation; Thr = intensity thresholding segmentation as in MORPH2) and soma extraction subroutines (DR = Directional Ratio and Fast Marching; Morph = morphological opening operator as in MORPH2).

Table 2.4: Computational cost for different combinations of image segmentation and soma extraction routines using 20 images containing 71 somas in total.

	SVM + DR	SVM + Morp	Thr + DR	Thr + Morp
Detection rate (TP, FP, FN)	71, 0, 0	69, 6, 2	67, 0, 4	69, 16, 2
DC (extraction)	0.86	0.77	0.81	0.77
Comp. time	8.21 s	8.9 s	0.91 s	1.6 s

The table shows that our Directional Ratio routine for soma extraction consistently improves the algorithm performance. If the segmentation routine is implemented using our SVM-based segmentation, then a better DC value (the best overall) and a faster computing time is achieved using Directional Ratio rather than morphological opening operator. Similarly, if the segmentation routine is implemented using the morphological opening operator, again a better DC value and a faster computing time (the best overall) is achieved using the Directional Ratio.

Another observation is that the SVM-based segmentation routine has the largest impact in the overall computing time of the algorithm. By replacing this routine with a method based on intensity thresholding, the total computing time decreases from

8.21 to 0.91 s. However, this comes with a significant downgrade in performance as DC decreases from 0.86 to 0.81 and the detection rate worsens (4 somas are missed). This observation is consistent with the segmentation result shown in Fig 2.1 using a representative MIP image of a neuronal culture. The figure shows that the segmentation result based on intensity thresholding may miss regions inside a soma or produce over-segmented images leading to false positive or false positives, as is in fact observed in Table 2.3.

2.5.3 3D soma analysis

We tested our 3D algorithm for soma detection and extraction on two sets containing different types of imaging data: Set1 consists of three confocal image stacks of brain tissue, each stack containing 1-2 somas; Set2 consists of 3 confocal image stacks of neural cultures, each stack containing 7-8 somas.

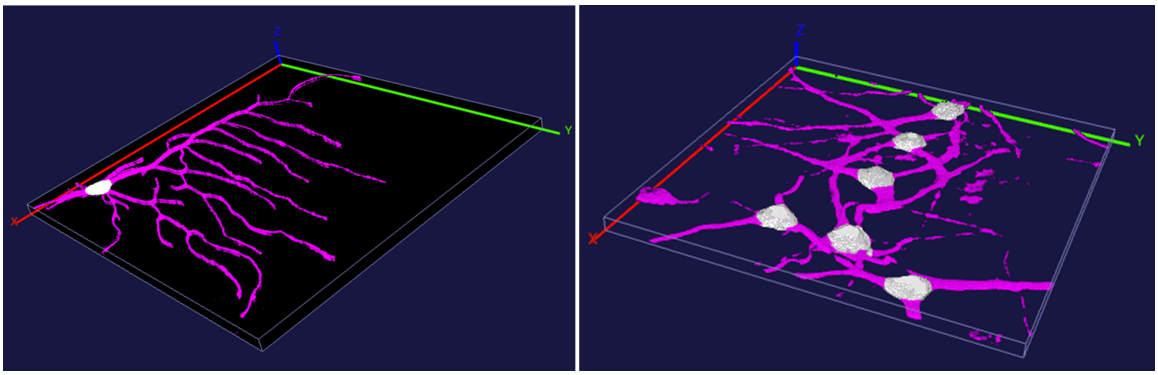

Figure 2.10: 3D soma extraction. Soma detection and extraction using our algorithm based on Directional Ratio and anisotropic Gaussian filters of a confocal image stack of a brain tissue (left) and a neuronal culture (right). Image size = 1024×1024 pixels (1 pixel = $0.28 \times 0.28 \mu m$).

Computational cost is a major issue for the analysis of 3D data due to the impact of 3D filtering. As discussed above, the application of our method based on Directional Ratio to the analysis of 3D data using conventional 3D filters would be highly impractical; on an image of size $512 \times 512 \times 512$, using 40 orientations, it would take over 100 hours. Therefore, in our tests of our algorithm, we only considered anisotropic Gaussian directional filters which are implemented via separable convolution and not rectangular filters. We used 40 orientations for our experiments.

For the soma extraction process we applied *Fast Marching method* instead of *Level Set method*. This modification changes the computational cost of method enormuously especially in multi-soma data. Table 2.5 and Table 2.6 shows comparison of two soma extraction methods on both single and multi soma images. In a single soma image soma segmentation cost reduces from 30 seconds to 2.8 seconds when the level set method is switched to Fast Marching method. In case the image has seven somas cost reduces from 390 seconds to almost 7 seconds, Table 2.5. Although the comparison bases only on two representative images, these results shows that the save from computational effort could be huge occasionally with the Fast Marching method. Fig. 2.10 illustrates the application of our algorithm for 3D soma extraction on representative image stacks.

Table 2.5: Comparison of level set and Fast Marching methods on a single soma image

method	DC	Time
Level Set Method	.92	30 seconds
Fast Marching Method	.90	2.8 seconds

Table 2.6: Comparison of level set and Fast Marching methods on multi soma(7 somas) image

method	DC	Time
Level Set Method	.85	390.81 seconds
Fast Marching Method	.84	7.65 seconds

To assess the performance of our approach, we employed the same metrics we adopted in 2D setting. As baseline comparison, we implemented a 3D extension of the method based on the morphological opening operator proposed by [46] (the method is only applied in 2D in the original paper). To adapt the method to the different geometry of the dataset, for the structuring element we used either a sphere, when processing Set1, or a cylinder, when processing Set2. We remark that using a sphere on Set2 would produce very poor results due to the small number of pixels available along the z direction. On the other hand, our method using Directional Ratio requires no *ad hoc* modifications for the two data sets.

The performance results of our method, denoted as 3D-DR, and the method based on the morphological opening operator, denoted as 3D-Mo are reported in Table 2.7.

Table 2.7: Performance analysis of our 3D soma detection and extraction algorithms (3D-DR) and 3D-Mo, a method based on morphological operators. Results in the table are averages from image stacks of brain tissue (Set 1 = 3 stacks, 3 somas) and neuronal cultures (Set 2 = 3 stacks, 16 somas).

	3D-DR Sets 1+2	3D-Mo Sets 1+2	3D-DR Set 1	3D-Mo Set 1	3D-DR Set 2	3D-Mo Set 2
Detection rate	100%	100%	100%	100%	100%	100%
TPR (extraction)	0.91	0.89	0.93	0.95	0.89	0.83
FPR (extraction)	0.21	0.20	0.11	0.17	0.18	0.24
DC (extraction)	0.89	0.86	0.90	0.90	0.88	0.81
Comput. time	78 s	20 s	106 s	13 s	35 s	5 s

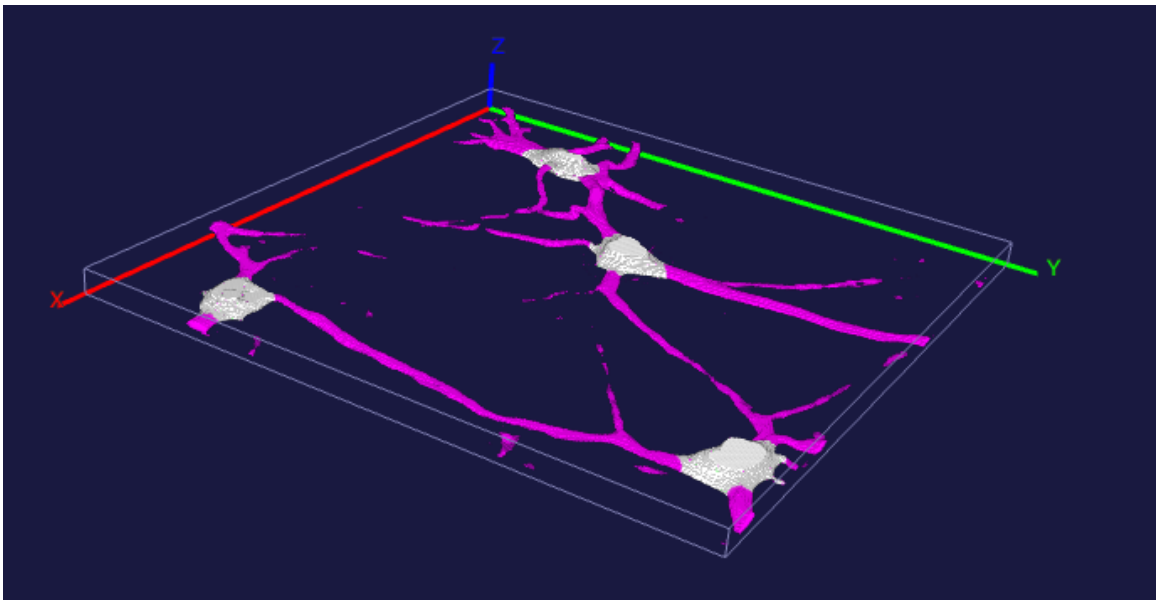

Figure 2.11: 3D soma extraction. Soma detection and extraction using our algorithm based on Directional Ratio and anisotropic Gaussian filters of a confocal image stack of a neuronal culture. Image size = 1024×1024 pixels (1 pixel = $0.28 \times 0.28 \mu m$).

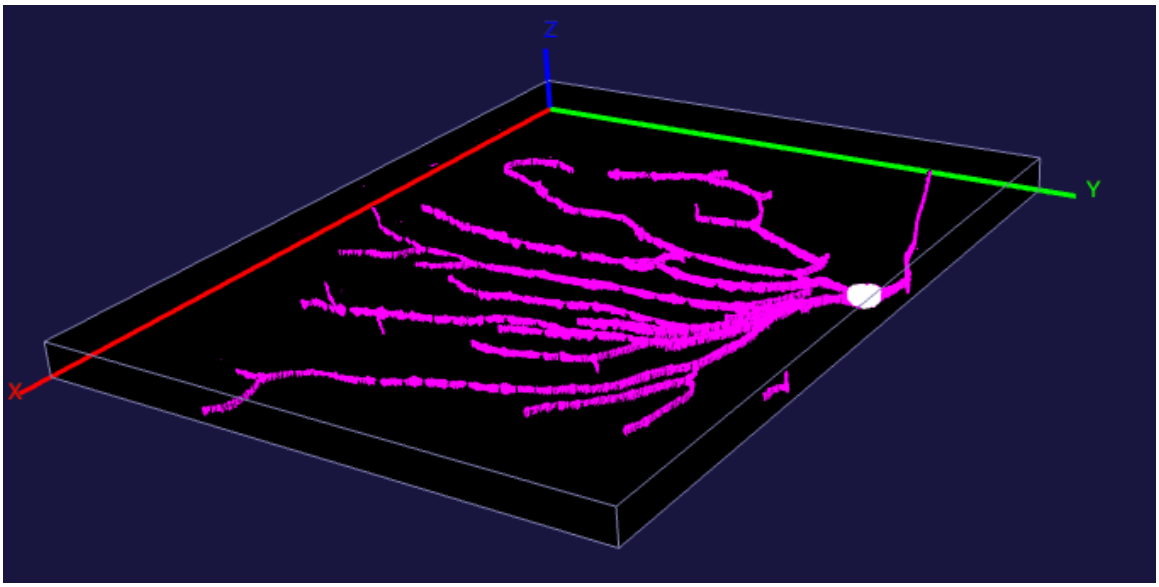

Figure 2.12: 3D soma extraction. Soma detection and extraction using our algorithm based on Directional Ratio and anisotropic Gaussian filters of a confocal image stack of a brain tissue. Image size = 1024×1024 pixels (1 pixel = $0.28 \times 0.28 \mu m$).

