Enhancement and Segmentation of Vascular Structures for Biological Image Analysis

---

A Dissertation

Presented to

The faculty of the School of Engineering and Applied Science

University of Virginia

In partial fulfillment

of the requirements for the degree

Doctor of Philosophy (Electrical and Computer Engineering)

by

Suvadip Mukherjee

August 2015

Approval Sheet

This dissertation is submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy (Electrical and Computer Engineering)

Author: Suvadip Mukherjee

This dissertation has been read and approved by the examining committee:

Scott T. Acton, Dissertation Adviser

Stephen G. Wilson, Committee Chair

Zongli Lin, Committee Member

Connelly Barnes, Committee Member

Barry G. Condron, Committee Member

Accepted for the School of Engineering and Applied Science:

Dean, School of Engineering and Applied Science

August 2015

Abstract

The neuronal content of an organism, the individual morphology of each neuron and the variability of these components constitute the atlas of the *neurome*. The description of such an atlas will be critical in determining the complex neural system of an organism, eventually providing clues to how animals think and function. As the organisms under investigation scale from the worm to the human, the number of neurons range from hundreds to trillions. Image analysis is a key ingredient towards the neurome for complex organisms. Specifically, from an image processing engineer's perspective, an essential aspect of this study is development of segmentation algorithms to obtain an automated digital reconstruction of the neural morphology.

The primary objective of this thesis is to develop novel image analysis methods for neuromics – the study of neurons. We investigate automated algorithms for segmenting filamentous neuronal structures from noisy and cluttered confocal microscopy images. The end goal is to embed the shape and geometry of the segmented neurons in a mathematically representable graph theoretic tree. Different categories of neurons of the fruit fly *Drosophila* are studied, and segmentation algorithms are proposed for detecting single neuron cells which are genetically tagged with florescence protein. The neuronal trees, which are automatically identified from 2D or 3D images are represented in a *SWC* format. This provides a digital reconstruction that could be used for future neurological studies such as shape comparison, structural categorization and disease identification.

In this regard, we propose three automated neuron segmentation algorithms. The first method, Tree2Tree-2 is a graph based tracer that uses graph theory in conjunction with image analysis to perform tracing. The primary obstacle in neuron tracing is to identify the correct connections between the neuron sub compartments in presence of noise and clutter. Tree2Tree-2 addresses this issue using a graph based geometry analysis procedure. Tree2Tree-2 improves on its predecessor Tree2Tree by establishing a robust variational formulation for initial segmentation and a shortest path based curve refinement policy to accurately link the neurites. Using this strategy we observe an improvement in tracing performance over Tree2Tree by xx%.

While the above mentioned solution is effective for relatively simpler morphologies, we hypothesize that a more efficient solution can be achieved by introducing a methodology which is adaptive to the object topology. One approach to solve this problem is by using geometric active contours, which are capable of automatically merging and splitting the deformable model in a natural framework. This motivates our next solution, Legendre Level Sets (L2S), which is designed to perform region based segmentation of 2D images in

presence of heterogeneous intensity. L2S is a general purpose segmentation technique, and is applied to neuron segmentation for 2D images. The primary contribution of L2S is to enable robust processing in scenarios where the illumination is inconsistent and contrast is low. We have shown our results on multiple 2D microscopy images and the results suggest significant improvement over the state of the art with an improved accuracy of yy %.

To extend the framework for 3D neuron segmentation, we introduce the third method Tubularity Flow Field (TuFF). TuFF uses level sets for segmentation, and is guided by the local orientation of the filamentous structures for curve propagation. Where L2S is a generalized method, TuFF is specifically designed for applications involving filamentous objects. The advantage of using a geometric deformable model is evident from TuFF's ability to correctly identify local connections which cannot be established by explicit methodologies. The experimental results on a set of 2D and 3D images are evaluated against three popular semi automated and automated tracers, and we observe an accuracy of zz % mean absolute error which is an improvement of mm% over the competing methods. We further improve TuFF's performance by proposing a novel methodology to enhance low contrast vascular structures and some non biological applications in civil engineering is finally discussed which suggests the broad applicability of the methodology.

Contents

List of Figures

List of Tables

Symbols

a	Scalar notation
$\mathbf{a}$	Vector notation
[A]	Matrix notation
Ω	Domain of the image function ($\Omega \subset \mathbb{R}^2/\mathbb{R}^3$)
$\mathbf{x}$	Position vector (x,y) or (x,y,z) $\in \Omega$
$*$	Convolution operator
$\cdot$	Vector dot product operator
t	Scalar variable denoting time/pseudo time
σ	Scale parameter
$f(\mathbf{x})$	Continuous image function, $f : \Omega \mapsto \mathbb{R}$
$g_\sigma(\mathbf{x})$	Zero mean isotropic Gaussian kernel, with std. deviation σ
∇	Gradient operator
$\text{div}(\mathbf{a})$	Divergence operator, $\nabla \cdot \mathbf{a}$
∇^2	Laplacian operator
$N_\sigma(\mathbf{x})$	*Vesselness* response at scale σ
$N(\mathbf{x})$	Scale-space *vesselness* response
$\mathcal{H}_\sigma(\mathbf{x})$	Hessian matrix of $f(\mathbf{x}) * g_\sigma(\mathbf{x})$
$\phi(\mathbf{x}, t)$	Level set function, $\phi : \Omega \times \mathbb{R}^+ \mapsto \mathbb{R}$
$H(\phi)$	Heaviside function
$\delta(\phi)$	Dirac delta function
$\mathbf{C}(p)$	Parametric curve, $p \in \{0, 1\}$
$\mathbf{C}_t$	$\frac{\partial \mathbf{C}}{\partial t}$
$\mathbf{n}$	Unit outward normal vector to a curve
$\mathbf{t}$	Unit tangent vector to a curve
$\mathcal{P}_k(\mathbf{x})$	Legendre polynomial of degree k
$\mathbb{P}(\mathbf{x}) = (\mathcal{P}_0(\mathbf{x}), \ldots, \mathcal{P}_N(\mathbf{x}))^T$	Vector of Legendre polynomials

Chapter 1

Introduction

Recent years have witnessed an increasing trend in collaborative research between the field of biological sciences and engineering. In particular, advances in modern day imaging techniques have enabled biologists to image cellular and subcellular structures over a vast range of spatial resolution, ranging from micrometer to nanometer scale. Using state of the art imaging protocols, biologists are now able to generate image data at an unprecedented scale. However, while imaging is no longer considered a bottleneck for experimental biology, the sheer volume of data being collected calls for automated processing for high throughput study in cell biology. This has established a new field of interdisciplinary research – *Bioimage Informatics* [5], leading to a number of cross disciplinary publications as well as software suits for image processing techniques which are specialized for such biological tasks [6–8].

1.1 Neuroimage analysis

A subcategory of the aforementioned discipline of bioimage informatics involves image analysis for neuroscience. Researchers in this field of *Neuroimage analysis* borrow techniques from digital image analysis and computer vision for deeper understanding of the brain's functionality (for a model organism) through image based studies.

Functions of an animal's brain are largely governed by its neurons, and the number of neurons vary between a few hundreds in the roundworm *C. elegans* [9] to a hundred billion in an adult human brain. The relationship between the morphology and functionality of neurons was established by Ramon Y Cajal in the 19th century. Cajal's hypothesis serves as the basis for modern day neuroimage analysis. Studies based on morphological properties of individual neurons and neuronal components such as dendritic spines, synapses, mitochondria etc. have shown promise in better understanding and diagnosis of various neurological disorders and neuro-degenerative diseases [10–14]. It is evident that neuroimage analysis becomes a big data problem as we prepare ourselves to study the nervous system of vertebrates. This suggests that the prevalent norm of data interpretation by a trained human personnel needs to be replaced with sophisticated automation. It is not surprising that this problem has been receiving significant attention over the last few years. For example, the publicly accessible website *neuromorpho.org* [15] was published in 2006 with only a few hundreds of neurons in its repository. As of June 2015, neuromorpho.org contains more than ten thousand digitally reconstructed neurons, contributed by researchers from over 120 laboratories worldwide [16].

Two basic steps are involved in designing a platform for image based study of the brain – image acquisition and image analysis [17].

1.1.1 Image Acquisition

Choice of a particular imaging modality depends on the specific application. Fluorescence microscopy is a popular choice when the study involves a global structural analysis of the neurons or some neuronal components in the micrometer scale. For such imaging techniques, the specimen is genetically tagged with a fluorescence protein (GFP, YFP etc.) which emits photons when illuminated by a light source [13]. These photons are eventually detected by a sensor to produce a digital image of an optical plane. Laser

scanning confocal microscopes are commonly used for fast three dimensional imaging of neurons of model animals such as Drosophila, rat, mice etc. Depending on the application, other imaging techniques such as bright-field microscopy [18], multiphoton microscopy [19] etc. are also used to image neuronal structures. Electron microscopy (EM) is a popular choice for imaging neuronal structures at nanometer scale. EM is particularly useful in analyzing subcellular objects and surrounding structures such as mitochondria, synapse, vesicles etc.

1.1.2 Image analysis

While we are still far away from achieving our end goal of understanding the brain, recent research suggest that detection and quantification of morphological anomalies of certain neuronal structures can answer some relevant questions related to diagnosis of certain neural disorders. Specifically, morphological structure of individual neurons, dendritic spines and certain characteristics of subcellular objects such as synapses, mitochondria etc. reveal important information regarding the brain's functioning. Anomaly quantification can be performed via comparison of the shape of the objects, which in turn requires a robust segmentation technique. Broadly, the relevant research in neuroimage analysis can be categorized into the following groups: segmentation and shape analysis of individual neurons [1, 2, 20–22], study of the types of dendritic cells and characteristics of the intra neuronal structures [23–26]. While the end goal remains the same, all these methods differ considerably from the engineering point of view and require different imaging modalities. As a result, the processing algorithms differ considerably in nature, thus making each of these techniques individual topic of extensive research.

1.2 Problem formulation

In the recent years there have been concerted efforts to develop analytic models for global morphological comparisons of neurons [17]. This is because anatomical distortion of neurons provide initial clues toward neurological disease understanding, diagnosis or monitoring. This refers to the branch of study where the geometric and morphological properties of single neuron cells are studied for better understanding of its functioning. Since it is widely believed that structural anomaly of neurons correlate well with changes in its functioning, such global morphological assessments are essential for performing tasks such as quantification of the neuron degeneration due to a disease or drug usage, identifying young and adult neurons in the brain, differentiating between cells in different layers of the brain etc. [10–14, 27].

Designing a workflow for the aforementioned global shape based study involves two critical steps. First, a digital reconstruction should be obtained from the raw image data. This is the segmentation or tracing stage. Automated algorithms for performing high throughput digital reconstruction is crucial in developing building the neuronal atlas for a species. Till date, such an atlas exists for the round worm *C. elegans* [9], where the neurons exhibit significantly simpler structural patterns. However, for developed species such as the fruit fly, mice, zebrafish, huumans, *etc.* developing such shape based neuron atlas is still an unsolved problem.

Having a shape based neuronal atlas for an animal would open doors for further statistical studies based on the neuronal anatomy. Also, digital reconstructions allow us to mathematically compare the cell shapes for detecting morphological anomalies. It turns out that both these sub-problems come with their own sets of challenges and complications and deserve to be treated separately.

The pertinent challenge for global structural analysis is to develop appropriate pipeline

for identification and quantification of the morphology of a single neuron. Confocal microscopy is generally the chosen modality for imaging, since the entire neuron cell can be imaged in the micrometer resolution. Neuron reconstruction (or tracing) refers to the problem of acquiring the neural anatomy from microscopy. Image processing is challenging both due to the structural complexity of neurons as well as due to imaging artifacts such as poor contrast, presence of non-neuronal clutter and low signal to noise ratio of the images.

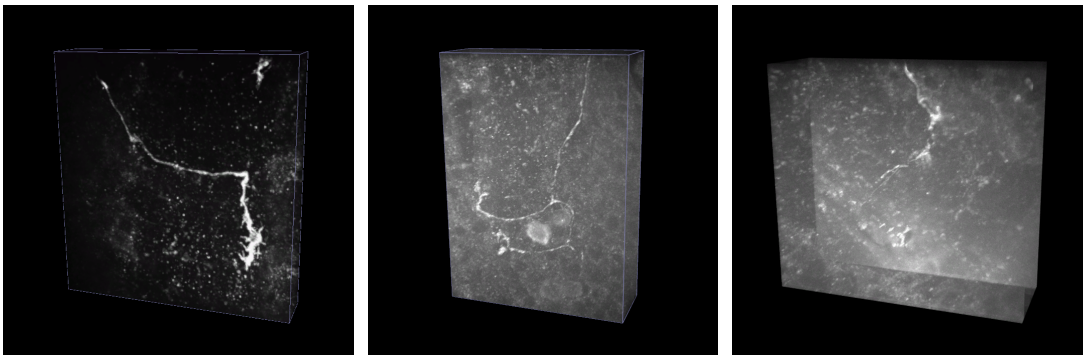
This dissertation addresses the problem of developing automated algorithms for segmenting single neurons from 2D and 3D microscopy data. The final goal is to construct digital reconstruction of the neurons, so that their structural pattern can be embedded in a mathematical structure for future shape based comparisons. Furthermore, automated algorithms are necessary for developing the "Neurome" or atlas of neurons for a model organism for future studies.

We demonstrate the applicability of our developed methods primarily on neuron images of the fruit fly *Drosophila*, which are imaged in Dr. Barry Condron's laboratory at the University of Virginia (Department of Biology). Since developing imaging protocols is not one of our aims, we briefly discuss the image acquisition step in the following segment.

1.2.1 Single neuron imaging

We are interested in investigating the morphological properties of single neurons of the fruit fly Drosophila. A detailed survey of the imaging protocols is elaborate, and is out of scope of this dissertation. However, a brief summary of the imaging method is discussed here to understand on the dataset that we will be using for analyzing our algorithms.

Biologists have shown interest in studying the neuronal processes (axons, dendrites, synapses etc.) of the Drosophila, which has been a preferred model organism for to study genetics and developmental biology for several years. The central nervous system (CNS)



(a) (b) (c)

FIGURE 1.1: Drosophila neurons imaged by confocal microscope. (a) is a low SNR sensory neuron. The background clutter are due to illuminated non neuronal filaments. (b) and (c) are interneurons, the image quality severely degraded by photon noise, low signal intensity and in-homogeneous contrast.

of the Drosophila contains a vast array of interacting synapses and neuronal processes in addition to containing about 20,000 neurons in the larval stage.