Table 2.7 shows that 3D-RD, our method based on Directional Ratio, provides overall a modest performance improvement with respect to 3D-Mo (DC: 0.89 vs. 0.86) at the expense of a higher computational cost (78 vs. 20 s). However, a closer examination of the results shows that 3D-DR exhibits a rather significant better performance with respect to 3D-Mo when the analysis is restricted to the image stacks of neuronal cultures (DC: 0.88 vs. 0.81). As observed above, confocal image stacks of neuronal cultures are more challenging to process since they contain a relatively small number of images, typically about 10-20 images, so that only 10-20 pixels is available along the z-direction. Unlike conventional morphological operators, the method based on Directional Ratio performs in this situation about as efficiently as for the other data. Even though, processing time is faster for 3D-Mo, computing

time is very reasonable for 3D-DR.

CHAPTER 3

Automated Extraction of Neuronal Trees

Digital reconstruction of the graph connectivity of neuronal structures in fluorescence microscopy images is an especially critical task as this reconstruction provides the backbone for building a geometrical representation of a neuron. A major effort in improving the tracing and reconstruction of dendritic arbors emerged during the last decade in response to the DIADEM Challenge [35] and again more recently as part of the ambitious BigNeuron project [44]. As a result of these initiatives, several powerful algorithms were proposed delivering robust and accurate neuronal tracing. Those methods can be categorized in two classes as *local tracing methods* and *global tracing methods* [2]. Global tracing approaches mostly base on skeletonization.

Therefore trace does not start from cell body but instead starts from "key points" detected by algorithm which are not necessarily cell body [56]. There may be several key points and at the end all traces are combined to have the complete skeleton of input. Hence, traces from all cell bodies are united as one, and this trace does not have any information regarding cell bodies. Although there are successful global tracing methods, they are not useful for our studies since these methods does not present the tree structures of cells. In contrast, local tracing methods start from cell body, trace through each neurites [10] and they are capable of satisfying the tree structure of neurons. However existing methods in that categorization, are poor on tracing through intersecting branches. This is a critical drawback of those methods especially in analysis of neural cultures. As such data typically consist of stacks containing 10-25 images (only 10-25 pixels are available along the z-direction, as compared with the x and y directions where length can be 512 pixels or more), the 'data cube' is very thin along one of its axes and, as a result, it is very inefficient or even impossible to process these imaging data as true volumes. Instead, maximum projection is applied and analysis of these data types done in 2D projected images which causes many intersecting branches. Therefore a method which is unable to discriminate intersecting branches is not suitable for analysis of neuronal cultures. As a result, despite of many studies based on various methods, there is no efficient method to extract tree structure of neural cultures accurately.

In order to overcome this need for analysis of neural cultures, we proposed a method which detects the neurites and extracts the tree structure with high accuracy. Our method can be categorized in local tracing approaches, which is not necessarily

attempting to fully trace every neurite in an image. Instead, starting from the soma locations, that we detected in the preprocessing step, our method aims to extract the trees associated to each neuron in the image by computing front-propagated traces. To carry out this task, the major challenge, as it is said before, is to resolve crossing and/or partially overlapping neurites. For that purpose algorithm tries to keep the orientation of corresponding neurite as stable as possible throughout the propagation, which is a reasonable assumption based on the natural images from neuronal cultures. Method is tailored to the previous project and it is fully automated.It is designed to work on 2D binary (segmented) images with segmented soma regions predefined. Our numerical experiments show that the method handles with intersecting branches efficiently with the help of a novel well designed seed search process. Performance of the method highly depends on the accurate segmentation of input. The algorithm then proceeds as follows.

3.1 Method

3.1.1 Seeding

We determine seeding points along the centerlines of the neurites by finding those points inside a neurite that are the farthest from its boundary with an adaptation of [26, 24].

For $x \in \mathbb{R}^2$, we define $Df(x) = \min_y\{ \|x - y\| : f(y) = 0\}$, where f is a binary segmented image. The local maxima of Df inside the structure are the points that

are furthest away from the boundary of the neuron, since $f(y) = 0$ if y belongs to the image background. To enhance the magnitude of those local maxima, the function Df is next convolved with the filter $\begin{pmatrix} -\frac{1}{8} & -\frac{1}{8} & -\frac{1}{8} \\ -\frac{1}{8} & 2 & -\frac{1}{8} \\ -\frac{1}{8} & -\frac{1}{8} & -\frac{1}{8} \end{pmatrix}$. After this step, we use a thresholding filter to select candidate seeding points along the centerline. Clearly, the lower the threshold, the more the seed points we derive. However, if the threshold value is too small, one may find more than one seed along the centerline resulting in irregular or inaccurate traces. On the other hand, if the threshold value is too large, then seeds may be very sparse and the distance between consecutive seeds might be so large that the tracing routine connecting potential seed points may terminate earlier than expected. We remark that the selection of the 'best' threshold value is dependent on the thickness and tortuosity of the neurite, therefore it is very difficult to determine this value automatically. Therefore, after seeds are generated using a reasonable threshold value (we set the value 0.16 in our experiments), we proceed as follows. For each generated seed s, we compute a ball centered at s with radius $Df(s)$ and eliminate all other seeds found within this ball. If this process generates gaps along the centerline (since balls associated with different seed points do not intersect), then we generate additional seeds by computing again the distance function, Df, locally within that gap region. Within this region, we proceed as above by eliminating candidate seeds potentially accumulated near the same centerline location. As shown in [26, 24], this method is very reliable and efficient in generating centerline traces.

3.1.2 Initialization

We applied successive dilation operators with rates $r = 1.1, 1.2, 1.3$ on each soma mask (Figure 3.1, panel B). Let us denote by S_0 the soma mask and by S_i, $i = 1, 2, 3$, the three dilated masks, ordered by increasing size. Next take the symmetric differences $S_1 \triangle S_0$ and $S_3 \triangle S_2$ and its intersection with the segmented structure. For each neurite, this operation will identify two short neuritic segments in the proximity of the soma (Figure 3.1, panel C). Next we find the centroids of these regions and connect each one of them to its nearest S_i only proximal. Thus we find the starting location of each neurite and its initial orientation, which is given by the line connecting the centroids (Figure 3.1, panel D). The closest seed point to the centroid of $S_1 \triangle S_0$ is chosen as the starting neurite seed. However, tracing should start as close as possible to the corresponding soma. Therefore another line segment is drawn from the neurite starting seed to the soma on opposite direction of the neurite. The intersection of the line segment and soma is determined as the starting point of the tracing along the neurite.

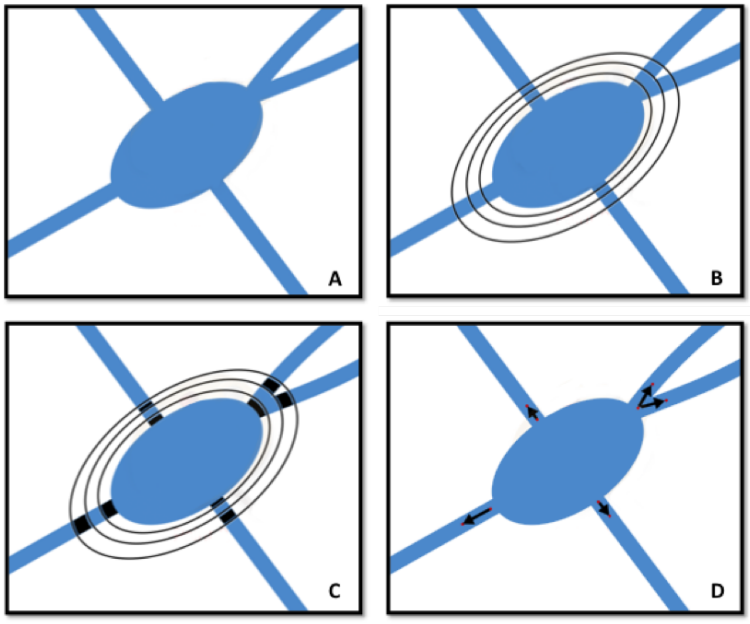

Figure 3.1: Initialization. (A) Idealized model of a binary segmented soma with three emanating neurites, **(B)** Soma boundary is dilated three times with increasing dilation factor and **(C)** the intersection of the symmetric difference of the successive masks with the structure produces the black regions shown in the panel. **(D)** By computing the centroids of the six black regions from panel C and connecting the centroids located on the same branches to the soma regions, we find the starting location of each neurite and its initial orientation, as indicated by the black arrows.

3.1.3 Tracing

Starting from the initial location of a neurite found in Step (i), the algorithm searches for the closest seed location within a small search window which gives proximity to points in the direction of initial direction of neurite. It then connects the two seed points. This process is repeated after each new seed is connected to the trace and it stops when no more seed is located within each regarding window.

The search window plays a key role in this task, because it determines that the

tracing continues on the same branch. When branches intersect in a maximum intensity projected image, the risk of switching to another branch becomes significant. The process for choosing where to continue is illustrated in panels A and B of Figure 3.2. The main idea is that neuritic processes change orientation in a smooth way. Therefore abrupt changes of orientation of the tracing process are likely to lead a turn into a different branch. Hence, first the algorithm searches for the next seed within a long rectangular region whose long side is aligned with the expected orientation of neurite (Figure 3.2, panel A). For the initial location of each neurite, such orientation is estimated according to Step (i); for successive locations, the expected orientation of the neurite is estimated by measuring the direction of the two preceding seeds in the trace. If no seeds are found within this rectangle, then a pair of rectangular windows are generated with orientations forming a small angle with respect to the expected orientation of the neurite (Figure 3.2, panel B). The length of those new rectangles is slightly smaller than the previous rectangular region. This process continues (Figure 3.2, panel C) until either a seed is found or the orientation of the new rectangles exceeds a given angle ($4\pi/5$ in our experiments). This searching process is repeated multiple times generating every time a new approximate circular sector region of larger radius. That is, every time the searching process is repeated, longer rectangles are used to generate the new window region (Figure 3.2, panel D). If this process does not find a new seed after a number of attempts, we terminate the search and assume that the neurite is completely traced. This tracing routine is illustrated in Figure 3.2. In our numerical experiments we set the length of the initial rectangle to be 10 pixels. The searching process is repeated up to 10 times,

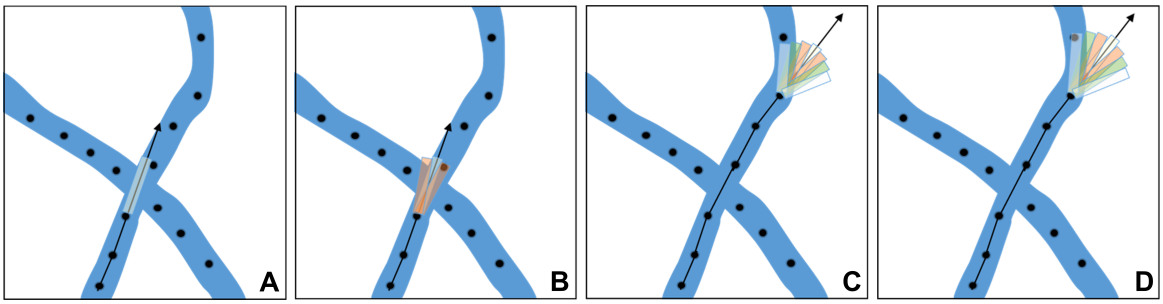

Figure 3.2: Tree tracing. (**A**) The search for the next node in the trace is initially restricted within a long rectangle whose main axis is oriented according to the local orientation of the neurite (black arrow). (**B**) If no seed is found, two additional rectangles are generated with orientations forming a small angle with respect to the local orientation of the neurite. This process ensures that the trace follows the given neurite and not the intersecting one. (**C-D**) If no seed is found within the approximately circular sector region, the search is repeated over a larger region obtained by increasing the length of the rectangular windows.

every time increasing the length by 2 pixels. In our algorithm we choose the radius of the circular sector to be 30 and the central angle to be equal to $4\pi/5$.

3.2 Computation of Fluorescent Intensity Profiles

The trace extracted in the previous section provides a spatial reference system to compute the local fluorescent intensity signal along each neurite. As the background intensity of a fluorescent image is typically non-zero, this background value needs to be subtracted in order to get a reliable measure of the fluorescent signal along a neurite. Furthermore, this value varies spatially. Therefore, to estimate its local value at a location near a neurite, we average the background signal computed on a pair of small windows (3×3 pixels) centered on a segment perpendicular to the neurite trace and displaced slightly away from the neurite (2 pixels away in our

experiments) as shown in Figure 3.5. This estimated value is subtracted from the fluorescent intensity value computed at the neurite location and the difference is the fluorescent intensity value at the particular location.

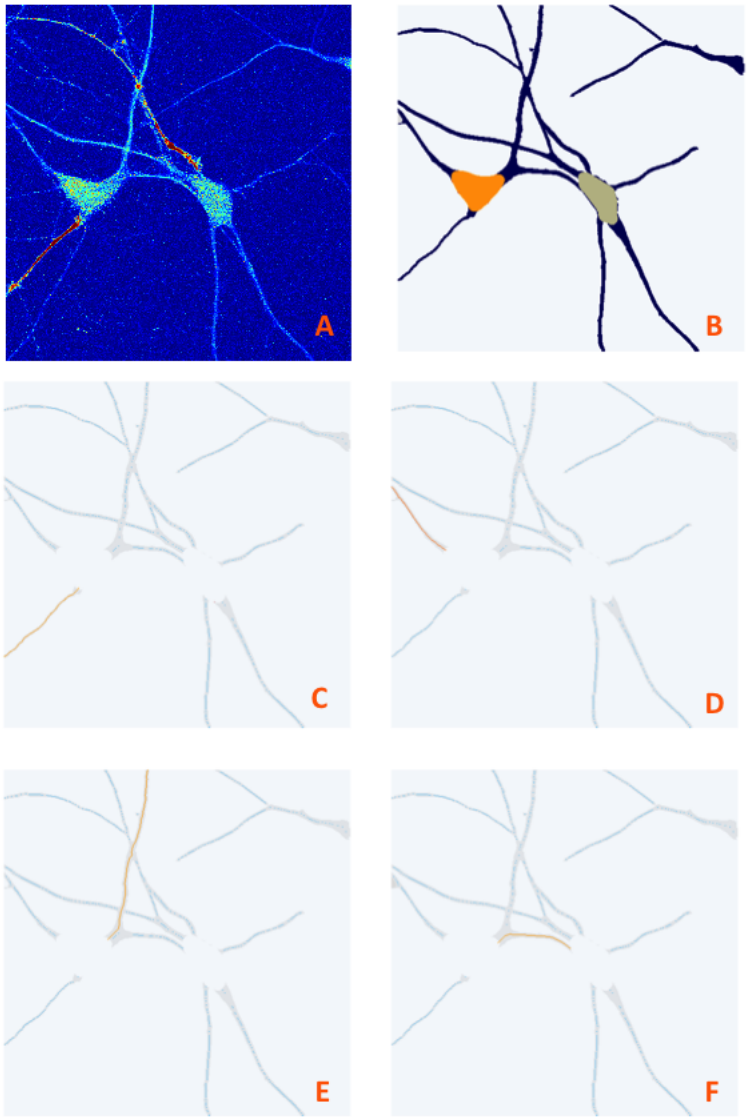

Figure 3.3: Tracing of neurites part 1. Algorithm requires segmented image with each soma regions determined. For each soma firstly neurites are detected and each of the neurites is traced individually. **(A):** denoised image, **(A):** overlap of segmented Image with segmented soma regions, **(C-F):** traces for first soma. Image size = 512×512 pixels (1 pixel = $0.28 \times 0.28 \mu m$).

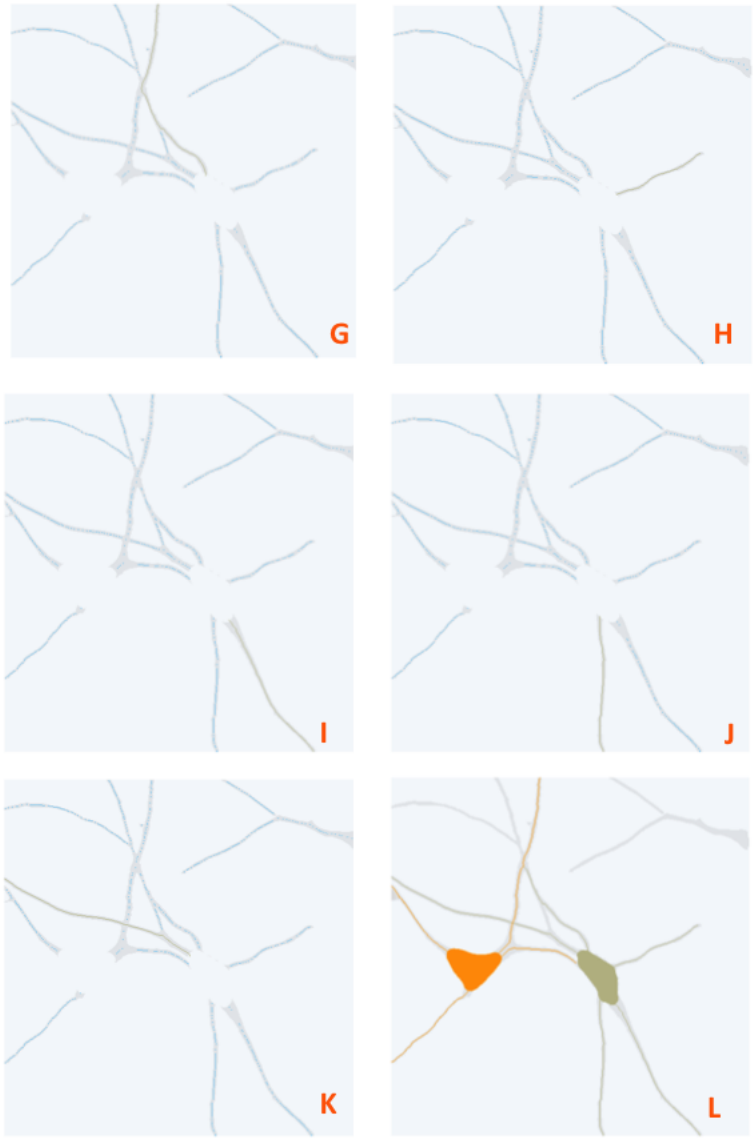

Figure 3.4: Tracing of neurites part 2. Process continues until all detected neurites for each somas are traced. **(G-K):** traces for soma 2, **(L):** Result shows the somas with the corresponding traces. Image size = 512×512 pixels (1 pixel = $0.28 \times 0.28 \mu m$).

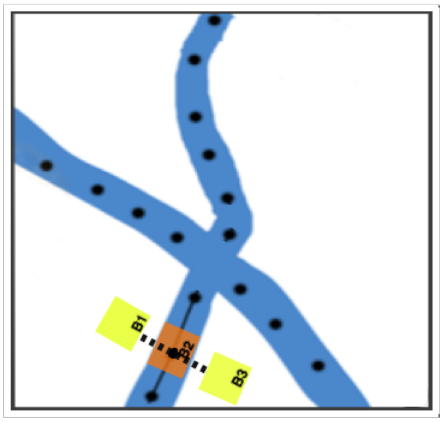

Figure 3.5: Collecting fluorescent intensity. Three rectangles, which lie on a line perpendicular to direction of neurite, are generated to estimate the correct intensity at a seed point. Two (yellow) rectangles B1 and B3, 2 pixels away from the boundary of neurite, are to estimate average intensity of background, orange rectangle B2 is to estimate average intensity inside neurite. Difference between estimated average intensity of background and the average intensity of neurite will be assigned as the corrected intensity value for seed.

3.3 Results and Discussion

Accurate and efficient neurite tracing is one of the foremost challenges in neuroscience. Although existing local tracing methods are not able to handle the intersecting neurites problem, our propagation algorithm is remarkably successful to handle with such issues as it is seen in Figure 3.6.

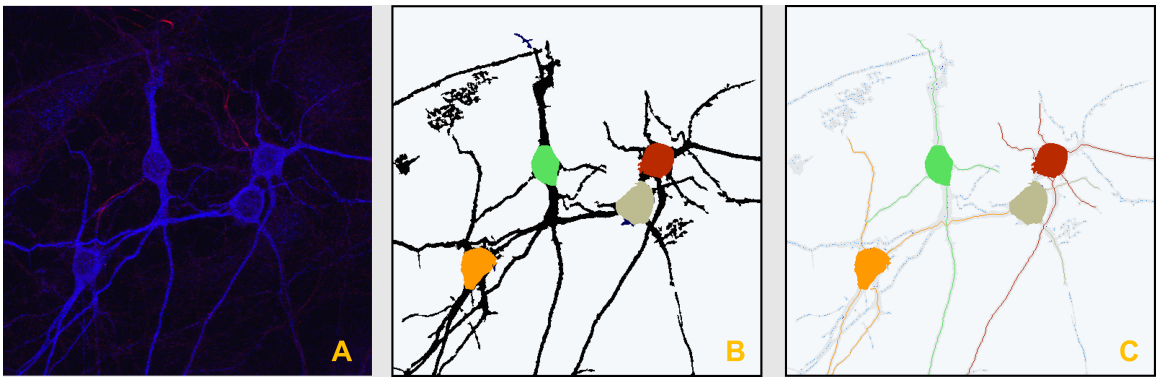

Figure 3.6: Tracing of neurites on multi cell image. Automated tree structure extraction is illustrated on multi soma image. Image size = 512×512 pixels (1 pixel = $0.28 \times 0.28 \mu m$).