Green Florescence Protein (GFP) is used to label the neuronal cells, which are produced through a combination of a FLP/FRT system and GAL4/UAS system. Approximately 10 hr old embryos were heat shocked at 37°C for 1-2 hrs, that generates single GFP-labeled cells [12]. Using this protocol, most Ventral Nerve Cord (VNC)'s have about 5 labeled cells, thus allowing for imaging of individual neurons.

For imaging the labeled cells, confocal microscopy was used to image the cells in three dimensions, with resolution in the micrometer range. Three images captured in the Condron Lab at the University of Virginia are shown in Fig. 1.1

1.3 Contributions of this thesis

The major emphasis of this dissertation will be on developing novel algorithms for segmenting single neurons from confocal microscopy data. We realize that a large scale structural analysis of neuron groups demand efficient, automated segmentation to generate the digital morphology. Therefore, in this work, we primarily focus on developing

and improving the first stepping stone for *neuromics*– automated neuron segmentation algorithms.

Contribution 1: Graph theoretic neuron segmentation

We devise a novel neuron tracing technique Tree2Tree-2 [21], which combines the strengths of variational segmentation and graph based connectivity analysis of disjoint connected components. We also introduce a novel methodology to generate a *medialness* function [28] for tubular objects, for the purpose of obtaining a smooth tracing along the neuron centerline.

Contribution 2: Region based 2D segmentation with level sets

2D analysis often serves as a preliminary step for understanding the anatomy of neurites. Furthermore, certain categories of neurons (e.g. sensory neurons on the cuticle layer of *Drosophila* larva) are topologically flat and therefore, the third dimension of imaging does not yield useful information for analysis. In this regard, we have developed a general purpose segmentation algorithm which uses geometric active contours. The proposed algorithm, *Legendre Level Set* (L2S) [29] aims at segmenting the objects from microscopy images in presence of heterogeneous illumination.

Contribution 3: Tubularity field based 3D segmentation with level sets

We propose a novel neuron tracing architecture, Tubularity Flow Field (TuFF) [22], which uses geometric active contours to perform segmentation guided by the local tubularity of the neurites. One advantage of using geometric active contours is that these techniques are adaptive to the topology of the objects. As a result, joining disjoint neurites can be handled in a natural framework unlike Tree2Tree-2, where error is often introduced in the solution due to improper connectivity handling. We also provide a mechanism to combat

the sporadic signal loss across the structures by incorporating a specialized attraction force in our solution to merge nearby fragments.

1.4 Thesis outline

The rest of the dissertation is organized as follows: In Chapter 2, we discuss popular neuron segmentation techniques, as well as relevant methodologies for pre-processing. The graph based segmentation algorithm is discussed in Chapter 3 and the results are scrutinized for both 2D and 3D data. We identify the pros and cons of our method and discuss the motivation of using geometric active contours for more flexibility in establishing component connectivity.

In Chapter 4, a brief summary of geometric active contours is presented, followed by discussion of the proposed 2D segmentation method L2S in Chapter 5. The 3D segmentation case using TuFF is presented in Chapter 6, along with strategies to improve segmentation results using a robust non local vessel detection technique. Finally, we conclude in Chapter 7 with discussion of the methods, their future extensions and possible applications.

Apart from the aforementioned primary contributions, we found that the developed algorithms are quite general, and have a wide variety of applications involving vascular structures. Two such applications are discussed in the Appendix; the first one involves segmenting human arteries from low SNR ultrasound imagery, and the second method relates to the civil engineering discipline, where the goal is to detect cracks in concrete structures such as bridges and pavements.

1.5 Publications resulting from this work

Journal Publications

- **S. Mukherjee**, B. Condron and S.T. Acton, "Tubularity Flow Field – A Technique For Automatic Neuron Segmentation," *IEEE Transactions on Image Processing*, vol.24, no.1, pp.374,389, Jan. 2015

- **S. Mukherjee** and S.T. Acton, "Region Based Segmentation in Presence of Intensity Inhomogeneity Using Legendre Polynomials," *IEEE Signal Processing Letters*, vol.22, no.3, pp.298,302, March 2015

- R.Sarkar, **S. Mukherjee** and S.T. Acton, "Dictionary Learning Level Sets" *IEEE Signal Processing Letters* (*under minor revision*)

- **S. Mukherjee**, L. Boulton and S.T. Acton, "Concrete crack detection using edge assisted Tubularity Flow Field with local directional evidence", *in preparation.*

Conference Publications

- **S. Mukherjee** and S.T. Acton, "Oriented Filters for Vessel Contrast Enhancement With Local Directional Evidence", *IEEE ISBI 2015*(accepted).

- M. Consylman, **S. Mukherjee**, D.P. Mukherjee, B. Condron and Scott T. Acton, "Social behavior analysis of Drosophila larvae via motion activity recognition", *IEEE SSIAI 2014.*

- **S. Mukherjee**, B. Condron and S.T. Acton, "Neuron segmentation with level sets", *ACSSC 2013*:1078-1082

- R. Sarkar, **S. Mukherjee** and S. T. Acton, "Shape descriptors based on compressed sensing with application to neuron matching", *ACSSC 2013*: 970-974

- **S. Mukherjee** and S. T. Acton, "Vector field convolution medialness applied to neuron tracing," *ICIP 2013*: 665-669

- **S. Mukherjee**, B. Condron and S. T. Acton, "Chasing the neurome: Segmentation and comparison of neurons," *EUSIPCO 2013*: 1-4

- **S. Mukherjee**, S. Basu, B. Condron and S.T. Acton , "Tree2Tree2: Neuron tracing in 3D," *ISBI 2013*: 448-451

- **S. Mukherjee**, S. Basu, B. Condron and S.T. Acton, "A geometric-statistical approach toward neuron matching", *ISBI 2012*: 772-775.

Chapter 2

Background

In this chapter we present a detailed background survey of different techniques for segmenting single neurons from confocal microscopy. As we have mentioned earlier, we restrict ourselves to the case of detecting single neurons, and therefore techniques which fall under the multi neuron category, or are based on other imaging modalities (such as EM, two photon microscopy) are excluded from the discussion.

The different neuronal components (also popularly known as neuronal processes in biology) such as dendrites and axons, imaged via confocal microscope typically exhibit filamentous appearance. A majority of the neuron image analysis algorithms exploit the filamentous nature of the structures to devise tracing methodologies in presence of background noise and clutter. Vascular or filamentous/tubular structures occur in abundance in medical imaging. This includes, but are not restricted to, retinal blood vessels, blood vessels and arteries in the brain and other organs imaged via CTA, human arteries as seen in ultrasound images etc. A majority of these images suffer from poor contrast and background noise and thereby require a preprocessing step to enhance the vascular structures before performing segmentation.

Perhaps one of the most widely used and well known vessel enhancing method was proposed by Frangi *et al* [4]. In the recent years, there have been significant research to

develop robust algorithms to improve vessel detection from low contrast imagery. This includes 3D line filters due to Sato *et al.* [30], diffusion based methods [31] and steerable ridge detectors [32]. Vessel enhancement is an integral step in many tracing and segmentation algorithms, and will be discussed in significant detail in the next few chapters of this dissertation.

2.1 Neuron segmentation and tracing

The terms tracing and segmentation are often used interchangeably in the relevant literature. In most cases, tracing refers to the algorithms that focus on reconstruction of the neuron by identifying the centerline of the filaments. Segmentation algorithms, on the other hand are more traditional and aim at delineating the entire neuronal structure using traditional segmentation tools such as thresholding, morphological operations, connected component analysis, active contours, graph cuts etc [33]. The neuron tracing can be obtained from the segmented result by computing the centerline of the binary segments using skeletonization algorithms [33]. However, before we discuss the algorithms for neuron tracing, let us briefly talk about the biological motivation and imaging technique for data acquisition.

2.1.1 Challenges in image processing

Confocal microscopy is a popular method for imaging single neuron cells. The ability to visualize neurons at a single cell resolution is a significant step towards automated structure identification. Confocal laser scanning microscopes (CLSM) use focused laser beams to excite selectively tagged neurons (which are labeled with a florescent protein) to emit energy in form of photons, which are subsequently absorbed by a photodetector to create an image of the objects at the particular optical plane. Out of plane scatter is suppressed by the detector aperture, thus allowing sharper image formation. Information

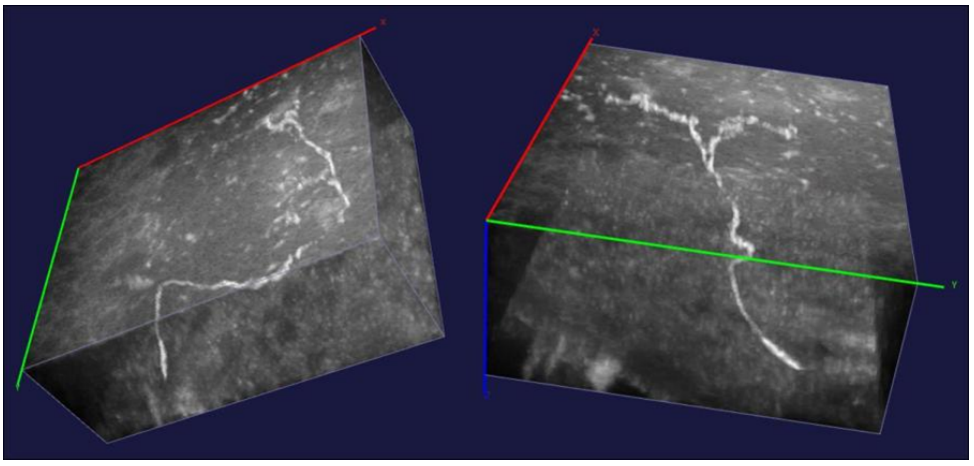


FIGURE 2.1: A 3D confocal microscopy image of Drosophila interneuron is shown from two viewing angles. The image is degraded by low contrast and inhomogeneous illumination of the neurite. Furthermore, presence of noise and background clutter deteriorates the image quality.

can be collected from different focal planes by altering the focal length of the system, thus creating multiple slices (z-stacks), which are eventually fused to form a 3D image of the specimen.

Like many other light microscopy techniques, neuron imaging using confocal microscopy comes with its own set of challenges. First, most images are low contrast and suffer from low signal to noise ratio. Photobleaching effect restricts the imaging time, which directly affects the signal intensity and contrast. Also, irregular photon emission from the filaments, especially those at greater depths, creates inhomogeneous signal intensity and renders a fragmented appearance of the neurites. Finally, the images are affected by photon noise and clutter resulting from non neuronal fluorescing tissues. All these aspects make image processing challenging and a suitable neuron tracer needs to manage these artifacts using sophisticated image processing. Fig. 2.1 shows an example of the imaging artifacts discussed here.

2.1.2 Neuron tracing strategies

In this section we are going to highlight some popular neuron tracing and segmentation works. Most of these tools are extensively used by biologists for segmentation of single

neurons. However, the list is definitely not exhaustive and more algorithms are being developed each day as the field continues to grow.

We can broadly categorize the neuron segmentation schemes in two basic approaches. The first set of methods use user defined (or automatically detected) initial seed points to perform tracing in a local exploratory manner. Such methods are popularly known as *tracing* algorithms or *tracers*, since the algorithms locally trace the neuron centerlines. The second category of algorithms avoid seed initialization and perform segmentation using a combination of local and global features. In the next few subsections, we will discuss a few popular neuron tracing and segmentation algorithms.

2.1.2.1 NeuronJ

One of the earlier works in neuron tracing was proposed by Meijering *et al.* [34]. The tracing algorithm, NeuronJ, is a 2D neuron tracer, which uses specialized steerable ridge detector kernels [32] to determine the evidence of a neuron filament, which the authors call *neuriteness*. A semi automated graph based tracing method *Livewire* [35] is then used to trace the neurite centerline, using evidence of neuriteness from the ridge detector response.

NeuronJ is widely used in the scientific community for 2D tracing, and it owes its popularity to its robustness and simplicity, and is available as a plugin for the open source, bio-imaging software ImageJ [36]. However, the method is custom made for 2D imagery only, and carrying out such user interactive tracing in 3D appears less trivial. Also, the algorithm relies significantly on user assistance, especially for determining branch points and neurite terminals. As a result, the applicability of this tool is somewhat limited for high throughput analysis.

2.1.2.2 Simple Neurite Tracer

Simple Neurite Tracer [37] is another open source, publicly available, semi-automated 3D neuron tracer. Simple Neurite Tracer (SNT) falls under the category of seed-based tracing algorithm, where the user selects a set of initial points on the neuron centerline. A graph is then created, with these seed points serving as the nodes of the graph, and the edge weights depending on their relative distance and the vessel intensity at the seed locations. A shortest path algorithm [38] is then deployed to compute the best possible path between the adjacent seeds.

Like many vessel tracing algorithms, SNT also uses vessel enhancement technique [4, 30] for determining the shortest path between the seeds. Similar to NeuronJ, SNT is also available as a plugin for ImageJ. However, the major drawback of SNT include the excessive use of human assistance (for seed detection, merge point identification etc.) for tracing. That being said, SNT is popular among biologists and is often used as a post processing platform for correcting or editing neuron reconstructions from automated tracers.

2.1.2.3 Vaa3D and associated algorithms

Vaa3D [39] is a neuron tracing software suit which is gaining significant popularity in the recent years. The C++ based tracing suit hosts a number of neuron tracing algorithms, both seed based and automated. One popular method, Graph Augmented Deformable model (GD model) was proposed by Peng *et al* [1]. The GD model performs neuron tracing based on some user chosen seeds, which are typically chosen as the neurite terminals. With these set of chosen locations, tracing is performed between the pairwise seeds using a deformable, shortest path approach. Fast and accurate segmentation is possible using the above mentioned approaches if the neuron morphology is simple and the image noise level is low. Recent algorithms proposed by the same group [40, 41] attempts to make the

tracing more user independent by initially performing an over segmentation by choosing a conservative threshold, followed by pruning to delete unwanted components.

2.1.2.4 k-Minimum Spanning Tree

Gonzalez *et al.* [42] introduced a graph theoretic technique to delineate the optimal neuronal tree from an initial set of seeds by computing a k-Minimum Spanning Tree. To automatically generate the seed points, the authors propose using a supervised learning methodology to determine potential neurite locations. This neuron probability map is importance sampled to generate a set of seeds, with special care taken to preserve the important locations such as tips and bifurcations.

Since the minimum spanning tree of a graph created from all the nodes can contain non neuronal clutter, the authors propose a novel solution by posing am optimization problem that finds the k^{th} minimum spanning tree. An approximate solution to this NP-hard problem is realized by minimizing a global energy function in a linear integer programming framework. Recently, there have been efforts to improve the accuracy of the algorithm, by using a different optimization criteria [43].