Our method is proved to be accurate in the detection of neurites in the numerical tests. The test are done on 10 images from neural cultures. Those images contain 2-8 somas, which yields lots of intersecting neurites with a few clustered somas, hence our dataset is a realistic choice to test our method. The performance, is given in Table 3.1 which shows that overall neurite detection accuracy is sufficiently high. For the validation, detected neurites with accurate or inaccurate traces, are accepted as *true positive*. Branches which are traced although they are not a dendrite or axon are accepted as *false positive*. Missed branches which are actually neurites are accepted as *false negative* . Our method on 10 images with more than 190 neurites had no false positive detection and 90% of the neurites detected accurately.

Table 3.1: Overall performance

number of somas	number of neurites	number of detected neurites	accuracy percentage
49	191	171	0.90

One major problem is high dependence of our method on the image segmentation quality. Even one incorrectly segmented pixel can effect the performance of the result. We have seen this effect considerable especially on node of the intersecting neurites. These nodes are relatively large areas and show features of blob-like but not vessel-like structure. Therefore our seeding algorithm can not generate seeds along a centerline of neurites in those nodes but it generates one seed at centers of each node. At these situations, because of the missing seeds, traces are highly dependent on the image segmentation even though our trace algorithm tries to minimize these artifacts. It is because the location of that one seed, at the center of node, is determined by the boundary of neurites around the seed. When the boundary is shifted one pixel, it will cause this seed, at the center of node, to shift one pixel as well, which would affect the trace dramatically. Another problem is the neurites with irregular widths. If a neurite gets very thin at some region, seeds will be distributed with large intervals for that part. This situation creates a large region without any seed, which causes trace to stop earlier than expected. In order to overcome these drawbacks, we may modify our seeding algorithm to generate seeds along the major axes of such nodes or introduce a user intervention to force trace continue when it stops unexpectedly. These possible issues are illustrated in synthetic image in Figure 3.7.

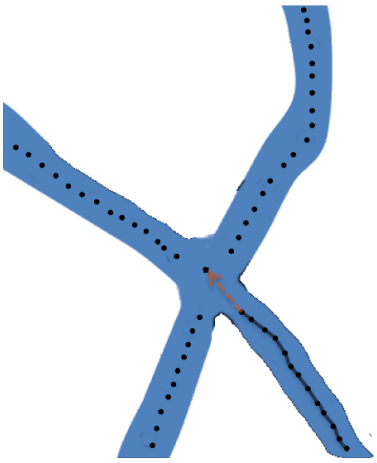

Figure 3.7: Tracing of intersecting neurites. At the intersection of neurites, only one seed is generated at the center of intersecting region. Since the distance which is shown in orange dashed arrow is large, trace can not continue and stops at the start point of orange arrow.

During the trace, in each step of the propagation, whole image is processed with computationally expensive calculations such as convolution. Hence cost of method increases with the size of input. In the large images, although the accuracy of the method is not affected, the cost is insuperable. In order to reduce the cost for such cases, we partitioned the large size inputs into small subregions, which are determined by user, and processed them in parallel loops. After the traces of all subregions are completed, they are collected and unified. In Figure 3.8 and Figure 3.9 two subregions from a large image (with over 40 cells) are shown which are processed by our algorithm.

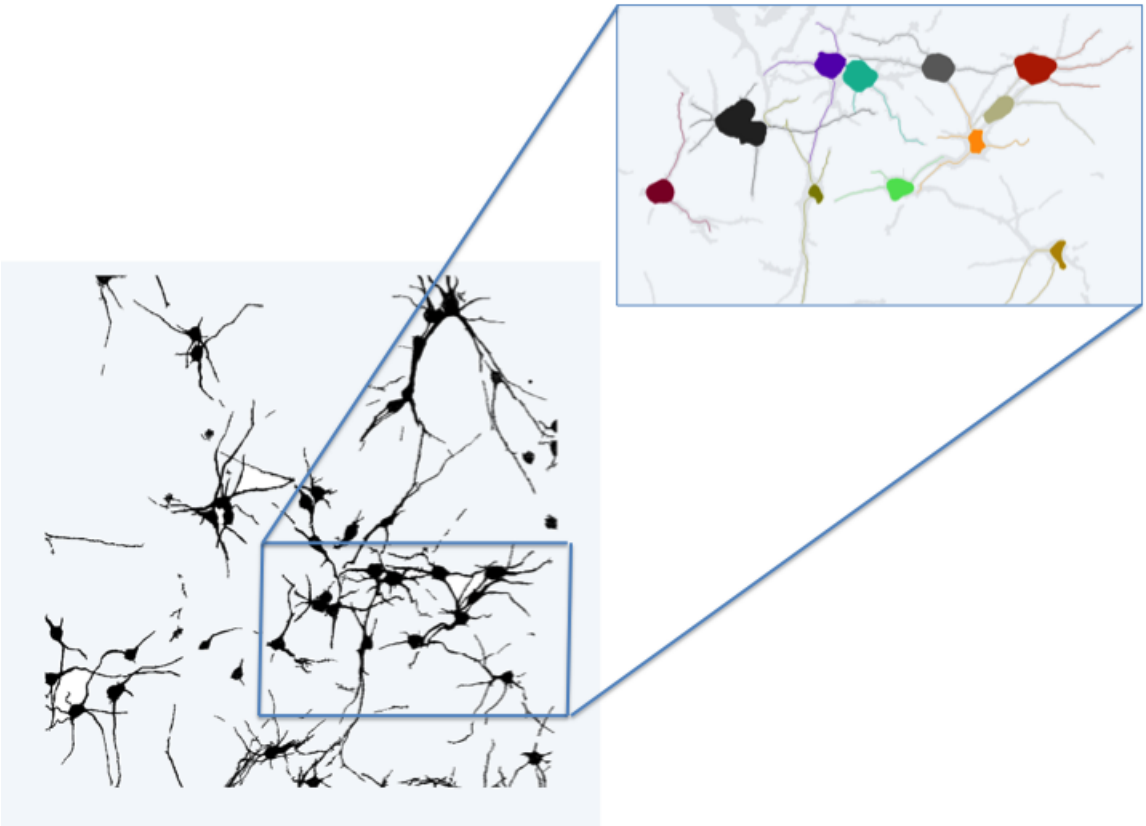

Figure 3.8: Tracing of neurites in large images part 1. Application of the method on image with large number of cells. The method is applied on manually chosen subregions in order to save from computational cost. Each cell and their corresponding neurite tracings are given in different colors. Image size = 1894 × 1894 pixels (1 pixel = 0.28 × 0.28μm).

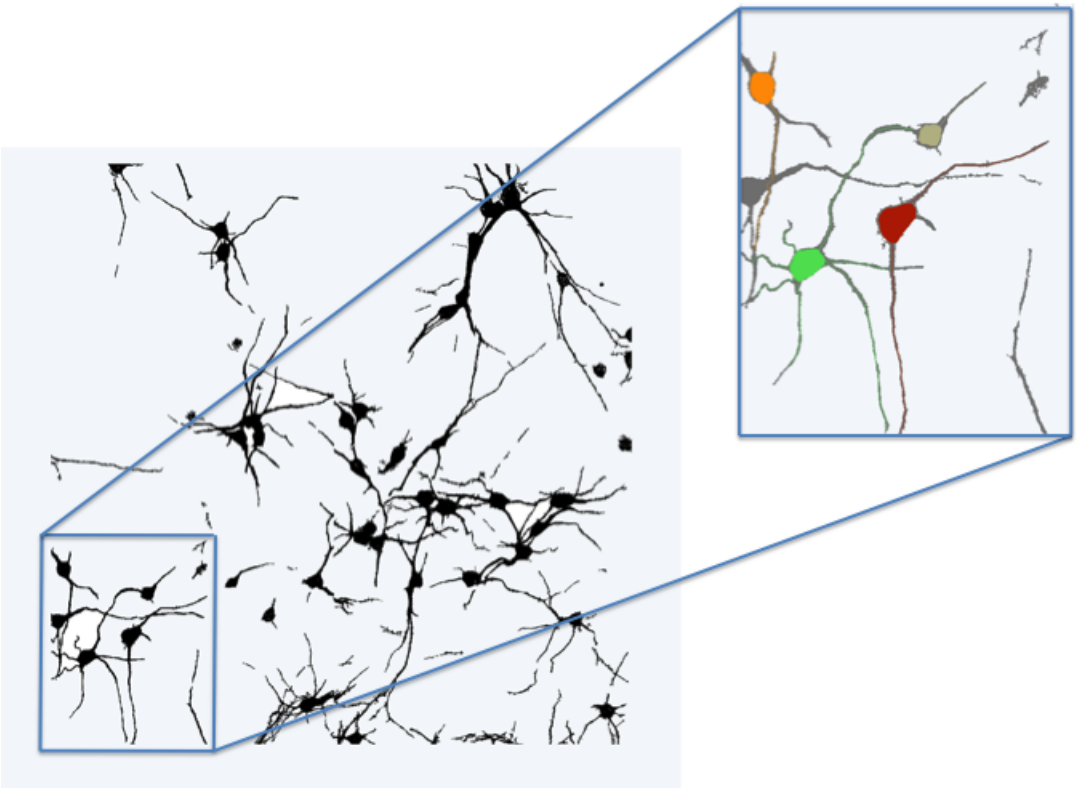

Figure 3.9: Tracing of neurites in large images part 2. Application of the method on input with large number of cells. The method is applied on manually chosen subregions in order to save from computational cost. Each cell and their corresponding neurite tracings are given in different colors.Image size = 1894×1894 pixels (1 pixel = $0.28 \times 0.28 \mu m$).

CHAPTER 4

Neuronal Classification

This part of my work focused on developing an algorithm to infer molecular mechanism relevant in models of neural plasticity in brain circuits.

Neuronal plasticity is the ability of neurons to modify output signal and signaling pathways according to external stimuli. That adaptation is done through varying distribution and composition of some specialized molecular complexes in the axonal compartment and dendritic spines of neuron [10]. Although there is a significant progress in revealing mechanism of neural plasticity in the dendritic spines, plasticity at the axon initial segment (AIS) is still not well understood. AIS is the membrane component of axon of neurons where the action potentials are initiated and it plays

a critical role in neuronal communication. Voltage-gated sodium (Nav) channels in particular are fundamental constituents of the AIS. They form a large macromolecular complex with scaffolding proteins, such as ankyrin, spectrin, that regulate AIS plasticity, and fibroblast growth factors that are critically important in regulation, localization and activity [7]. The distribution and composition of AIS proteins are determined by kinases, but the explicit relation between kinase signaling and the AIS proteins is still not fully understood. Besides there is no quantitative tools to analyze correlation between the composition of those proteins and the cell morphology, neuronal firing and eventually synaptic communication. To address this knowledge gap, our goal was to establish what perturbations associated with kinase inhibitors are explanatory variables of changes in cell network architecture and expression levels of macromolecular complexes of the AIS. In order to correlate the kinase inhibitors to composition of AIS proteins we analysed expression levels of macromolecular components through dendritic component and AIS. By applying our method to images of neural cultures exposed with different kinase inhibitors, we measured the expression levels along AIS and dendrites together with the morphological features of somas were collected. The major problem to address is the variability in the measurements due to cellular noise and noise caused by acquisition method. Although data were denoised and measurements were normalized to eliminate the misleading effects of data acquisition methods and natural noise, we observed variability among classes possibly caused by the feature extraction step. In order to eliminate irrelevant information from the features we applied shift-invatiant wavelet transform to the features which lead to a significant benefit in data classification. We applied support vector

machines classification to data. Our results revealed how perturbations associated with kinase inhibitors are correlated with the expression levels of macromolecular complexes in AIS. Our neuroscientist collaborators were particularly interested to discover that some protein pairs are more sensitive to the perturbations than expected, leading to important implications for the formulation of the biological model [4].

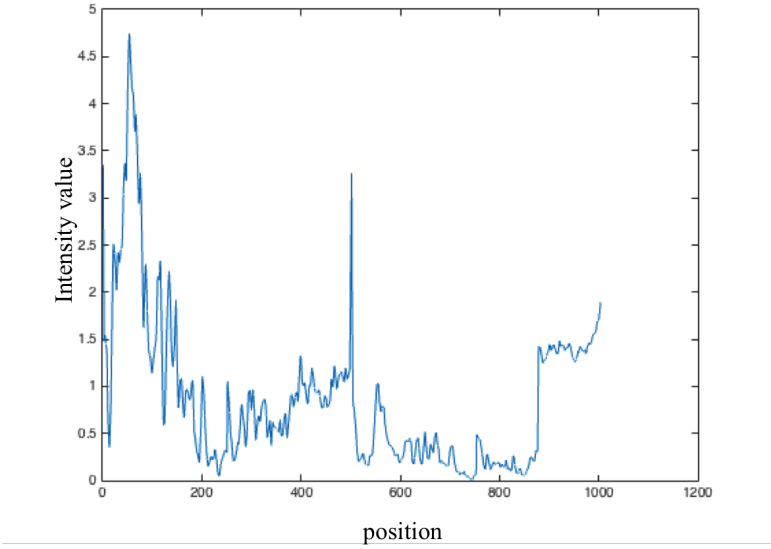

Figure 4.1: A raw feature vector sample. Example of a raw feature vector. Feature vectors for each sample is 1004 pixels as shown in the horizontal axis. Blue plot shows the intensity values collected from the fluorescent images. Intensities are collected from neurites of each cell both from red and green color channels.

4.1 Features: Translation Invariant Wavelet Transform Coefficients

Wavelet transform is frequently used as a feature extraction method because of its useful properties. However traditional wavelet transform is not sufficient to eliminate some artifacts caused by preprocessing steps. Therefore we employed the translation-invariant wavelet transform to modify feature vectors and minimize the errors caused by the feature extraction process.

The mechanism of translation invariant wavelet transform is slightly different than the usual wavelet transform. Other than the input, circular shift of the input is also used during the transformation, and coefficients from both of them are concatenated at the end, where circular right shift of a vector is basically shifting all elements to right by one step and moving the last input of vector to the first place. On each layer low pass and high pass filters are applied to each of the input and circular right shifted input. On each layer after filtered data coefficients are downsampled. At the end of each layer, coefficients of high pass filtered data, form both raw input and circular right shift of the input, are concatenated as the coefficients of wavelet transform as the corresponding layer, where first layer as assumed to be the raw input as seen in Figure 4.2 . These coefficients are separated and the low pass filtered data, from both raw input and circular right shift of the input, is used as input for the next layer. This process is repeated for each layer until the last layer. In the last layer, as previously done, high pass filtered data coefficients are concatenated as wavelet coefficients of last layer, while the low pass filtered data coefficients are concatenated

as the first layer wavelet coefficients. The illustration of the transformation is given in Figure 4.2. For the implementation of translation invariant wavelet transform we used Wavelab850 [15].

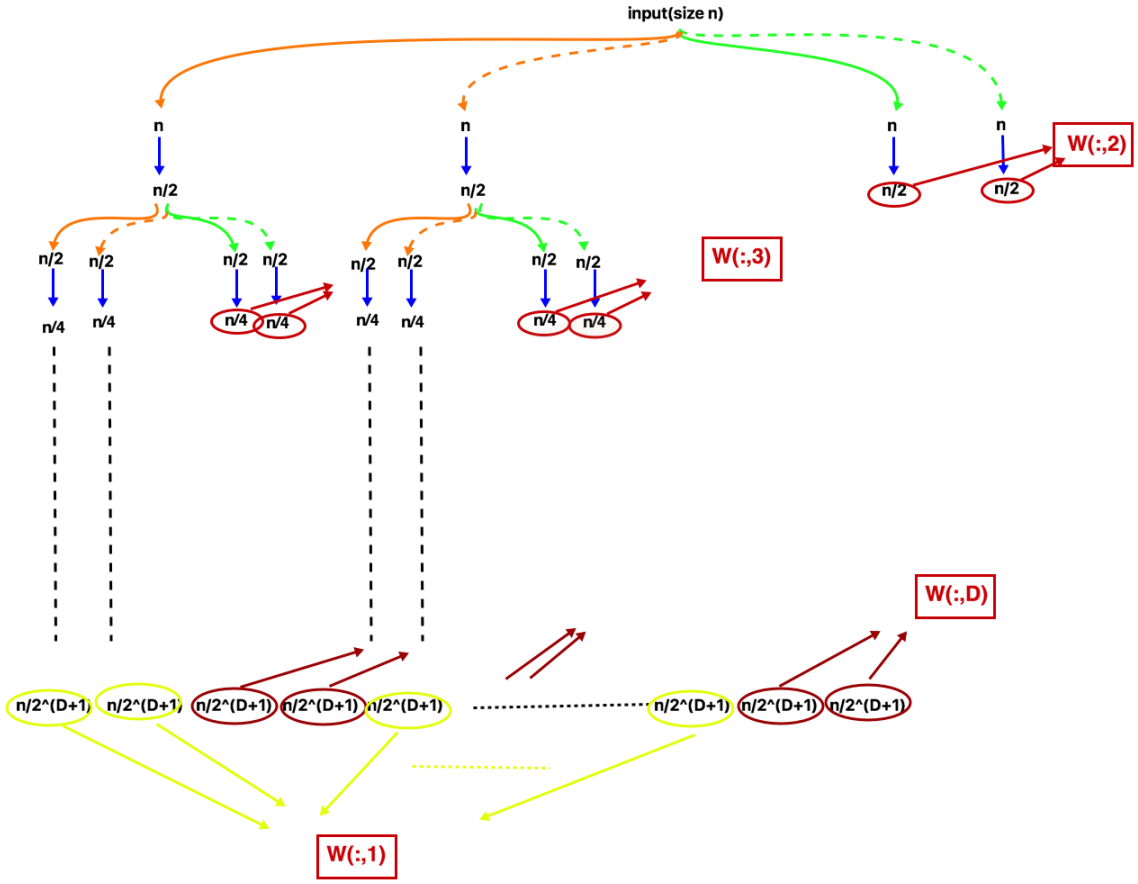

Figure 4.2: Translation invariant wavelet transform scheme. *orange arrows:* low pass filter convolution, *orange dashed arrows:* low pass filter convolution with the circular right shifted input, *green arrows:* high pass filter convolution, *green dashed arrows:* high pass filter convolution with the circular right shifted input, *blue arrows:* downsampling, $W(:,m)$ *:* m th order coefficients of the translation invariant wavelet transform of input

4.2 Support Vector Machine

Support vector machine (SVM) is a supervised pattern recognition technique. The main idea of SVM is generating a linear subspace which separates the samples of two different class with maximum margin. Support vectors are the data points which are closest to the separating subspace. When any sample other than support vectors is removed, the separating subspace does not change location, however removing any of the support vectors would affect the location of subspace directly. Since this separating subspace is determined by the support vectors, method is called as Support Vector Machine. After the support vectors are determined, the separating subspace is located to keep the margin, which is the minimum distance from the support vectors to subspace, maximum. Linear SVM aims to find a linear separator, but if data is not linearly separable then non-linear SVM is employed. In nonlinear SVM chosen kernel function maps the data to an infinite dimensional Hilbert Space and data is assumed to be linearly separable in that space. After that, as happened in Linear SVM, data is separated by linear hyperplane. SVM is a model based method. It generates a model according to given training data, labels and kernel function, and classifies the samples according to the generated model. In origin, SVM is a two-class method. In order to use SVM for multi class cases a modification is necessary. In our case, we used one-against-all SVM method. This method creates a binary-class problem for each of the classes. In each time it separates samples of corresponding class from all other samples associated with other classes.

For our numerical tests, we applied multi-class non-linear SVM. As a kernel we

used radial basis function. All SVM parameters are determined through grid search and density of radial basis function is fixed to 0.0017, the cost parameter is fixed to $C = 7.5938$. libsvm Matlab toolbox is used during the numerical tests.

4.3 Results and Discussion

In this project, our goal was to establish what perturbations associated with kinase inhibitors are explanatory variables of changes in cell network architecture and expression levels of macromolecular complexes of the AIS. For each of the cells we received two color channels, each of them associated with one macromolecular complex. The expression levels of macromolecular components are measured through the intensity values along the centerline of dendritic components and AIS in these images. We collected intensities from two dendritic components and one AIS from each of two color channels. Those values are concatenated for each cell and considered as the feature vector of corresponding cell. Since the lengths of those components are not necessarily same, the size of feature vectors varied. Because of that, the feature vector values are interpolated and at the end all feature vectors are organized to be in the same size, 1004.

Before the pattern recognition step, data had long processing steps. The noise caused by data acquisition techniques and the errors caused by our feature extraction method would create an artifact which can possibly effect the classification process. In order to eliminate these artifacts, we applied translation invariant wavelet transform. For that, we needed dyadic vectors therefore we zero padded the original

feature vector in order to reach 1024. The order of the coarsest scale is determined as 8, through the grid search. When we realized the positive effect of translation invariant transform on the feature vector to increase discrimination between classes, we investigated other invariant transforms. However since the discriminations between samples are already small, transformations such as scattering transform which are stable under small deformations do not positively effect the classification results.

After the feature extraction process is concluded, we analyzed several classification methods. Even though we attempted to use multi-kernel SVM, it gives the highest discrimination when the kernel is same for each sources (sources are considered as dendritic components and AIS from each of the color channels, which makes four sources). We wanted to apply sophisticated machine learning techniques as well, but the sample size were too small to apply those methods. Since the feature space is large, we analyzed feature reduction methods with the linear classifiers, but those combinations could not overcome nonlinear SVM. As a result we concluded to use nonlinear SVM for the classification process.