2.1.2.5 NeuronStudio

NeuronStudio [44] is a neuron segmentation software toolkit. Unlike the aforementioned methods, which are essentiially tracing techniques and rely on initial seed selection, NeuronStudio is an example of a traditional segmentation algorithm. The algorithm implemented is known as *voxel scooping* method [2]. It assumes that the neurites are locally tubular filaments and iteratively searches for voxel clusters in a manner similar to region growing. A pruning step is then deployed to eliminate spurious end nodes.

The biggest aspect of NeuronStudio is that it is relatively less dependent on user interaction, and segmentation only requires a single seed location to identify the starting position for region growing. However, the region growing algorithm becomes less stable

when the images are degraded by low contrast, and region leakage becomes an issue for processing.

2.1.2.6 Open-Curve Snake

The Open-Curve Snake method [45, 46] is implemented using open ended parametric active contours. This can be considered to be a tracing method, where the 1D curves are evolved along the vessel centerline to perform tracing. An external force field is suitably designed for curve evolution, which encourages the trace to lie along the neuron centerline. A preprocessing step based on tensor voting [47] is also used to enhance the neuron contrast and the snake initialization is performed by using graph cuts [48] to get an initial overestimate of the neurite locations. Combined with a post-processing step to eliminate false filaments, this method is efficient in segmenting neuronal structures from low SNR confocal stacks. However, due to the inability of parametric active contours to naturally handle topological changes such as object merging, neurite branch point detection depends requires a non-trivial post processing to determine snake merging at the junctions.

2.1.2.7 Tree2Tree

Tree2Tree [3] aims to solve the neuron segmentation problem in a graph theoretic framework. Unlike traditional seed selection approaches, where manually designated points are treated as the nodes of the graph, an initial segmentation algorithm is devised to produce a set of disjoint binary connected components. Connectivity between the components is analyzed based on their separating distance and orientation, which determines the weights of the graph edges to perform segmentation using a minimum spanning tree approach.

Although the primary contribution of Tree2Tree is to connect the fragmented neurite segments automatically, this connectivity analysis relies on heavily on the initialization.

Noise and clutter in the images create undesired artifacts in the global segmentation, resulting in loss of structural information. Moreover, linking the components based on their relative geometric orientation requires computation of the leaf-tangents from the object centerlines, which is sensitive to the irregularities of the neurite surface. One drawback of Tree2Tree is that the disjoint neighboring neurite fragments are connected by a splined curve, which may not coincide with the actual neurite location in the image. This aspect of Tree2Tree is addressed in Tree2Tree2 [21], which will be discussed in more details in Chapter 3.

2.1.2.8 Other methods

Apart from the above mentioned algorithms which are popular in the scientific community since the implementation is readily available, there are many more tracing methodologies that deserve mention. Al-Kofahi *et al.* [49] used the medial response of multiple directional templates to determine the direction to generate successive seed points along the neuron medial axis. This local tracing method shows good performance in high-contrast images, but requires continuity in the neuron branches for reliable segmentation. Other notable neuron tracers include the wavelet based algorithm due to [20], morphology and graph theory based technique of Chothani*et al.* [50] etc.

Other methods of segmentation include the use of probabilistic region merging [51], tracking based algorithms [52, 53] and active contours [54]. A survey of recent tools for neuron reconstruction is available due to Dr. Erik Meijering [17].

The medical imaging community has performed substantial research in developing algorithms to detect and segment filamentous shapes in non-microscopy medical images [55]. While not related directly to neuron tracing or segmentation, methods for segmenting vasculature have been extended or modified for tracing neurons, and thereby deserve mention.

The CURVES algorithm by Lorigo *et al.* [56] evolves a 1D curve along a 3D vessel centerline guided by the curvature of a 1D curve. Gooya *et al.* [57] developed an elegant and generalizable regularization methodology to enhance the performance of the popular geometric curve evolution methods. The method allows for anisotropic curve propagation which minimizes contour leakage when vessel edge information is weak. One apparent downside of this technique is that the ultimate solution somewhat depends on the shape of the initialized contour. Another recent work by Gooya *et al.* [58] generalizes the flux maximizing flow [59] on Riemann manifolds and uses a vessel enhancing tensor, which improves segmentation when edge detection is less accurate.

Shang *et al.* [60] developed a vessel tracing algorithm where wider vessels are first segmented using a region based criteria. Then the eigenvectors of the hessian matrix are utilized to derive a geometric flow equation to segment the thinner vessels. Santamaria-Pang *et al.* [19] use a multistage procedure for detection of tubular structures in multi-photon imagery, which includes a pre-filtering stage to identify the filaments based on supervised learning. However, this requires offline learning of the model parameters and prior knowledge about the vessel appearance information, which necessitates a set of accurate training examples and demands extensive human involvement to generate the ground truth.

2.2 Discussion

The neuron segmentation algorithms discussed in this chapter are classified either as local tracers, or global segmentation techniques. We hypothesize that seed based local algorithms are useful if the imaged neurons are not too complicated structurally. In such scenarios, where manual seed selection is easy, reliable segmentation can be obtained. However, since automatically choosing the correct set of seed points is still an open problem, it is difficult to use the above mentioned techniques for high throughput,

no intervention analysis. Also, since proper selection of seeds points is instrumental in these methods, the segmentation accuracy is sometimes compromised if a sub-optimal set of points is chosen. Furthermore, the connectivity analysis between the seeds assume uniform signal intensity, and noise and low contrast in the images may degrade the segmentation quality.

In contrast to the seed based local techniques, traditional segmentation approaches are more global, typically requiring an initial pre-processing of the image followed by a specialized segmentation step. Although a global approach may suffer from expensive computation, they are more suitable for neurite junction and end point detection. Typically, such methods rely on a four stage processing pipeline – enhancement, segmentation, centerline detection and post processing. However, despite being less dependent on user interaction, global segmentation algorithms should be adequately modeled to capture the finer details of neurite geometry, such as complex branching patterns, including twists and turns of filaments and should be capable of handling signal attenuation due to imaging artifacts.

Due to the aforementioned reasons, we refrain from using local seed based techniques for segmenting neurons. In the following chapters, three novel neuron segmentation methods are presented, which are completely automated and do not require manual identification of seed points for tracing.

Chapter 3

Graph Based Neuron Tracing

In this chapter we will discuss our first neuron tracing algorithm, which uses graph theoretic algorithms to perform tracing. The proposed algorithm, Tree2Tree-2 [21] extends the former neuron tracer Tree2Tree, which was proposed by Basu *et al.* [3]. Both Tree2Tree and Tree2Tree-2 are hybrid algorithms in a sense that they combine strategies of both segmentation based techniques and graph based tracing. The idea is to start with an initial set of neurite segments, which are possibly disjoint, and then identify the connectivity between them. In the next few sections, we will discuss the properties of Tree2Tree-2 in more detail.

3.1 Tree2Tree-2

The general workflow of Tree2Tree-2 consists of three major steps: image denoising and filament enhancement, initial segmentatation and connectivity analysis .

3.1.1 Hessian based vessel enhancement

Frangi's [4] method for enhancing vascular structures is based on analyzing the geometric features of tubular objects. The vesselness or tubularity measure at a point $\mathbf{x} \in \Omega$ in

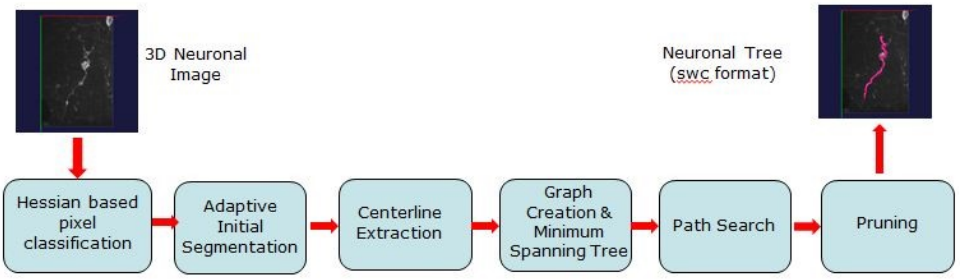


FIGURE 3.1: The working pipeline of Tree2Tree-2

the image $f : \Omega \to \mathbb{R}$ can be obtained by examining the hessian matrix of the gaussian smoothed image over a set of scales. The hessian of the d-dimensional image $f(\mathbf{x})$ at a position $\mathbf{x}$ and scale σ is the square matrix $\mathcal{H}_\sigma(\mathbf{x}) = [h]_{i,j}$ $(1 \leq i, j \leq d,\ \mathbf{x} \in \Omega)$ which is given by

$$h_{i,j} = \frac{\partial^2 G(\sigma)}{\partial x_i \partial x_j} * f(\mathbf{x}) \tag{3.1}$$

Here $\mathbf{x}$ is the d-dimensional vector $\mathbf{x} = (x_1, \ldots, x_d)^T$, $G(\sigma)$ is the zero mean normalized Gaussian kernel with variance σ^2. Here $d = 2$ or 3 for 2-D or 3-D images respectively.

At any position $\mathbf{x}$ on the image surface, the local curvature in the direction of the vector $\mathbf{v}$ is given by $\mathbf{v}^T \mathcal{H}_\sigma(\mathbf{x})\mathbf{v}$. If $\mathbf{v}$ is the direction along the vessel centerline (or major axis if we approximate a vessel locally as a cylinder), we expect the curvature to be negligible, assuming that the vessel intensity is homogeneous. On the other hand, in the directions orthogonal to the vessel axis, we encounter significantly higher curvature. Therefore, based on these curvature values, one can develop a discriminatory function (*vesselness*) to differentiate between vascular and non vascular objects.

3.1.1.1 Tubularity field

From the above discussion, at a position $\mathbf{x} \in \Omega$, a 3-D tubular structure can be characterized by three principal directions: (i) an axial direction along which the second derivative

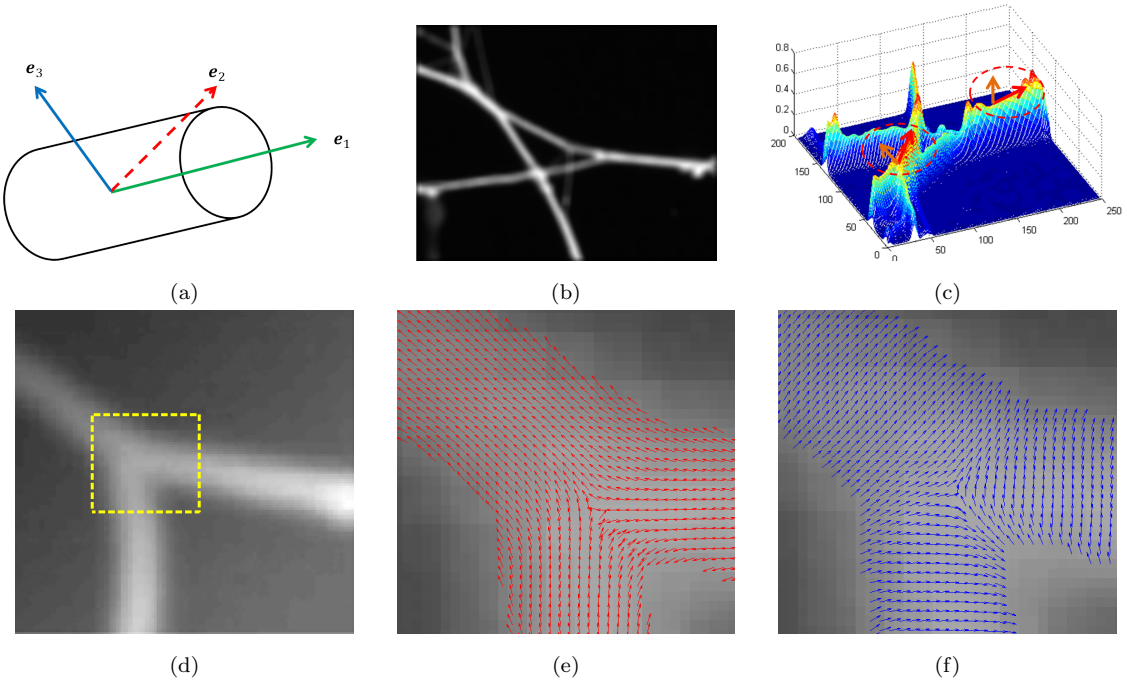


FIGURE 3.2: (a) Graphic demonstration of the tubularity fields. (b) An image of a neurite and (c) corresponding surface plot of the image. Vessel orientations are shown by the red and orange vectors. This figure is best viewed in color. (d) A real example of a vascular structure. (e) Axial field $\mathbf{e}_1$ and (f) Normal field $\mathbf{e}_2$ for the zoomed portion.

is negligible, and (ii) two orthogonal directions along which the second derivative magnitude is significant. These directions are given by the orthonormal set of eigenvectors $\{\mathbf{e}_1(\mathbf{x}), \mathbf{e}_2(\mathbf{x}), \mathbf{e}_3(\mathbf{x})\}$ of the hessian matrix(see Fig. 3.2(a)). The corresponding second derivative magnitudes or the curvature values are obtained from the respective eigenvalues as follows: $|\lambda_1(\mathbf{x})| \leq |\lambda_2(\mathbf{x})| \leq |\lambda_3(\mathbf{x})|$. It may be observed that for a filamentous voxel $\mathbf{x}$, the eigenvalues of its hessian matrix (computed at scale σ) should satisfy the following criteria:

$$\begin{aligned} |\lambda_1(\mathbf{x})| &\approx 0 \\ |\lambda_2(\mathbf{x})| \gg |\lambda_1(\mathbf{x})|,\ |\lambda_3(\mathbf{x})| &\gg |\lambda_1(\mathbf{x})| \\ |\lambda_2(\mathbf{x})| &\approx |\lambda_3(\mathbf{x})| \end{aligned} \tag{3.2}$$

Also, assuming that vascular filaments are brighter than the background, we have $\lambda_2(\mathbf{x}) < 0$ and $\lambda_3(\mathbf{x}) < 0$. A voxel that belongs to a vessel should satisfy the above mentioned

criteria. Based on that, there has been several vessel enhancement measures proposed in the literature [3,4]. For example, Basu *et al.* [3] suggests the following vesselness function N_σ at scale σ for enhancing the tubular structures:

$$N_\sigma(\mathbf{x}) = \begin{cases} \dfrac{|\lambda_1(\mathbf{x}) - \lambda_2(\mathbf{x})|^2}{|\lambda_1(\mathbf{x})||\lambda_2(\mathbf{x}) - \lambda_3(\mathbf{x})|} & \text{if } \lambda_2(\mathbf{x}), \lambda_3(\mathbf{x}) < 0 \\ 0 & \text{otherwise} \end{cases} \tag{3.3}$$

Also, since the filament thickness can vary, a multiscale analysis is essential. This can be evaluated by computing the maxima across a set of predefined scales, known as the scale space $S = \{\sigma_{min}, \ldots, \sigma_{max}\}$.