For the numerical tests we had samples associated with ten different couple of macromolecules. Samples from each of these ten sets are exposed with 3 or 4 perturbations. The classification result for the sample sets exposed with 4 perturbations is shown in Figure 4.3, where it is visible that some of the molecular pairs are more sensitive to the perturbations. In order to see the sensitivity to the perturbations, tests are done individually and separately for each of the molecular pairs. Results can be seen in Figure 4.4

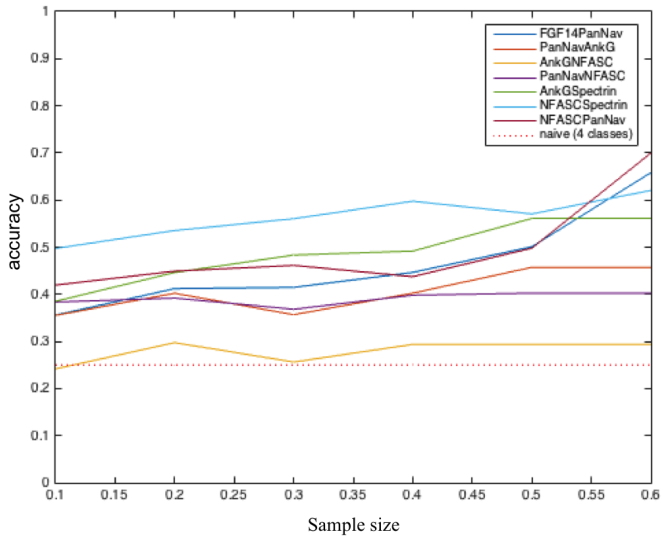

Figure 4.3: Classification of data. Translation-invariant wavelet transform is applied to raw feature vectors.

For the future direction of the project, we are planning to increase the sample size which would let us to apply machine learning methods efficiently. We are considering to expand the feature vectors too by including morphological features of neurites.

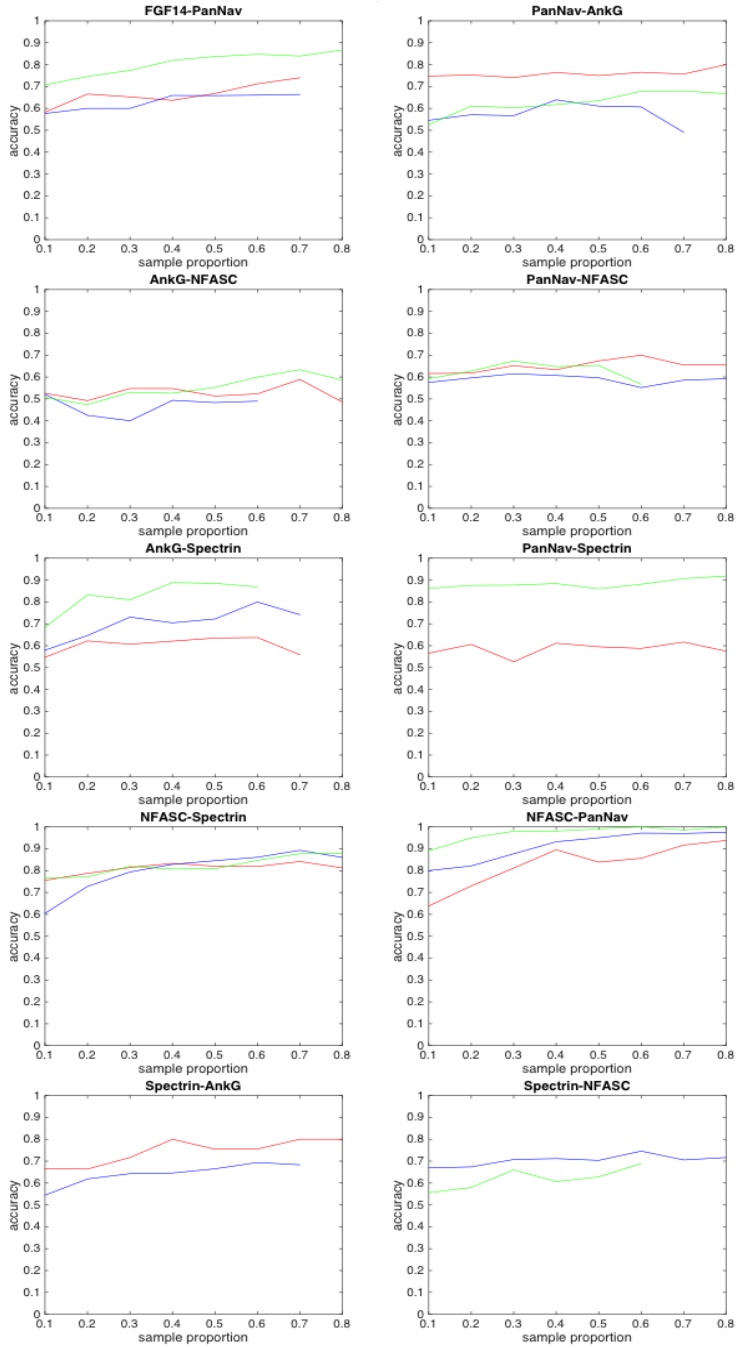

Figure 4.4: Sensitivity of channel pairs to the perturbations. These plots show the sensitivity of each of the channel pairs to the perturbations. *Red plot* is the control group, *Blue plot* is the perturbation CHIR, and *Green plot* is the perturbation Wee1

CHAPTER 5

Appendix

5.1 Infinite Impulse Filtering of Gaussian Function

*This material is referenced from [55]

The discrete convolution of the image f with the gaussian function g at the point x is given as $f(x) = \sum_{k=-K}^{k=K} f[x-k]g[k]$ where $K \to \infty$. Since $\lim_{x\to\infty} g(x) = 0$ the error will be negligible if summation is truncated at a proper finite point. This point is taken as $K = 3\sigma$ or $K = 5\sigma$ [55]. However operation required to

calculate that summation will be $O(K)$ per pixel which is linearly dependent on the density of Gaussian function. Therefore the cost of filtering process will be enormous when filter size is large. Young and Vliet [55] proposed a new approach to reduce the cost of gaussian filtering implementation. This approach is called as *infinite impulse filtering*. Method approximates the gaussian filtering response of one line of sample with a two step procedure. The advantage of this implementation is that the computational effort is independent of the density of gaussian function and it is $O(1)$ for each pixel.

$$\omega[x] = f[x] + \sum_{n=1}^{3} b_n \omega[x-n]$$

$$F[x] = B\omega[x] + \sum_{1}^{3} b_n g[x+n]$$

where F is the filtering response.

For that purpose, 6th order Taylor expansion of Gaussian function is used to approximate to its Fourier transform. The 6th order Taylor approximation of $\frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{-x^2}{2}}$ is given by as $(a_0 + a_1 x^2 + a_2 x^4 + a_3 x^6)^{-1} + \epsilon(t)$ with $\epsilon(t) \leq 2.7 \times 10^{-3}$ where $a_0 = 2.490895, a_1 = 1.466003, a_2 = -0.024393, a_3 = 0.178257$ in [1]. For $g(t) = \frac{1}{\sigma\sqrt{2\pi}} e^{-t^2/(2\sigma^2)}$ the Fourier transform is $G_\sigma(\omega) = e^{-\omega^2\sigma^2/2}$. By the above given approximation we have

$$G_\sigma(\omega) \cong= \frac{\sqrt{2\pi}}{a_0 + a_1(\omega\sigma)^2 + a_2(\omega\sigma)^4 + a_3(\omega\sigma)^6}.$$

The Laplace transform of it is

$$\mathcal{L}G_\sigma(\omega) \cong= \frac{\sqrt{2\pi}}{a_0 - a_1(s\sigma)^2 + a_2(s\sigma)^4 - a_3(s\sigma)^6}.$$

This can be factorized into two fractions:

$$G_1 = \frac{(\sqrt{2\pi}^{1/4})}{(1.1668 + \sigma s)(3.20373 + 2.21567.\sigma s + (\sigma s)^2)}$$

and

$$G_2 = \frac{(\sqrt{2\pi}^{1/4})}{(1.1668 - \sigma s)(3.20373 - 2.21567.\sigma s + (\sigma s)^2)}.$$

G_1 and G_2 represent two differential equations and can be transformed to difference equation. For that, forward difference technique is used for G_1 and backward difference technique is used for G_2. For that, one needs to substitute $s = (1 - z^-1)/T$ in G_1 and $s = (z - 1)/T$ in G_2 where T is the time step and is assumed to be 1 from now on. Hence forward and backward difference equations will be:

$$G_1 = \frac{A}{b_0 - b_1 z^{-1} - b_2 z^{-2} - b_3 z^{-3}}$$

and

$$G_2 = \frac{A}{b_0 - b_1 z^{1} - b_2 z^{2} - b_3 z^{3}}.$$

where

$$b_0 = 1.57825 + 2.44413\sigma + 1.4281\sigma^2 + 0.422205\sigma^3,$$

$$b_1 = 2.44413\sigma + 2.85619\sigma^2 + 1.26661\sigma^3,$$

$$b_2 = -1.4281\sigma^2 - 1.26661\sigma^3,$$

$$b_3 = 0.422205\sigma^3.$$

G_1 and G_2 correspond to difference equations. Therefore above calculation suggests the following: filtering the input with a gaussian function can be implemented as combination of filters correspond to these forward and backward difference equations. Therefore firstly input is filtered according to G_1 then the response is filtered according to G_2 which yields a recursive relation. The difference equations correspond to G_1 and G_2 are following:

forward:

$$w[n] = (1 - \frac{b_1 + b_2 + b_3}{b_0})input[n] + \frac{b_1 w[n-1] + b_2 w[n-2] + b_3 w[n-3]}{b_0}$$

backward:

$$output[n] = (1 - \frac{b_1 + b_2 + b_3}{b_0})w[n] + \frac{b_1 output[n+1] + b_2 output[n+2] + b_3 output[n+3]}{b_0}$$

Numerical experiments in [19] show that this implementation gives a very accurate approximation to Gaussian function with $\sigma \leq 20$. Since this upper bound is reasonable for our dataset we used this implementation safely. However for larger magnification factored images the error would be larger.

5.2 Characterization of Neuronal Components Through Multiscale Analysis

In order to examine the multiscale directional analysis methods on characterizing the neural images, we used pattern recognition techniques. Firstly we filtered the synthetic neural image with filters at 10 orientations. This process is repeated with a set of filters with increasing scales. The smallest filter scale is chosen slightly larger than radius of thinnest neurite and scales are increased with step size equals to radius of thinnest neurite. Filtering responses associated with each point are concatenated at the end to create feature vectors. Then machine learning methods are applied to see how discriminating the directional filtering responses for neuronal images. Training sample size is determined as 1 percent. Average classification accuracy over 10 trials was 81%. The classification result of one trial is given in figure 5.1. The figure shows that the classification accuracy is overall good but central part of one neurite is classified as soma. Then the same test is done with multiscale directional filters. The image is convolved with filters at 10 orientation and 10 scale, where the difference between two consecutive scale is larger than radius of neurite. Training sample size kept same and average classification accuracy over 10 trials was 97%t. This test shows that multiscale directional representation of neural images would be very efficient to characterize components of neurons. By using the same svm model, which is created on same synthetic data, we labeled the components of neurons in real data. The result overall is satisfactory and given in figure 5.2.

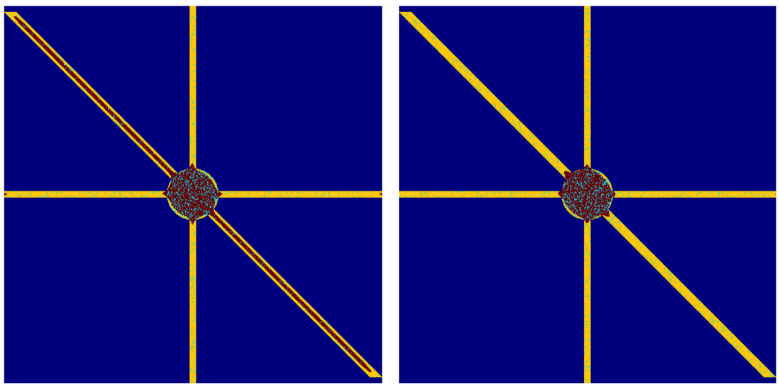

Figure 5.1: Directional analysis of neural images. *Left* One scale directional filtering responses are used as features to classify, blue points are training data, yellow points are classified as neurite and red points re classified as soma.*Right* Multi scale directional filtering responses are used as features to classify, blue points are training data, yellow points are classified as neurite and red points re classified as soma.

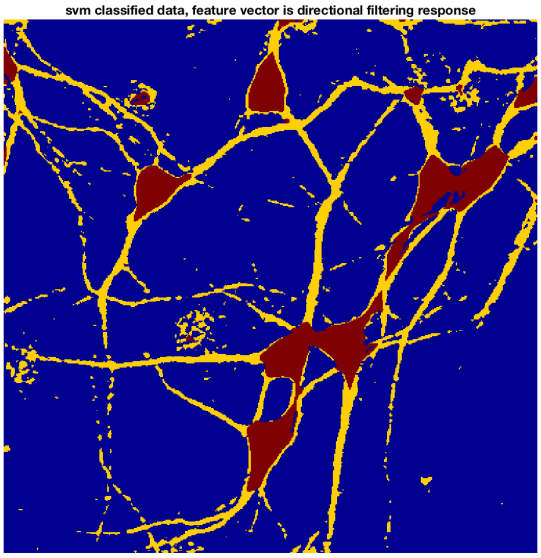

Figure 5.2: Multiscale directional analysis. By using directional filtering responses as features soma regions are detected through SVM based classification method. SVM model is generated on a basic single soma synthetic data, Figure 5.1. Model works surprisingly well on real data. Red regions are detected soma locations, red regions are detected neurites. Image size = 512×512 pixels (1 pixel = $0.28 \times 0.28 \mu m$).

5.3 Algorithms

Algorithm 5.1 Scale selection

Input: Following inputs are required.

- S - Segmented binary image.

Output:

- The domain for orientation, D_l is determined as $[\pi/4, 3*\pi/4]$. In numerical tests it is seen that two perpendicular orientation would be enough to approximate the scale.
- The domain for scale, D_j is determined as $[2 : 20]$ This interval works for all images that we worked on. However if one works on image with very small or very large cells this interval could be changed accordingly.

1: **for** $l \in D_l$ **do**
2: **for** $j \in D_j$ **do**
3: Convolve S with ϕ_{jl}.
4: **end for**
5: **end for**
6: Find the points in S which gives maximum filtering response, 1 ,for each directions l at the smallest scale $j = 2$. This step basically determines the points which are locally isotropic at minimum scale in the range. (For that step a tolerance value is given as 0.001)

7: For the each of the remaining points and for each of the directions, find scale where the filtering response start decreasing. For each point $p \in S$ 2 scale value will be determined. Then for each point $p \in S$ choose the minimum of 2 determined scales. For any point p this scale will give the information about the radius of the smallest circle which is completely inside of the cell and centered at p.

8: In the previous step the scale of local isotropy of each point is determined. The maximum of those scales will be returned as output of the algorithm.

Output: The output will be the largest radius of circle completely contained in cell.

Algorithm 5.2 Soma detection and segmentation

Input:
- Segmented Image S,
- Filter size which is determined in algorithm 5.1.

1: Anisotropic Gaussian filters are generated with rotations of 10 equally distanced angles.

2: Convolve S with each of the rotated filters.

3: For each $p \in S$ find minimum and maximum filtering response among 10 filtering responses.

4: Calculate *Directional Ratio* = minimum filtering response/maximum filtering response.

5: The points where *Directional Ratio* ≥ 0.85 are detected soma regions.

6: Soma segmentation process starts here. Velocity map for Fast Marching method is minimum filtering response^3/maximum filtering response. Starting curve for the evolution is the boundary of the detected soma regions. Evolution will start when the growth rate is smaller than 0.1.

Output: Segmented somas

Algorithm 5.3 Neurite Tracing-seed generation

Input: • Segmented Image S,

• Segmented Somas I.

1: $D =$ the distance transform of S, which assigns the value of the shortest distance from each point to the background.

2: Convolve D with matrix $\begin{pmatrix} -1/8 & -1/8 & -1/8 \\ -1/8 & 2 & -1/8 \\ -1/8 & -1/8 & -1/8 \end{pmatrix}$ to emphisize values the center-line of neurite on D.

3: Threshold D with value 0.16 to pick seed points.

4: Find the local maximum of each region of 3x3 squares.

5: *Process for second round seeding:*

6: **for** each point p in S **do**

7: Find seeds on $B_{D(p)}(p)$. If there is no seed pick the one farthest from the boundary as extra seed and add it to generated seeds.

8: **end for**

9: *Process to eliminate extra seeds*

10: **for** each of the seeds generated **do**

11: Find the deleted neighbor of seed with radius 2. If there is any other seed in the neighborhood eliminate the one closer to boundary.

12: **end for**

Output: • Coordinates of seeds P,

• Distance map D.

Algorithm 5.4 Neurite Tracing

Input: • Segmented Image S,

• Segmented Somas I.

1: **for** each of the somasI_i in I **do**

2: $S(I_j) = 0$ for each $j \neq i$, this ensures that tracing will not continue through other soma regions.

3: *Process to find starting seed for trace*

4: $SD_0 = I_i$,

5: SD_1 = Dilate I_i with disk structure elements whose radius is 6,

6: SD_2 = Dilate SD_1 with disk structure elements whose radius is 10,

7: SD_3 = Dilate SD_2 with disk structure elements whose radius is 8,

8: Calculate symmetric difference $SD_1 \triangle SD_0$ and $SD_3 \triangle SD_2$. This step will create two rings around I_i where R_1 is inner ring, R_2 is outer ring.

9: $R_1 = R_1 \cap S$ and $R_2 = R_2 \cap S$. Number of connected components in R_2 is the number of number of possible neurites of I_i.

10: The line from centroid of R_1 to centroid of R_2 gives the direction of neurite, d.

11: Starting from centroid of R_2 generate a line with direction $-d$ with large enough length. The intersection with I_i is the starting of trace s.

12: **for** each neurites of I_i **do**
13: **while** search for next seed returns at least one seed **do**
14: Start the trace from soma starting point s. Search for the next seed, algorithm 5.5 along the direction of neurite d. Connect next seed point to trace and update s as end point of trace.
15: **end while**
16: **end for**
17: **end for**
Output: Traced path for each neurite of each soma in I.

Algorithm 5.5 Search for the next seed

Input:
- initial seed s,
- Segmented Image S,
- Generated seeds P,
- Direction of neurite d.
- Number of the directions $nBand$

1: Rotation angle, $\alpha = 0$, subpath=[s], thickness of rectangular search region, thk=4, length of rectangular search region, l=30, number of line length, l, increment steps, inc,
2: $inc = 1$,
3: **while** $inc \leq 18$ **do**
4: l and thk are directly dependent on s. Therefore while s increasing l and thk increase linearly. Update thk and l,
5: **while** $\alpha \leq 2\pi/3$ or subpath is empty **do**
6: $d_1 = d + \alpha, d_2 = d - \alpha$, according to rotation angle direction of the rectangle is changed in both clockwise and counterclockwise.
7: Create two rectangles with direction d_1 and d_2, thickneess thk and length l,
8: Set s_n as the closest one among all seeds covered by at least one of the rectangles generated in the previous step.
9: **if** s_n is nonempty **then**
10: Connect s with s_n, update subpath as the trace from s to s_n and $s = s_n$.
11: **else**
12: $inc = inc + 1, \quad \alpha = \alpha + \pi/\text{nBands}$,
13: **end if**
14: **end while**
15: **end while**
Output: Next seed to add to path

5.4 Soma Segmentation Accuracy Table

Table 5.1: Detailed soma segmentation performance of our method with and without automated scale selection algorithm for 20 images

Image	DC	DC_autoScale	Scale	Auto_Scale	#soma	#continugous Somas
1	0.91	0.87	9	7	7	0
2	0.92	0.86	9	11	6	1
3	0.84	0.84	9	11	7	0
4	0.90	0.88	9	10	7	1
5	0.89	0.85	9	10	4	0
6	0.70	0.65	9	11	7	0
7	0.78	0.86	9	10	5	0
8	0.76	0.82	13	9	2	0
9	0.88	0.80	13	7	1	0
10	0.90	0.88	13	15	1	0
11	0.92	0.86	13	8	1	0
12	0.84	0.83	13	14	1	0
13	0.89	0.91	9	8	1	0
14	0.92	0.81	9	12	2	0
15	0.89	0.91	9	10	5	0
16	0.77	0.77	9	9	4	0
17	0.74	0.74	9	10	5	0
18	0.77	0.81	9	8	2	0
19	0.74	0.85	9	3	2	0
20	0.79	0.86	9	6	1	0

Table 5.2: Detailed soma segmentation performance of morphological methods with and without automated scale selection algorithm for 20 images

Image	DC	DC_autoScale	Scale	Auto_Scale	#soma	#continugous Somas
1	0.91	0.91	15	12	7	0
2	0.92	0.91	15	18	6	1
3	0.92	0.91	15	18	7	0
4	0.85	0.87	15	17	7	1
5	0.85	0.76	15	17	4	0
6	0.90	0.80	15	18	7	0
7	0.86	0.86	15	17	5	0
8	0.37	0.40	23	16	2	0
9	0.70	0.23	23	12	1	0
10	0.49	0.53	23	27	1	0
11	0.89	0.80	23	14	1	0
12	0.88	0.55	23	25	1	0
13	0.70	0.39	15	13	1	0
14	0.48	0.70	15	20	2	0
15	0.92	0.90	15	17	5	0
16	0.74	0.74	15	15	4	0
17	0.81	0.71	15	17	5	0
18	0.70	0.76	15	13	2	0
19	0.40	0.38	15	5	2	0
20	0.47	0.65	15	10	1	0

5.5 Neurite Detection Accuracy Table

Table 5.3: Neurite detection accuracy: images1-4

imageName_SomaNumber	number of detected neurites	number of neurites
image1_soma1	3	4
image1_soma2	4	4
image1_soma3	4	4
image1_soma4	3	4
image2_soma1	3	3
image2_soma2	3	3
image2_soma3	5	5
image2_soma4	2	4
image2_soma5	5	5
image2_soma6	2	3
image2_soma7	4	4
image3_soma1	6	6
image3_soma2	2	3
image3_soma3	5	6
image3_soma4	2	2
image3_soma5	3	3
image3_soma6	5	7
image3_soma7	4	5

Table 5.4: Neurite detection accuracy: images5-10

imageName_SomaNumber	number of detected neurites	number of neurites
image4_soma1	4	4
image4_soma2	4	4
image4_soma3	5	5
image4_soma4	2	2
image4_soma5	2	3
image4_soma6	3	3
image4_soma7	2	2
image5_soma1	2	2
image5_soma2	5	5
image5_soma3	3	5
image5_soma4	3	4
image6_soma1	3	3
image6_soma2	5	5
image7_soma1	4	5
image7_soma2	3	4
image8_soma1	4	5
image8_soma2	5	5
image8_soma3	4	4
image8_soma4	4	5
image8_soma5	4	4
image8_soma6	3	3
image8_soma7	1	1
image9_soma1	4	4
image9_soma2	3	3
image9_soma3	4	4
image9_soma4	4	4
image9_soma5	3	3
image10_soma1	2	3
image10_soma2	6	6
image10_soma3	4	4
image10_soma4	1	2

Bibliography

[1] Abramowitz, M., and Stegun, I. A. *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*, ninth dover printing, tenth gpo printing ed. Dover, New York, 1964.