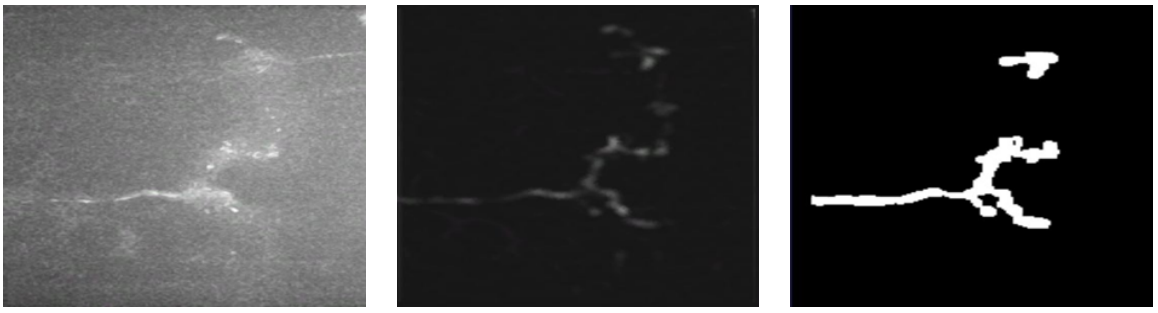
$$\sigma^*(\mathbf{x}) = \underset{\sigma \in S}{\operatorname{argmax}}\, N_\sigma(\mathbf{x}) \tag{3.4}$$

$$N(\mathbf{x}) = \max_{\sigma \in S} N_\sigma(\mathbf{x}) \tag{3.5}$$

The scale space vesselness function $N(\mathbf{x})$ assumes higher value at locations of local tubularity over non-filamentous positions. It should be noted that (3.5) yields evidence of the presence of vasculature by suppressing the non-filamentous structures, thus introducing a mechanism for dealing with noise and non tubular clutter.

3.1.2 Initial Segmentation

Once the raw confocal image is denoised and the vessel contrast improved using the aforementioned technique, the obtained vesselness function (3.5) is segmented to get an initial estimate of neurite locations. The major aspect of Tree2Tree-2 is to integrate the possibly disjoint components obtained in the segmentation step. Since the connectivity analysis step is non-generative, we choose a local segmentation technique using variational minimax [61] to get a possibly overestimated solution having multiple binary connected



(a) (b) (c)

FIGURE 3.3: Illustration of the enhancement and initial segmentation stage of Tree2Tree-2. (a) 3-D confocal image, (b) hessian enhanced image $N(\mathbf{x})$ and (c) binary image, showing the two disjoint connected components after thresholding.

components. The mathematical formulation of our segmentation framework is presented here.

Let $f : \Omega \mapsto \mathbb{R}$ be an image, and N denotes the vesselness function which is computed using (3.5). We define a threshold surface $s(\mathbf{x})$, $\mathbf{x} \in \Omega$ such that our initial segmentation result is given by the binary image $b : \Omega \mapsto \{0, 1\}$ which is defined as follows:

$$b(\mathbf{x}) = \begin{cases} 1 & \text{if } N(\mathbf{x}) > s(\mathbf{x}) \\ 0 & \text{otherwise} \end{cases} \tag{3.6}$$

The optimal threshold surface is computed by minimizing a convex energy functional $E(s, \alpha)$

$$E(s, \alpha) = \frac{1}{2}\sqrt{1-\alpha^2} \underbrace{\int_\Omega (N-s)^2 \left|\nabla N\right|^2 d\mathbf{x}}_{E_1} + \frac{1}{2}\alpha \underbrace{\int_\Omega |\nabla s|^2 d\mathbf{x}}_{E_2} \tag{3.7}$$

The first term in (3.7), E_1 is the data term, which computes the weighted mean square distance between the threshold function the vesselness function. The second term, E_2 is a smoothness enforcing term that restricts the threshold function to develop high gradients. The relative contribution of these two functionals is controlled by the parameter α.

The gradient of the vesselness map corresponds to the filament edges. Therefore,

the data term of the energy functional is significantly high at the filament edges, and the threshold function $s(\mathbf{x})$ should be close to $N(\mathbf{x})$ at edge position to reduce the cost functional in (3.7).

We seek to minimize (3.7) and since $E(s, \alpha)$ is convex with respect to $s(\mathbf{x})$, the minimizing function s^* is a global minima. Also, further analysis of (3.7) suggests that the functional is concave with respect to the scalar parameter α. It may be observed that the characteristics of E_1 and E_2 are conflicting in a sense that the minimizer of E_1 does not entirely satisfy E_2, and vice-versa. One solution to this parameter selection problem is to use minimax technique. In a minimax paradigm, the optimal threshold surface is the one which minimizes the worst case cost function. Mathematically, this is stated as:

$$\{s^*(\mathbf{x}), \alpha^*\} = \arg\max_{\alpha} \min_{s} E(s(\mathbf{x}), \alpha) \tag{3.8}$$

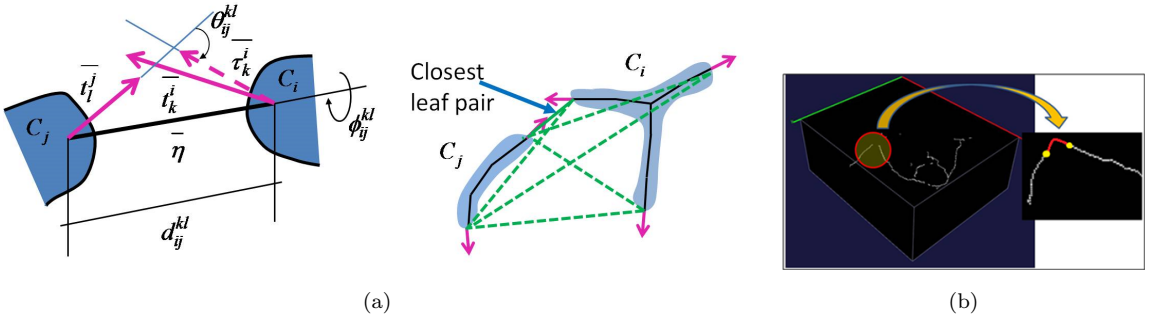
We solve (3.8) using variational calculus for s, and the obtained Euler-Lagrange equation is solved using gradient descent. We use alternating maximization to compute α. The solution is derived as follows. Please refer to the Appendix C for details of derivation.

$$\begin{aligned} \frac{\partial s}{\partial t} &= \sqrt{1-(\alpha^*)^2}|\nabla N|^2\,(N-s) + \alpha^* \nabla^2 s \\ \alpha^* &= \frac{E_2}{\sqrt{E_1^2 + E_2^2}} \end{aligned} \tag{3.9}$$

We solve (3.9) via discrete approximation of the partial differential equation, with Neumann boundary conditions. The optimal threshold surface is obtained at convergence, and then the initial segmentation $b(\mathbf{x})$ is derived using (3.6). The initial segmentation gives us an estimate of the neuron morphology. However, the imaged neurons may appear fragmented due to imaging artifacts. This can create multiple segments, instead of getting a single connected component (see Fig. 3.3(c)), some of which may not belong

to the original neuron. Therefore, to establish the global neural morphology, we introduce a method to analyze the connectivity between the different fragments, followed by a specialized pruning step to discard clutter.

3.1.3 Connectivity analysis and component linking



(a) (b)

FIGURE 3.4: (a) The calculation of orientation and distance between leaf pairs. 2 synthetic connected components C_i and C_j shown in blue. 6 pairs of leaf node connectivity (shown in dotted green) are tested for the closest distance between the components. The closest leaf pair is indicated. (b)Two disjoint components are joined by the global connectivity algorithm (shown in red). However, the actual path is unknown and is decided by PathSearch.

The primary contribution of Tree2Tree-2 is to analyze and determine proper connectivity between the disjoint segments (see Fig. 3.4). We adopt a graph theoretic approach to determine possible connections between the branches. Once this is established, the next step would be to physically link the components by a curve which traces the medial axis of the neurite filaments.

The initial segmentation procedure classifies the image voxels into neuronal (foreground) and non-neuronal (background) portions, as shown in Fig. 3.3(c). Since the obtained foreground segments may be disjoint, the next step is to connect the components to obtain a global graph, and eventually, the neuronal tree. To ensure proper connectivity, we adopt a two stage procedure.

Let us denote the set of disjoint components as $C = \{C_1, \ldots, C_n\}$. The first step involves a global graph connectivity algorithm that decides the connectivity between these components. The edge weights of the complete graph, created with the set of nodes $\{C\}$

are computed such that more probable connections have lower weights. Basu *et al.* [3] designed the edge weight function such that the weight D_{ij} is low when two components C_i and C_j are close according to the euclidean distance metric and they are oriented towards each other. Mathematically, this is can be stated in the following manner:

$$D_{ij} = \lambda dist(C_i, C_j) + (1 - \lambda) orientation(C_i, C_j) \tag{3.10}$$

Here $dist()$ is a function that computes the shortest euclidean distance between the terminals of C_i and C_j. The function $orientation()$ assumes a high value if the outward leaf tangent vectors of the connected components are facing in the same direction. An illustrative example is shown in Fig. 3.4(a). For detailed description, we refer the reader to the original paper [3].

A Minimum Spanning Tree (MST) of the global graph produces an initial neuronal tree. This method is computationally simple and provides efficient and accurate connectivity information between the components. The objective now is to detect the actual path that connects them in the image domain.

We devise a novel algorithm, PathSearch, which finds the physical connection (in the image domain) between the components which are deemed suitable for joining in the initial graph based analysis step.

3.1.4 PathSearch

The PathSearch algorithm to find the physical path between the terminal points of two disjoint components uses Dijkstra's method [38] to find the shortest path between the end points. Treating each voxel as a node in a graph, the edge weight between two neighboring voxels $\mathbf{x}$ and $\mathbf{y}$ is computed as

$$w(\mathbf{x}, \mathbf{y}) = \frac{2}{\mu(\mathbf{x}) + \mu(\mathbf{y})} \tag{3.11}$$

Non-neighbors are not connected by an edge. Here, we assume 26-connectivity of the voxels in 3D (8-connectivity in 2D). $\mu(\mathbf{x}), \mu(\mathbf{y})$ denote the medialness value at voxel positions $\mathbf{x}$ and $\mathbf{y}$, which assumes a higher value at voxels near the medial axis than those away from the medial axis [28][1]. Choice of this weight function ensures that two voxels on a neuron medial axis are connected by a lower edge weight than two arbitrary, non-neuronal voxels. With this edge weight defined on the graph, we now use Dijkstra's shortest path algorithm [38] to determine the path connecting the end points between two segments. To get a smooth connection, we further interpolate this path by fitting a cubic B-spline to this curve. The workflow of the PathSearch method can be summarized as follows:

1. Determine the connectivity between the end nodes of the disjoint binary components using relative orientation and euclidean distance between the components.

2. Compute the neuron tree using MST.

3. Compute a medialness map to produce evidence of a path along neuron medial axis (cf. [28]).

4. Join the end points (in the image domain) by computing the shortest path between them using suitable edge weight function.

5. Smooth the obtained trace by spline fitting.

3.2 Multiscale medialness map

In the previous section, we mentioned that the PathSearch algorithm joins the components via shortest path along the medial axis of the underlying neuron. This requires computing

[1]For implementation, only a subset of pixels in a window containing the two endpoints are considered, to make processing faster.

a *medialness* function $\mu(\mathbf{x})$ (see (3.11)), which assumes higher values at the object medial axis.

Finding medial axis for binary structures is a well studied problem in mathematical morphology [33]. However, the problem is non trivial for gray scale objects. In this section, we introduce a technique for computing a medialness map for non binary objects and we use the computed medialness value for neuron tracing via Tree2Tree-2.

3.2.1 Vector field convolution medialness

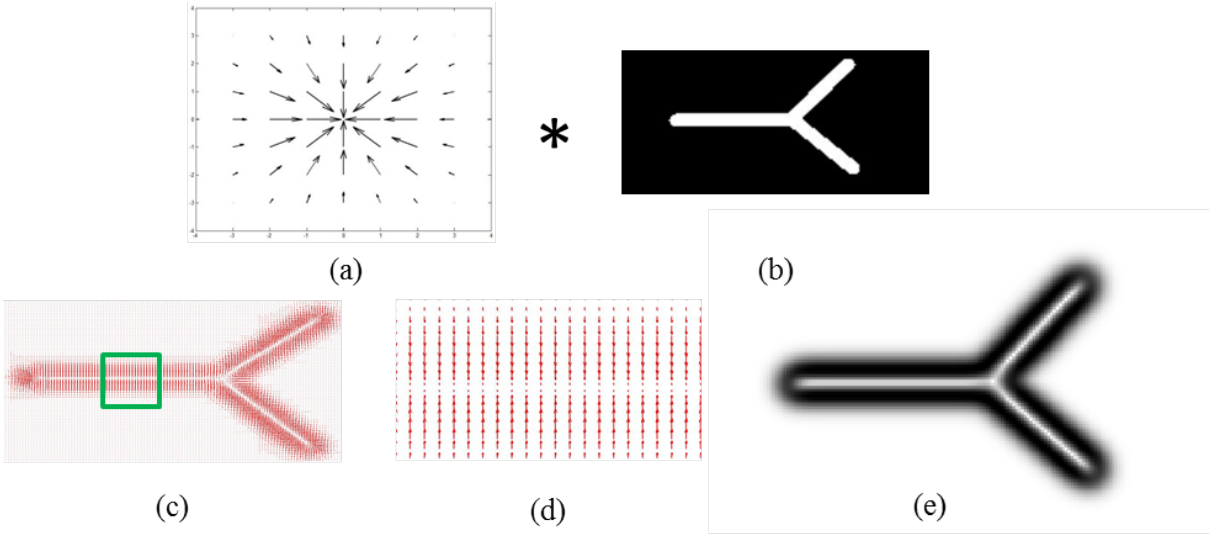


FIGURE 3.5: A discrete VFC kernel is shown in (a). (b) shows a synthetic tubular structure. (c) illustrates the vector fields after convolving (b) with the kernel, at some scale. An enhanced view is shown in (d), which reveals that the vectors cancel at the medial axis. The computed scale-space medialness map is shown in (c).

We propose a novel method to compute the medialness function of a grayscale object using vector field convolution (VFC) [11], which is computationally efficient and robust to noise. The vector field convolution kernel is defined as

$$K_\gamma(\mathbf{x}) = m_\gamma(\mathbf{x})\frac{\mathbf{x}}{||\mathbf{x}||} \tag{3.12}$$

The kernel is designed such the the vectors are directed towards the kernel center $\mathbf{x} = 0$. Magnitude of the vectors is determined by the magnitude function $m_\gamma(\mathbf{x})$. Choosing

$m_\gamma(\mathbf{p}) = \exp(-||\mathbf{p}||^2/\gamma^2)$, we create a kernel such that the influence of the vector field diminishes away from the center. This *kernel range* can be tuned by the scale parameter γ. A sample VFC kernel is shown for illustration in Fig. 3.5(a). At the correct scale, convolution of an image with the VFC kernel produces a zero magnitude vector at the points of symmetry of the grayscale object, since the vectors cancel each other. The points of symmetry of an object can be recognized by finding the local minima in the convolved image (assuming the object is brighter than the background). Mathematically, at a scale γ, the vector field obtained after convolving the image $f(\mathbf{x})$ with the kernel in (3.12) is given as $V_\gamma(\mathbf{x}) = f(\mathbf{x}) * K_\gamma(\mathbf{x})$. At this particular scale, the medialness is defined as

$$\mu_\gamma(\mathbf{x}) = 1 - \frac{|V_\gamma(\mathbf{x})| - \min|V_\gamma|}{\max|V_\gamma| - \min|V_\gamma|} \tag{3.13}$$

3.2.2 Multiscale VFC medialness

Since the neurite thickness varies, a multiscale approach is required to obtain a scale invariant medialness function. Let Θ be the scale space, which captures thickness of the objects of interest. In order to evaluate the correct medialness response over the scale space, we observe that the medialness response of a non-medial point diminishes with increasing scale, whereas, the response is significant for the medial points for each scale. The scale space medialness map is computed as

$$\mu(\mathbf{x}) = \frac{1}{|\Theta|} \sum_{\gamma \in \Theta} \mu_\gamma(\mathbf{x}) \tag{3.14}$$

Fig. 3.5 illustrates the medial map computation on a synthetic tubular object. Fig. 3.5(c) shows the vector field, which is generated by convolving the image (b) with the VFC kernel (at a fixed scale). The scale space medialness map is shown in Fig. 3.5(e).

3.3 Computing the neuronal tree

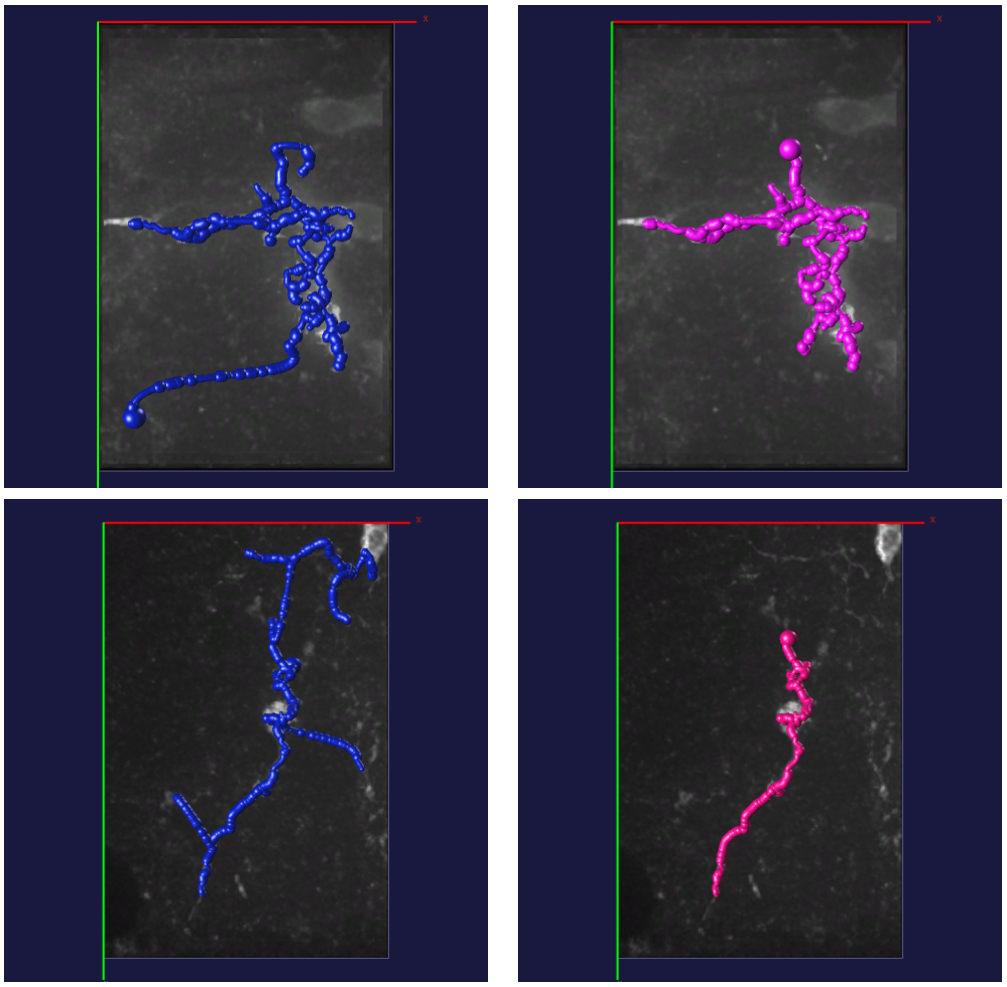
To summarize Tree2Tree-2, we first compute a global neuron tree using MST, where the connectivity information between the terminals of the neurite fragments are derived based on the relative geometry of the components. Then, PathSearch is used to physically connect the endpoints, by finding the shortest path between the terminals, which adheres to the medial axis of the neurite filaments. Let us designate the global tree obtained using MST and PathSearch as $\Gamma = \{E, V, W\}$. Here $E = \{e\}, V = \{v_{ij}\}$ and $W = \{w_{ij}\}$ denote the sets of edges, nodes and edge weights of the global tree. The nodes $v \in V$ correspond to the individual segments obtained after initial segmentation, and the edges $e_{ij} \in E$ between the nodes (v_i, v_j) are computed via PathSearch. To assign the weight $w(e_{ij}) \in W$ to an edge e_{ij}, we compute the sum of the edge weights of the shortest path between v_i and v_j. This ensures that false edges have a higher path weight than the true connections.

However, since the initial segmentation step may result in over segmentation, we may eventually get undesired nodes in the neuron tree due to the presence of clutter. This leads to the final step of Tree2Tree-2, where a pruning methodology is deployed to eliminate the false nodes.

3.3.1 Tree pruning

We designate the volume (area) occupied by a tree in the 3D (2D) binary image as $\psi(\Gamma)$. Our assumption is that the neuronal portions occupy more volume than the clutter, and the cost of connecting them via PathSearch is more compared to a true node. We devise the following method to prune the tree:

Let $e \in E$ be an edge in the tree, i.e.$w(e) \geq w(e')$, $\forall e \in E$. Removing this edge from the tree Γ creates two disjoint subtrees Γ_1 and Γ_2. Let us denote $\Gamma^* = \arg\min_{\Gamma_j} \psi(\Gamma_j)$.



(a) (b)

FIGURE 3.6: Results of graph pruning. (a) A global neuron tree using MST+PathSearch. (b) The pruned tree with the false nodes removed.

From standard graph theory, the graph $\Gamma \setminus \Gamma^*$ that results from removing the subtree Γ^* from the global tree Γ, is also a subtree.

Our strategy for pruning the tree is as follows. We select a scalar variable $\rho \in [0, 1]$, which may be defined by the user. The subtree Γ^* is removed from the tree if $\frac{\psi(\Gamma^*)}{\psi(\Gamma)} \leq \rho$. The pruning process is repeated till no subtree can be further removed from the tree. Using this strategy, a subtree is deleted if it is connected by a significantly high cost path (which signifies higher possibility of a false connection), and the volume occupied by the subtree is small enough compared to the overall volume (since the area/volume of the clutter is significantly lower than the neurites). Typically, we use a low value of $\rho(\approx 0.1)$,

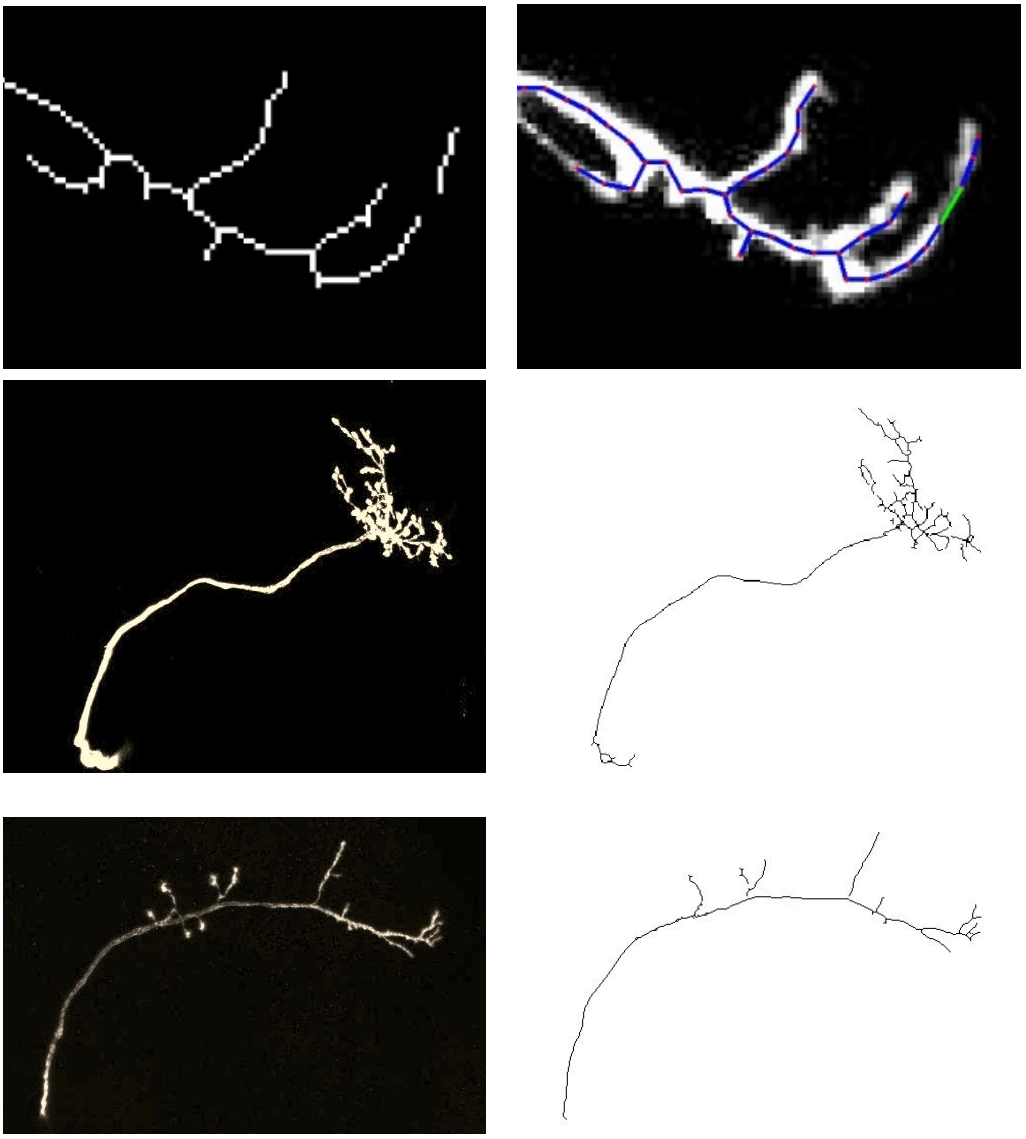


FIGURE 3.7: The top row shows an example how a broken link is joined by Tree2Tree-2 in 2D. The resulting tracing is shown in the second column and the detected connectivity is shown in green. Rows 2 and 3 show Tree2Tree-2 tracing results on 2D maximum intensity projection neuron images. The traced results are shown in the second column.

which encourages removal of clutter over a neuron segment (Fig. 3.6).

3.4 Results

Tree2Tree-2 is a tracing algorithm, applicable to both 2D and 3D images. We first present a few qualitative tracing results on 2D images. The 2D images are obtained using maximum intensity projection of the 3D stacks along the Z-axis (depth). Fig. 3.7 shows

the tracing results of Tree2Tree-2 on a few 2D images. Qualitative results for 3D tracing are shown in Fig. 3.8. The manual tracing results are shown in the first column, and are overlaid on the original images using the Vaa3D visualization software [39]. Tracing results due to Tree2Tree-2 are shown in the second column in blue.

To quantify the performance of Tree2Tree-2, we use the normalized Mean Absolute Error (NMAE) [3] of the traced neuron, with respect to the manually annotated tracings. Preliminary experiments were performed on a batch of ten 3D stacks, which contained noisy images of Drosophila neurites. We observed an average NMAE of 1.01%, as compared to 1.97% for that of its precursor algorithm Tree2Tree [3]. The improvement in performance is largely due to the fact that Tree2Tree-2 uses PathSearch to compute the physical path between two objects, where Tree2Tree uses splines to interpolate the positions between two fragments. Also, the proposed tree pruning method in Tree2Tree-2 is simpler and more efficient compared to the alpha-beta pruning step in Tree2Tree.