[2] Acciai, L., Soda, P., and Iannello, G. Automated neuron tracing methods: An updated account. *Neuroinformatics 14* (2016), 353–367.

[3] Al-Kofahi, K. A., Lasek, S., Szarowski, D. H., Pace, C. J., Nagy, G., Turner, J. N., and Roysam, B. Rapid automated three-dimensional tracing of neurons from confocal image stacks. *IEEE Trans Inf Technol Biomed 6*, 2 (June 2002), 171–187.

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

[5] Altman, D. G., and Bland, J. M. Diagnostic tests. 1: Sensitivity and specificity. *BMJ 308*, 6943 (1994), 15–52.

[6] Antoine, J.-P., Carrette, P., Murenzi, R., and Piette, B. Image analysis with two-dimensional continuous wavelet transform. *Signal processing 31*, 3 (1993), 241–272.

[7] Bamberger, R. H., and Smith, M. J. A filter bank for the directional decomposition of images: Theory and design. *IEEE transactions on signal processing 40*, 4 (1992), 882–893.

[8] Candès, E. J., and Donoho, D. L. Ridgelets: A key to higher-dimensional intermittency? *Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences 357*, 1760 (1999), 2495–2509.

[9] Candes, E. J., and Donoho, D. L. Curvelets: A surprisingly effective nonadaptive representation for objects with edges. Tech. rep., Stanford Univ Ca Dept of Statistics, 2000.

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

[11] Daubechies, I. *Ten lectures on wavelets*. SIAM, 1992.

[12] Deschamps, T., and Cohen, L. D. Fast extraction of minimal paths in 3d images and applications to virtual endoscopy. *Medical image analysis 5*, 4 (2001), 281–299.

[13] Do, M. N., and Vetterli, M. The contourlet transform: an efficient directional multiresolution image representation. *Image Processing, IEEE Transactions on 14*, 12 (2005), 2091–2106.

[14] Dolmetsch, R., and Geschwind, D. H. The human brain in a dish: The promise of iPSC-derived neurons. *Cell 145*, 6 (2011), 831–834.

[15] Donoho, D., Maleki, A., and Shahram, M. Wavelab 850. *Software toolkit for time-frequency analysis* (2006).

[16] Donoho, D. L. De-noising by soft-thresholding. *IEEE Trans Info Theory 41*, 3 (May 1995), 613–627.

[17] Donoho, D. L., and Johnstone, J. M. Ideal spatial adaptation by wavelet shrinkage. *Biometrika 81*, 3 (1994), 425–455.

[18] El-Laithy, K., Knorr, M., Ks, J., and Bogdan, M. Digital detection and analysis of branching and cell contacts in neural cell cultures. *J. Neurosci. Methods 210*, 2 (2012), 206–219.

[19] Geusebroek, J.-M., Smeulders, A., and van de Weijer, J. Fast anisotropic gauss filtering. *IEEE Trans Image Process 12*, 8 (Aug 2003), 938–943.

[20] Gonzalez, R. C., Woods, R. E., and Eddins, S. L. *Digital Image Processing Using MATLAB*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 2003.

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

[22] Heil, C. *A basis theory primer: expanded edition*. Springer Science & Business Media, 2010.

[23] Jimenez, D., Labate, D., Kakadiaris, I., Papadakis, M., and Kakadiaris. Improved automatic centerline tracing for dendritic and axonal structures. *Neuroinformatics 13*, 2 (2015), 227–244.

[24] Jimenez, D., Labate, D., Kakadiaris, I. A., and Papadakis, M. Improved automatic centerline tracing for dendritic and axonal structures. *Neuroinformatics* (2014), 1–18.

[25] Jimenez, D., Labate, D., and Papadakis, M. Directional analysis of 3d tubular structures via isotropic well-localized atoms. *Appl Comput Harmon Anal* (2015), 1–18.

[26] Jimenez, D., Papadakis, M., Labate, D., and Kakadiaris, I. Improved automatic centerline tracing for dendritic structures. In *Biomedical Imaging (ISBI), 2013 IEEE 10th International Symposium on* (April 2013), pp. 1050–1053.

[27] Kayasandik, C. B., and Labate, D. Improved detection of soma location and morphology in fluorescence microscopy images of neurons. *Journal of neuroscience methods 274* (2016), 61–70.

[28] Kherlopian, A. R., Song, T., Duan, Q., Neimark, M. A., Po, M. J., Gohagan, J. K., and Laine, A. F. A review of imaging techniques for systems biology. *BMC systems biology 2*, 1 (2008), 74.

[29] Kingsbury, N. Complex wavelets for shift invariant analysis and filtering of signals. *Applied and computational harmonic analysis 10*, 3 (2001), 234–253.

[30] Kutyniok, G., and Labate, D. *Introduction to Shearlets*. Birkhäuser Boston, Boston, 2012, pp. 1–38.

[31] Kutyniok, G., and Labate, D. *Shearlets: Multiscale Analysis for Multivariate Data*. Springer, 2012.

[32] Labate, D., Laezza, F., Negi, P., Ozcan, B., and Papadakis, M. Efficient processing of fluorescence images using directional multiscale representations. *Mathematical Modelling of Natural Phenomena 9*, 5 (2014), 177–193.

[33] Labate, D., Laezza, F., Negi, P., Ozcan, B., and Papadakis, M. Efficient processing of fluorescence images using directional multiscale representations. *Math. Model. Nat. Phenom. 9*, 5 (2014), 177–193.

[34] Lampert, C. H., and Wirjadi, O. An optimal nonorthogonal separation of the anisotropic gaussian convolution filter. *IEEE Trans Image Process 15*, 11 (Nov 2006), 3501–3513.

[35] Liu, Y. The diadem and beyond, 2011.

[36] Luisier, F., Vonesch, C., Blu, T., and Unser, M. Fast haar-wavelet denoising of multidimensional fluorescence microscopy data. In *ISBI* (2009).

[37] Mallat, S., and Peyré, G. A review of bandlet methods for geometrical image representation. *Numerical Algorithms 44*, 3 (2007), 205–234.

[38] Nieland, T. J. F., Logan, D. J., Saulnier, J., Lam, D., Johnson, C., Root, D. E., Carpenter, A. E., and Sabatini, B. L. High content image analysis identifies novel regulators of synaptogenesis in a high-throughput rnai screen of primary neurons. *PLoS ONE 9*, 3 (03 2014).

[39] Osher, S., and Fedkiw, R. P. *Level set methods and dynamic implicit surfaces*. Applied mathematical science. Springer, New York, N.Y., 2003.

[40] Ozcan, B., Labate, D., Jiménez, D., and Papadakis, M. Directional and non-directional representations for the characterization of neuronal morphology. *Wavelets XV (San Diego, CA, 2013), SPIE Proc 8858* (2013), 1050–1053.

[41] Ozcan, B., Labate, D. Jimenez, D., and Papadakis, M. Directional and non-directional representations for the characterization of neuronal morphology. In *Wavelets XV (San Diego, CA, 2013), SPIE Proceedings* (2013), vol. 8858, pp. 1050–1053.

[42] Ozcan, B., Negi, P., Laezza, F., Papadakis, M., and Labate, D. Automated detection of soma location and morphology in neuronal network cultures. *PloS One 10*, 4 (2015).

[43] Pawley, J. B. *Handbook of biological confocal microscopy.* Springer, New York (N.Y.), 2006.

[44] Peng, H., Meijering, E., and Ascoli, G. A. From DIADEM to bigneuron. *Neuroinformatics 13*, 3 (2015), 259–260.

[45] Perona, P. Steerable-scalable kernels for edge detection and junction analysis. *Image Vision Comput. 10*, 10 (Dec. 1992), 663–672.

[46] Schmitz, S. K., Hjorth, J. J., Joemai, R. M., Wijntjes, R., Eijgenraam, S., de Bruijn, P., Georgiou, C., de Jong, A. P., van Ooyen, A., Verhage, M., Cornelisse, L. N., Toonen, R. F., and Veldkamp, W. Automated analysis of neuronal morphology, synapse number and synaptic recruitment. *J Neurosci Methods 195*, 2 (2011), 185–193.

[47] Sethian, J. A. *Level Set Methods and Fast Marching Methods.*, 2 ed. Cambridge University Press, June 1999.

[48] Sharma, K., Choi, S.-Y., Zhang, Y., Nieland, T. J., Long, S., Li, M., and Huganir, R. L. High-throughput genetic screen for synaptogenic factors: Identification of {LRP6} as critical for excitatory synapse development. *Cell Reports 5*, 5 (2013), 1330 – 1341.

[49] Shavkunov, A. S., Wildburger, N. C., Nenov, M. N., James, T. F., Buzhdygan, T. P., Panova-Elektronova, N. I., Green, T. A., Veselenak, R. L., Bourne, N., and Laezza, F. The fibroblast growth factor 14voltage-gated sodium channel complex is a new target of glycogen synthase kinase 3 (GSK3). *J Biol. Chem. 288*, 27 (2013), 19370–85.

[50] Svoboda, K. The past, present, and future of single neuron reconstruction. *Neuroinformatics 9* (2011), 87–98.

[51] Vallotton, P., Lagerstrom, R., Sun, C., Buckley, M., Wang, D., and et al. Automated analysis of neurite branching in cultured cortical neurons using hca-vision. *Cytom. Part A 71*, 10 (2007), 889–895.

[52] Weaver, C. M., Pinezich, J. D., Lindquist, W. B., and Vazquez, M. E. An algorithm for neurite outgrowth reconstruction. *J. Neurosci. Methods 124* (2003), 197–205.

[53] Wojtaszczyk, P. *A mathematical introduction to wavelets*, vol. 37. Cambridge University Press, 1997.

[54] Yan, C., Li, A., Zhang, B., Ding, W., Luo, Q., and Gong, H. Automated and accurate detection of soma location and surface morphologyin large-scale 3D neuron images. *PLoS ONE 8*, 4 (2013), 1–12.

[55] Young, I. T., and Van Vliet, L. J. Recursive implementation of the gaussian filter. *Signal processing 44*, 2 (1995), 139–151.

[56] Yuan, X., Trachtenberg, J. T., Potter, S. M., and Roysam, B. Mdl constrained 3-d grayscale skeletonization algorithm for automated extraction of dendrites and spines from fluorescence confocal images. *Neuroinformatics 7*, 4 (2009), 213.

[57] Zhang, B., Fadili, M.-J., and Starck, J.-L. Wavelets, ridgelets, and curvelets for poisson noise removal. *IEEE Trans. Image Process 17* (2008), 1093–1108.