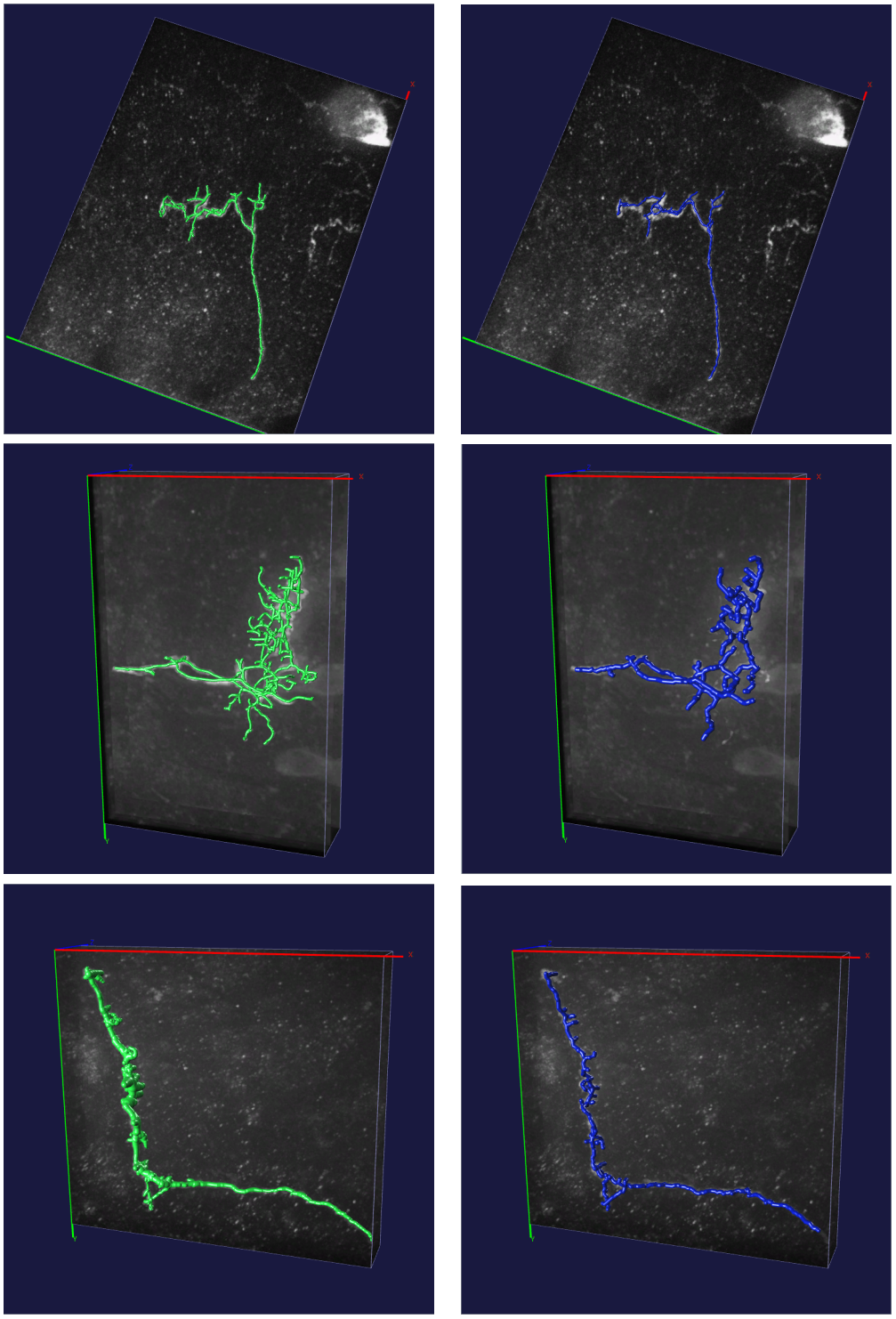


FIGURE 3.8: The first column shows manual tracing results(green). 3D tracing results of Tree2Tree-2 are shown in the second column (blue). The splined centerline is plotted on the original 3D confocal images of the Drosophila larva.

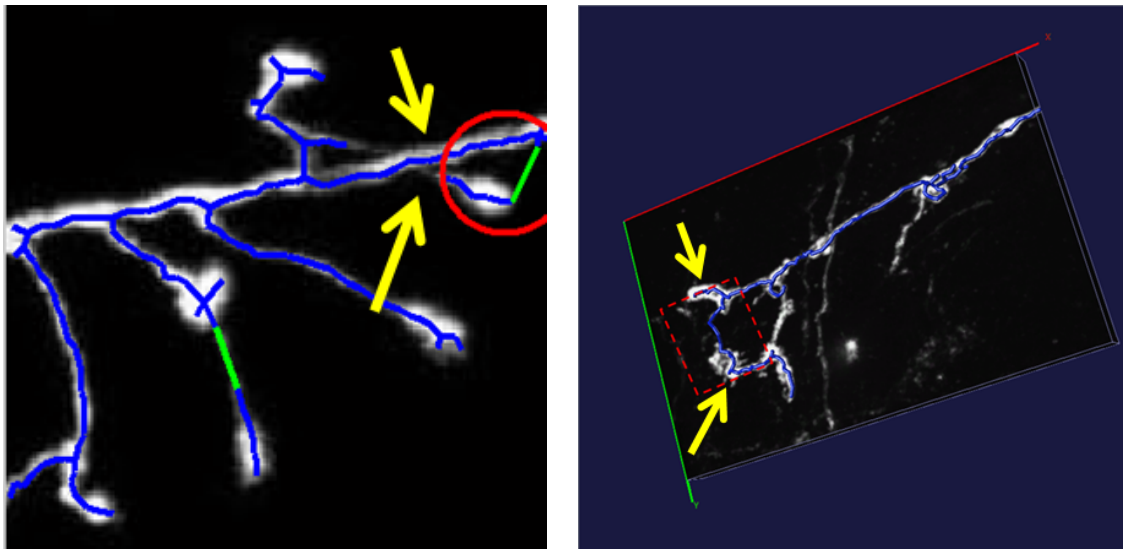
3.5 Discussion

The proposed algorithm, Tree2Tree-2 is essentially a neuron tracing method which leverages the capabilities of graph theoretic algorithms to identify the centerlines of the neuron filaments. The primary contribution of Tree2Tree-2 is to determine the connectivity between the different binary objects, which are obtained via initial segmentation of the neurites. From that perspective, Tree2Tree-2 can be considered an intelligent component analysis algorithm. Using simple graph based methods, we are able to compute the centerlines of neurites with considerable accuracy, given that the structural complexity is not significant. The initial segmentation step requires few parameters, and therefore helps automating the process. Moreover, Tree2Tree-2 revisits the component connectivity issue from the image domain perspective as well, once the initial connectivity information is obtained using graphs. This allows us to accurately connect the neurites along physical paths, as opposed to interpolating them in Tree2Tree. Also, since efficient algorithms exist to compute MST and shortest paths, Tree2Tree-2 is computationally efficient.

However, on the downside, Tree2Tree-2 relies on the initial segmentation step. If components are lost during thresholding, the algorithm is unable to interpolate the neurite structure. Therefore, it is preferable that the initial segmentation overestimates the solution.

However, such under segmentation can result in multiple binary components, which reduces the efficacy of the graph based connectivity analysis portion of Tree2Tree-2. In fact, since the neurite connections are analyzed only between the end points of the objects, it does not accommodate all possible cases of discontinuities. As a result, structural complexity of the filaments, as well as the number of binary objects in the solution degrades the tracing performance by predicting false edges between the fragments. Two examples are shown in Fig. 3.9. In Fig. 3.9(a), we demonstrate a 2D example, where the initial segmentation resulted in the tracing in blue. The green lines show the links

predicted by Tree2Tree-2. The region highlighted by the red circle shows a situation, where the component connectivity cannot be resolved by analyzing the end points of the fragments. However, since Tree2Tree-2 cannot handle discontinuities which are not between the object termini, a false branch is predicted. A similar result is shown in (b) for a 3D image, where Tree2Tree-2 forces the objects to be connected between the two terminals. While such a branch can be removed by pruning the tree, it would result in



(a) (b)

FIGURE 3.9: Examples of false edge prediction by Tree2Tree-2. In both examples, error in connectivity results from Tree2Tree-2's inability to handle discontinuities which are not between the end points of the objects. Improper branch connections are highlighted by yellow arrows. Regions enclosed by the red curves show the regions of interests.

the removal of a subtree, which contains a real neurite portion. Therefore, investing in a pruning strategy is not the most effective way to solve this problem. We hypothesize that such connectivity related issues can be better handled if the segmentation methodology is naturally able to adapt to the object topology. For example, so far we have been interested in determining the appropriate connections between the objects; however, a segmentation method which does not require such explicit connectivity analysis would fare better in circumstances where the neuron structures are complicated. This motivates us to use geometric deformable models [62, 63] for segmenting complicated structures like neurons. In such a paradigm the object topology is handled automatically, and if the model is

appropriately designed, there is no need to assess the connectivity information based on geometrical position of the objects. In the next chapter, we present a brief overview of geometric active contours and surfaces for segmentation, and we formalize a procedure to use them for our application in the following chapters.

Chapter 4

Geometric Active Contours

Object detection and image segmentation are arguably one of the more investigated fields in computer vision and image analysis. Although in certain cases the terms detection and segmentation are used almost synonymously, there exists subtle differences between them; an object detection algorithm can be deemed successful if it can effectively determine and localize the object of interest in the image. Segmentation, however, requires finer analysis. In segmentation problems, one is more concerned with the morphological properties of the object such as shape, connectivity etc. and therefore, a crude detection is not always sufficient. One may think of object detection as the first step for segmentation. In fact, many segmentation methods allow the user to perform an initial detection, either automatically or semi-automatically. The segmentation algorithm then starts from the initialized position and eventually converges when the object boundaries are captured.

Active contour based methods (a.k.a. *snakes*) fall under the category of segmentation algorithms which assume the image to be a continuous and possibly differentiable function. The basic reason for such continuous domain modeling is that the mathematical formulation can be done with the help of continuous domain calculus, which is remarkably strong and well established. The continuous model is eventually discretized, using tools from numerical analysis for practical implementation.

4.1 Framework for contour propagation

Let $f(\mathbf{x}) : \Omega \mapsto \mathbb{R}$ be a continuous image function, $\forall \mathbf{x} \in \Omega$. Traditionally, the domain $\Omega \subseteq \mathbb{R}^2$ or $\Omega \subset \mathbb{R}^3$ for 2D or 3D cases respectively. In the 3D case, we have *active surfaces* instead of active contours and the formulations are modified accordingly. However, most of the mathematics and models are consistent between 2D and 3D applications and therefore, for the sake of simplicity, we will restrict ourselves to the 2D case for basic analysis in this chapter. In the next chapter, where we introduce an explicit 3D model, we will discuss the theories for the 3D case separately. Also, we will only discuss closed curves in this chapter. Implementing open ended curves in a geometric framework is nontrivial and demands separate and elaborate treatment.

Considering we have a 2D image $f(\mathbf{x})$, $\mathbf{x} = (x, y) \in \Omega \subseteq \mathbb{R}^2$, we define a parametric curve $\mathbf{C}(p) = \{\mathbf{x}(p)\}$[1], where the parameter $p \in [0, 1]$. To perform segmentation, an initial contour is deformed under the influence of a force field. Mathematically, one can write the curve evolution equation as $\mathbf{C}_t(p) = \mathbf{F}$. Here $\mathbf{F}$ is the *curve velocity*, which is also known as *force vector*. Decomposing the velocity in the normal and tangential direction, we derive the general curve evolution equation as follows:

$$\mathbf{C}_t(p) = F_n(p)\mathbf{n}(p) + F_t(p)\mathbf{t}(p) \tag{4.1}$$

Here $\mathbf{n}(p)$ and $\mathbf{t}(p)$ are the unit outward normal and the tangent vectors to the curve respectively at $\mathbf{x}(p)$. For the sake of brevity, we will drop the implied parameter p from future equations. The above equation defines a model for the propagation of the curve. The speed terms F_n and F_t contribute to the curve motion in the normal and tangential directions respectively. Since the tangential component does not explicitly contribute to the motion, but only results in a re-parametrization of the curve, (4.1) may be modified

[1] $\mathbf{C}_t = \dfrac{\partial \mathbf{C}}{\partial t}$, where t is the parameter denoting pseudo time.

as follows:

$$\mathbf{C}_t = F\mathbf{n} \tag{4.2}$$

The major engineering issue is to come up with a proper *speed function* or *evolution force* term F. In fact, active contour based segmentation methods are primarily concerned with the development of a problem specific, suitable evolution force term. Before we go into the analysis and implementation details, let us discuss a few popular motion models for snakes.

4.2 Motion models for snakes

In this section, we will review some popular snake motion models. We will discuss the different choices of the force function and their effect on the curve motion.

4.2.1 Constant speed evolution

This is the most basic active contour model, where the speed function is a constant scalar. The motion model can be stated as follows:

$$\mathbf{C}_t = c_0\,\mathbf{n} \tag{4.3}$$

Here, the speed function is a constant scalar c_0 everywhere and the curve moves in the direction of the normal (or in the opposite direction if $c_0 < 0$) with a constant speed. One physical example of such a motion model is that of wave front propagation according to Huygen's principle. Depending whether c_0 is positive or negative, such a motion model either inflates or deflates the curve.

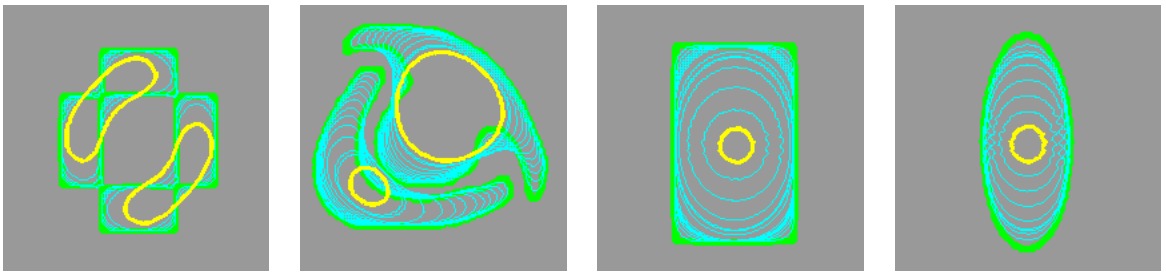


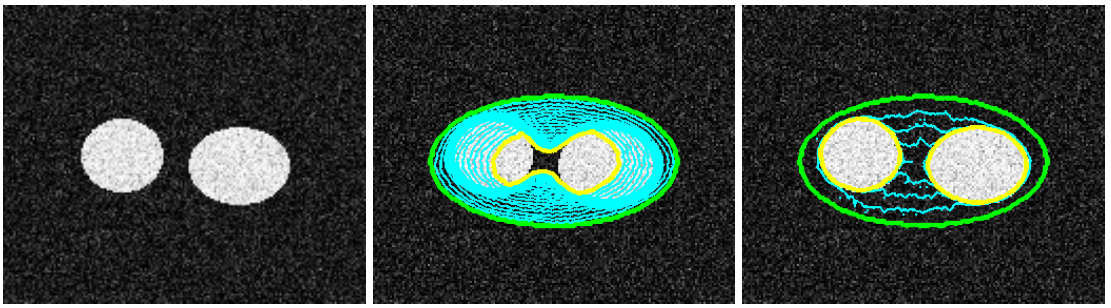
FIGURE 4.1: Example of motion by mean curvature. Initial curve is shown in green and the result after 500 iterations is shown in yellow. Intermediate stages of curve evolution are shown in cyan.

4.2.2 Curvature based motion

In (4.3), each point on the snake would move with the same speed. The curvature based motion gives higher priority to the snake regions with high curvature. This inhibits the snake to develop an irregular shape during evolution. If $\kappa(p)$ be the curvature of $\mathbf{C}(p)$ at positions $\mathbf{x}(p)$, the curvature based motion model is stated as follows:

$$\mathbf{C}_t = \kappa\, \mathbf{n} \tag{4.4}$$

The curvature can be computed as $\kappa = -\text{div}\,(\mathbf{n})$. The curvature based motion model has nice geometric interpretations. It can be shown that the 2D version of mean curvature motion results in minimizing the total euclidean length of the contour [64]. As a result, this model is often used in conjunction with other motion models for regularizing the shape of the active contour. An illustrative example is shown in Fig. 4.1. For 3D applications, we have a closed surface instead of a curve. An area minimizing regularizer can be obtained by replacing the curvature term κ in 2D by H which is the sum of the principal curvatures of the surface in 3D [65]. However, unlike the 2D case, mean curvature motion is not the optimal area minimizing flow and there has been other models proposed in the literature [65, 66].



(a) original image (b) Malladi-Sethian (c) GAC

FIGURE 4.2: Segmentation performance of geodesic active contour model over Malladi-Sethian's method. The initial curve is shown in green and the final contour in yellow. Intermediate steps are shown in cyan.

4.2.3 Malladi-Sethian model

The above mentioned motion models are not really suitable for object segmentation, since neither (4.3) or (4.4) consider any information from the image. Malladi and Sethian [67] introduced a flow based on the image model for object segmentation based on the gradient of the image. If $g(|\nabla f|)$ is a function which assumes a high value at the homogeneous portions of the image and significantly low value(≈ 0) at the edges. One particular realization may be computed as $g = 1/(1 + |\nabla f|^q)$. With this definition, the curve motion equation is realized as:

$$\mathbf{C}_t = g(|\nabla f|)(c_0 + \kappa)\,\mathbf{n} \tag{4.5}$$

The function $g(.)$ acts as an edge indicator, by slowing down the speed of the snake when it approaches an edge, characterized by high gradient value. As before, the sign of the scalar c_0 dictates whether the snake moves outward or inward. The curvature based term imparts smoothness to the solution by disallowing irregular contour shapes.

4.2.4 Geodesic Active Contour (GAC)

Caselles et al. [68] realized a drawback of Malladi's model which restricted its applicability in certain scenarios. This is because the function $g(.)$ does not stop the snake propagation

at the edges, but merely slows it down. Therefore, if the gradient magnitude at the edges are not significantly high, the propagating curve may not converge at the boundary but continues its motion and eventually either collapses or diverges. To mitigate this issue, the geodesic active contour model was proposed, which is mathematically expressed as:

$$\mathbf{C}_t = g(|\nabla f|)(c_0 + \kappa)\,\mathbf{n} - \langle \nabla g, \mathbf{n} \rangle\,\mathbf{n} \tag{4.6}$$

The first part of (4.6) is similar to that of Malladi's model. The second term improves segmentation by incorporating a speed term which attracts the evolving curve towards the edge. This results in an additional component which halts the snake at the object boundaries. This is particularly helpful, since the function $g(.)$ does not stop the curve from leaking across the edges; it merely slows down the propagation speed. With the additional attraction term, the moving contour is attracted toward the edges and forces the curve to stop moving (see Fig. 4.2).

This model can be interpreted as a flow that minimizes the geodesic curve length instead of the euclidean length, which makes it a geodesic length minimizing flow. For further detail, we refer the reader to the original paper [68].

4.3 Implementation using level sets

Proper model selection is the primary criteria for designing a deformable contour based segmentation algorithm. With a model already developed, the next challenge is to implement that in a computationally feasible manner. This is where two subtypes of active contours emerge: *parametric* and *geometric*.

One way to implement the curve evolution equation (4.2) is via discrete parametrization of the curve. The curve is represented by a set of finite number of contour pixels (or *snaxels*), and its evolution is computed by explicitly calculating the deformation forces at

these contour positions. Such algorithms fall under the category of parametric active contours and there have been significant research in this domain, involving both theoretical aspects [69–75] and practical applications [55, 76–79].

Parametric active contour methods are generally topology preserving, i.e., the topology of the evolving contour remains same during the motion. While this property may be desired in certain applications (e.g. tracking rigid objects), some applications demand a methodology which is topology adaptive. For example, when multiple objects are present, one may wish to design an active contour model, which is capable of detecting each object despite starting from a single initialization. Unfortunately, parametric methods are incapable of handling changes in topology, unless specialized algorithms are developed to perform contour merging or splitting.

4.3.1 Geometric active contour

Osher and Sethian [62] proposed an implicit curve evolution method which makes the curve adaptive to the topology of the objects. This set of algorithms, popularly known as level set methods or geometric active contours represent the curve $\mathbf{C}(\mathbf{x}, t)$ as the zero level set of a Lipschitz continuous function ϕ. With this representation, the curve evolution is obtained explicitly, by deforming the functional ϕ instead of calculating the forces on the discretized contour positions. The function $\phi : \Omega \times \mathbb{R}^+ \mapsto \mathbb{R}$ is defined such that the active contour is given by $\mathbf{C}(\mathbf{x}, t) = \{\mathbf{x} : \phi(\mathbf{x}, t) = 0\}\ \forall t \in \mathbb{R}^+$. Furthermore, if $\Gamma_{in} \subseteq \Omega$ and $\Gamma_{out} \subseteq \Omega$ denote the regions inside and outside the zero level set of ϕ, we have $\phi(\mathbf{x}) \geq 0\ \forall \mathbf{x} \in \Gamma_{in}$ and $\phi(\mathbf{x}) < 0\ \forall \mathbf{x} \in \Gamma_{out}$.

The advantage of using the embedding function ϕ is that all operations are carried out on ϕ and the evolving contour is represented by the zero level set of ϕ. Using this implicit representation for the curve enjoys the benefit of topological adaptivity; the

embedding function has natural ability of merging and splitting – which subsequently allows the zero level set to merge and split without the use of specialized schemes.

Naturally, the next question is how to modify (4.2) so that the implicit motion can be accommodated? As it turns out, it is trivial to develop a relationship between the explicit motion model and the implicit model. Assuming that the curve motion equation is given by (4.2), the explicit motion model is given by

$$\phi_t + \langle \mathbf{C}_t, \nabla\phi \rangle = 0 \tag{4.7}$$

The outward normal vector and curvature are computed as

$$\mathbf{n} = -\frac{\nabla\phi}{|\nabla\phi|} \tag{4.8}$$

$$\kappa = \operatorname{div}\left(\frac{\nabla\phi}{|\nabla\phi|}\right) \tag{4.9}$$

For derivation details, we refer the reader to the paper by Cassellas *et al* [68]. Now that we have obtained an equivalence between the implicit and explicit representations, we can modify the curve motion models in the level set framework.

Explicit curve motion equation (4.7) is popularly known as *geometric flow* equation. The benefit of the geometric model is its topological flexibility. Such models are efficient in segmenting multiple objects and have been used for numerous applications in image analysis. Implementation of the geometric flow is performed by discretizing the function

TABLE 4.1: Equivalent geometric flow equations using level set representation.

Name	**Curve motion model**	**Geometric model**
Constant speed motion	$\mathbf{C}_t = c_0\,\mathbf{n}$	$\phi_t = c_0\lvert\nabla\phi\rvert$
Mean curvature motion	$\mathbf{C}_t = \kappa\,\mathbf{n}$	$\phi_t = \operatorname{div}\left(\frac{\nabla\phi}{\lvert\nabla\phi\rvert}\right)\lvert\nabla\phi\rvert$
Malladi-Sethian model	$\mathbf{C}_t = g\,(c_0 + \kappa)\,\mathbf{n}$	$\phi_t = g\,(c_0 + \kappa)\,\lvert\nabla\phi\rvert$
Geodesic active contour	$\mathbf{C}_t = g\kappa\,\mathbf{n} - \langle\nabla g, \mathbf{n}\rangle\,\mathbf{n}$	$\phi_t = \operatorname{div}\left(g\frac{\nabla\phi}{\lvert\nabla\phi\rvert}\right)\lvert\nabla\phi\rvert$

ϕ over an uniform grid and using tools from numerical methods to solve the partial differential equation.

It is desirable that the embedding function is a differentiable function which is positive inside the zero level set and negative outside it. Also, since the zero level set of ϕ gives the position of the contour at each step of iteration, it is important to maintain the property of ϕ during the flow. This is performed by a process known as *reinitialization* which allows ϕ to retain the properties of a geometric embedding function and prevents instability. Several reinitialization techniques have been proposed in the literature including pde based methods and variational methods [62, 63, 80, 81].

While topology adaptability is a feature of geometric models which is unavailable with the parametric setup, one loses out in terms of computational time. While parametric models operate on a set of discrete points, the geometric models demand the entire function ϕ to be updated at each interval. There has been concerted efforts to reduce the computational time of such methods which includes narrow band methods (where update is performed only at positions near the zero level sets), fast marching and multigrid methods [67, 82–85].

4.4 Variational active contours

In the previous sections, we have described active contour models in a top down fashion. An equation for propagating the curve is first established (see eq. (4.1)) and then implicit curve motion is obtained by using an embedding functional. Designing such models generally involve computing a suitable speed function F (see eq. (4.2)). In many cases, it is convenient to design a solution by finding a suitable speed function or force function. This force function typically consists of a smoothness invoking part and an image based component that attracts the level set to the regions of high gradient in the image [63, 67]. Consequently, such techniques are categorized as *edge based* techniques.

However, there are examples where object segmentation may be performed as a (possibly local) solution to a suitable optimization problem. A popular example is that of region based segmentation framework proposed by Mumford and Shah [86]. The authors introduced a segmentation methodology which attempts to cluster the image into sets of foreground and background pixels based on their gray level intensity. The variational segmentation methodology can be mathematically stated as follows:

$$\phi^* = \underset{\phi}{\operatorname{argmin}}\, \mathcal{E}(\phi) \tag{4.10}$$

$$\phi_t = -\nabla_\phi\, \mathcal{E}(\phi) \tag{4.11}$$

The zero level sets of the local minimizer ϕ^* of an energy functional $\mathcal{E}(\phi)$ corresponds to the detected object boundaries (eq. (4.10)). Generally, a gradient descent based algorithm is deployed to find the local minima of the functional and the solution is computed iteratively. The gradient descent equation (4.11) corresponds to the curve propagation partial differential equation (similar to (4.2)) for the variational scheme which is derived using variational calculus [87]. This equation is often referred to as *variational gradient flow equation.*

The variational problem requires specification of the energy functional $\mathcal{E}(\phi)$. Traditionally, an energy function consists of two terms– a regularization term $\mathcal{E}_r(\phi)$ which generates smooth solution and another data term $\mathcal{E}_d(\phi, f(\mathbf{x}))$ which usually incorporates image based features for segmentation [88–91]. In addition, segmentation performance can be significantly improved by incorporating prior information about the object shape . Variational formulations provide an elegant way to introduce such priors in form of an additive energy term and may be introduced both as a hard prior [57, 92–94] or a soft prior [95].

4.4.1 Variational contour regularization

Here we discuss two popularly used energy functionals used for the purpose of obtaining a smooth active contour. One way to regularize the shape of the contour is by restricting its total length. This is performed by minimizing eq. (4.12), and the gradient flow equation is obtained by deriving the Euler-Lagrange equation and using gradient descent for local solution. This is shown in eq. (4.13).

$$\mathcal{E}_r^{(1)} = \int_\Omega |\nabla H(\phi)| d\mathbf{x} \tag{4.12}$$

$$\phi_t = \text{div}\left(\frac{\nabla\phi}{|\nabla\phi|}\right)\delta(\phi) \tag{4.13}$$

Here $H(.)$ is the Heaviside function defined such that $H(y) = 1\ \forall y >= 0$ and $H(y) = 0$ otherwise. The Heaviside function serves as an indicator function for the area inside the zero level set of ϕ. $\delta(.)$ is the Dirac delta function. Since the Heaviside function is not differentiable, for practical purposes the following regularized versions of the Heaviside and Dirac functions are used [89].

$$H_\epsilon(q) = \frac{1}{2}\left(1 + \frac{2}{\pi}\tan^{-1}\left(\frac{q}{\epsilon}\right)\right) \tag{4.14}$$

$$\delta_\epsilon(q) = \frac{d}{dq}H_\epsilon(q) \tag{4.15}$$

Analyzing eq. (4.12), it is not difficult to comprehend that the right hand side of the equation corresponds to the total length of the zero level set of ϕ. Therefore, eq. (4.13) represents the curve evolution equation such that the length of the curve is locally minimized. One can find a similarity between the length shortening gradient flow (4.13) and the mean curvature motion in (4.4), where the expressions differ by the multiplicative terms. In fact, if only a narrow band around the zero level contour is considered, one can find that both these formulations lead to almost identical solutions that regularizes the

curve via length minimization.

Another regularizer popularly used for getting a smooth solution is obtained by minimizing the total area inside the contour. The area minimizing functional and the corresponding gradient flow equation are computed as follows:

$$\mathcal{E}_r^{(2)} = \int_\Omega H_\epsilon(\phi) d\mathbf{x} \tag{4.16}$$

$$\phi_t = -\delta_\epsilon(\phi) \tag{4.17}$$

The regularizing energy functional contributes only to the smoothness of the solution. The regularizers are generally not dependent on image features. However, to perform segmentation, one needs to engineer a suitable data term that incorporates image based information (such as intensity, edge strength, texture etc.) to drive the level sets toward the desired object boundaries.

4.4.2 Chan-Vese's segmentation model

Chan and Vese [89] proposed a level set formulation to minimize the Mumford Shah functional [86] for segmentation. The Chan-Vese framework models the image as a set of constant illumination regions and performs a two-class segmentation by computing the optimal partition which satisfies the constant illumination constraint. The authors also propose a multi-phase variant [96] of their approach to perform multi class grouping.

The piecewise constant model of Chan-Vese [89] models the object foreground and background by the scalars c_1 and c_2. In a level set framework, the Chan-Vese energy functional $\mathcal{E}_d = \mathcal{E}_{CV}$ is written as follows:

$$\mathcal{E}_{CV}(\phi, c_1, c_2) = \int_\Omega |f(\mathbf{x}) - c_1|^2 H(\phi) d\mathbf{x} + \int_\Omega |f(\mathbf{x}) - c_2|^2 \left(1 - H(\phi)\right) d\mathbf{x} \tag{4.18}$$

Here $f(\mathbf{x})$ is the image defined over the domain Ω. The ϕ that locally minimizes (4.18) creates segments in a manner such that the foreground and background are best approximated by the intensity levels c_1 and c_2.

The (local) minima of (4.18) is obtained using alternate minimization. In the first step, the optimal values of the scalars c_1, c_2 are obtained as follows:

$$c_1^*, c_2^* = \underset{c_1,c_2}{\operatorname{argmin}} \mathcal{E}_d(\phi, c_1, c_2) \tag{4.19}$$

$$\phi^* = \underset{\phi}{\operatorname{argmin}} \mathcal{E}_d(\phi, c_1^*, c_2^*) \tag{4.20}$$

The solution for c_1^* andc_2^* is obtained in closed form as

$$c_1^* = \frac{\int f(\mathbf{x}) H_\epsilon(\phi) d\mathbf{x}}{\int H_\epsilon(\phi) d\mathbf{x}},\ c_2^* = \frac{\int f(\mathbf{x}) \left(1 - H_\epsilon(\phi)\right) d\mathbf{x}}{\int \left(1 - H_\epsilon(\phi)\right) d\mathbf{x}} \tag{4.21}$$

With these updated values, the local optima for the embedding function ϕ is computed by iteratively solving the following gradient flow equation.

$$\phi_t = \left[-(f(\mathbf{x}) - c_1^*)^2 + (f(\mathbf{x}) - c_2^*)^2\right] \delta_\epsilon(\phi) \tag{4.22}$$

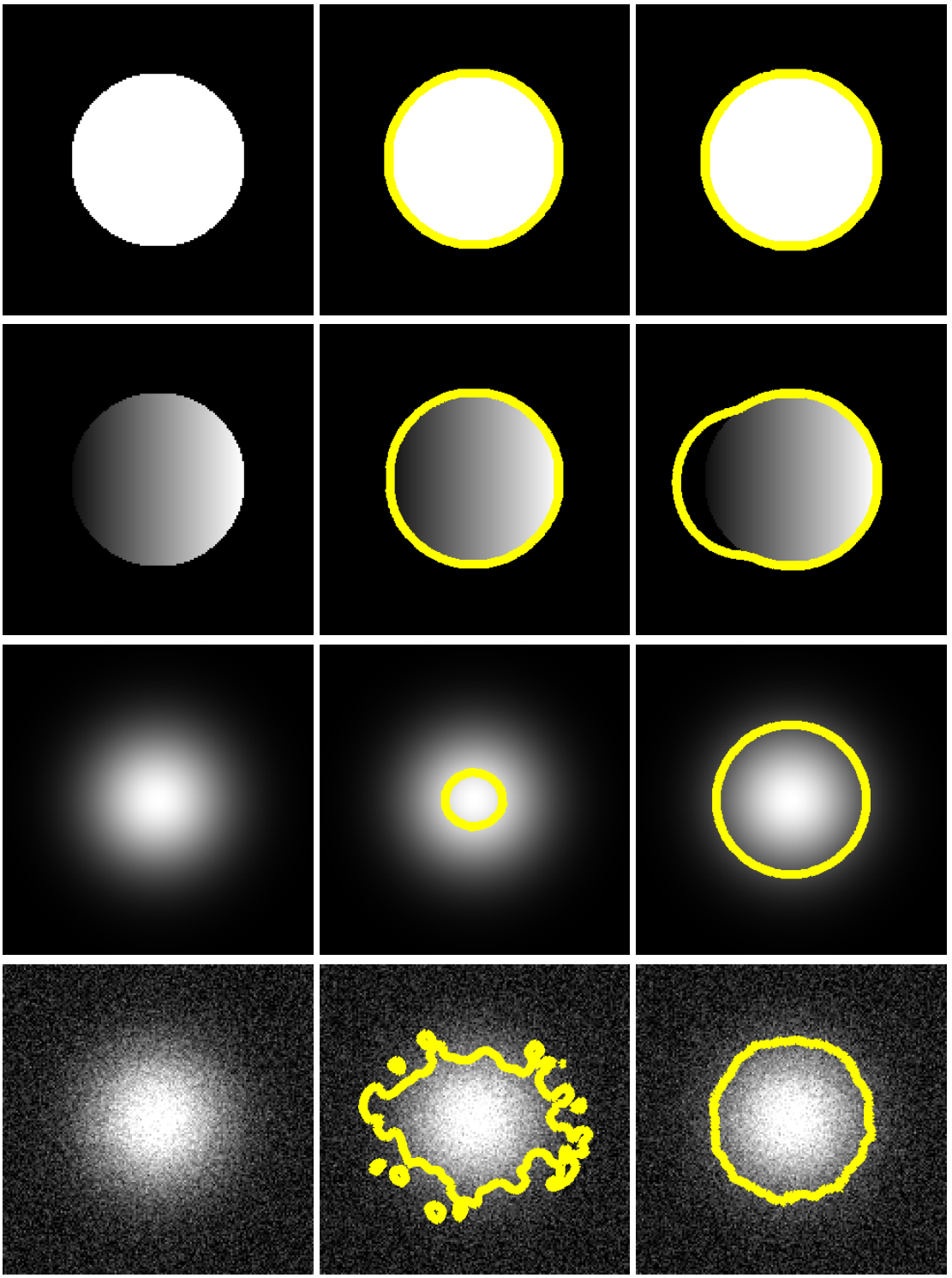
In addition to (4.22), a regularizer term as in (4.13) is also added to obtain smooth boundaries. This model is particularly useful when the object edges are not prominent or the edges are blurred. Also, region based methodologies exhibit significant performance benefits for noisy images where the accuracy of the edge map is sacrificed due to noise.

4.5 Edge based vs. region based models

4.5.1 Synthetic examples

Fig. 4.3 compares the performance of Geodesic Active Contour [63] algorithm which is an edge based framework, versus the region based segmentation technique due to Chan and Vese [89]. To demonstrate the properties of each algorithm, we simulate four images. The first image is that of a disc (Fig. 4.3(a), row 1) with homogeneous intensity level. This image has uniform illumination and the edges are well defined. Therefore, such an image is a good candidate for segmentation with either edge based or region based approaches. It is observed that both the algorithms perform almost identically in this case.

The second image (Fig. 4.3(a), row 2) is that of a disc with non uniform intensity. However, the object edges are still relatively well defined. As a result, almost perfect segmentation is obtained using the GAC model. However, Chan and Vese's algorithm is incapable of handling intensity inhomogeneity and fails to produce proper segmentation (Fig. 4.3(c), row 2).



(a) simulated image (b) GAC [68] (c) Chan-Vese [89]

FIGURE 4.3: (a)Simulated images. (b)Segmentation performance of (GAC [68]) and (c) Results via (Chan-Vese [89]). The final result is shown by the yellow contour.

In the third example (Fig. 4.3(a), row 3), the edges of the disc are blurred by smoothing it with a Gaussian filter. As expected, the edge based algorithm fails to identify the proper object boundary, whereas the region based technique performs significantly better.

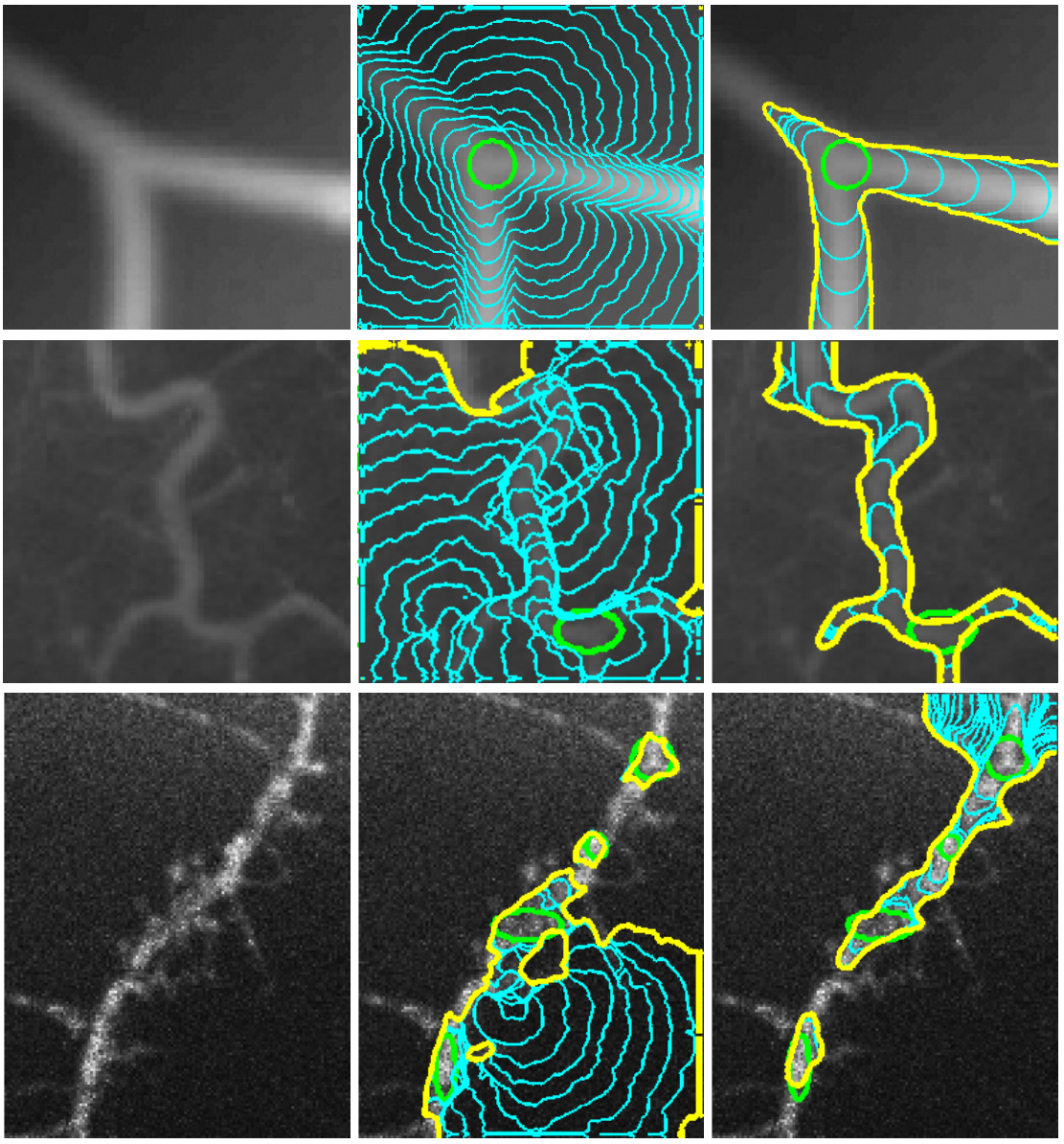
Finally, the fourth image (Fig. 4.3(a), row 4) is simulated by adding zero mean Gaussian noise to the blurry disc image. We observe that region based segmentation is particularly robust against additive noise and the segmentation performance is not degraded significantly. On the other hand, GAC model faces difficulties both due to blurred edges and due to noise.

4.5.2 Real examples

In Fig. 4.4, a comparative study of segmentation performance of geodesic active contour model and Chan-Vese's technique is presented on a set of real images consisting of vascular structures. Fluorescence microscopy images are often plagued by noise, inhomogeneous intensity and low contrast which leads to weak and fuzzy edges. As a result, we find that neither [63] nor [89] is perfectly suitable for segmentation under such conditions. Segmentation error is caused either due to the contour leakage phenomenon (Fig. 4.4(b)) or under segmentation (Fig. 4.4(c)). In the following chapter, we attempt to design a generic segmentation algorithm which is tolerant to the commonly occurring artifacts in fluorescence microscopy.

4.6 Discussion

In this chapter we have provided a broad overview of geometric active contours. Geometric contours are topology adaptive which makes them particularly attractive choice for segmentation. The segmentation model is either described in terms of explicit curve motion equation (4.1) and then implemented by implicit representation of the contour as the zero level curve of an embedding function (4.7). Another popular way to describe



(a) original image (b) GAC (c) Chan Vese

FIGURE 4.4: Cases of improper segmentation using geometric contours. (a) Five images including simulated and real examples with varying illumination and weak edges. (b) Shows segmentation using geodesic active contour and (c) Shows segmentation due to Chan-Vese's method. The initial contour is plotted in green and the final contour in yellow. Intermediate steps of curve evolution are shown in cyan. This figure is best viewed in color.

the segmentation process is via variational formulation which involves minimization of a suitable energy functional using variational calculus. The gradient flow equation (4.11) is derived using gradient descent technique for computing the local minima of the functional. In either case, segmentation is performed by iteratively propagating the zero level set of the embedding function on a discrete grid using numerical techniques.

Both formulations are widely used in the community and they enjoy their own set of benefits. While curvature flow equations are often easy to conceptualize (e.g. equavalent to physical curve motion models such as wave propagation), variational models have gained popularity because of the flexibility to add further constraints in the solution in terms of shape prior. We also showed that in many cases it is possible to draw an analogy between the curve flow equations derived from the two techniques and in a majority of cases the results are comparable.

We then presented a comparison between edge based and region based methods for segmentation. It is shown that edge based models are more suitable for high SNR images with prominent edges. Furthermore, such models are less susceptible to error if the object gray value is non homogeneous. Region based methods, on the other hand, are more robust to noise and do not depend particularly on the edge strength of the signal. However, it was demonstrated that the performance can degrade significantly in presence of intensity inhomogeneity.

In the next chapter we focus our attention on segmenting vascular structures from 2D fluorescence microscopy images. As we have discussed briefly, fluorescence microscopy imagery are charecterized by poor contrast, fuzzy boundaries, and varying intensity levels of the object. This restricts the performance of the popular traditional geometric active contour models. This motivates us to develop a geometric segmentation model which is capable of handling noise and contrast fluctuations in the images.

Chapter 5

Region based segmentation in presence of inhomogeneity

In the previous chapter we discussed the potential benefits of using geometric active contours for segmentation problems. Geometric snakes are capable of adapting to the topology of the objects and their ability to elastically deform and delineate object boundaries with sub-pixel accuracy make these methods attractive choice for several biomedical image analysis applications.

We have also illustrated how region based models are more suitable for segmenting noisy images with weak edges. One popular region based algorithm is due to Chan and Vese [89] where the authors model the image as a set of flat zones or regions with constant gray value. The authors also propose a multi-phase variant [96] of their approach to perform multi-class grouping.

The constant illumination assumption is challenged in applications where the signal intensity is inhomogeneous. This is encountered frequently in many medical and biological imaging applications like magnetic resonance (MR) imaging, ultrasound, X-ray, confocal and electron microscopy, etc. While edge based techniques are better suited for non

uniformly illuminated images, low SNR and weak edges of biological structures limit their general applicability.

5.1 Application to 2D neuron tracing

Before dealing with the 3D confocal images, which require more sophisticated processing, we first propose a segmentation algorithm which would work on relatively simpler 2D images. A 2D image is obtained from its 3D counterpart by taking a mean intensity projection along its vertical axis. Although the depth information is lost when the image dimension is reduced by projecting it to a 2D space, there are several interesting issues which demand our attention even after this simplification. First, even after conversion 2D, we still retain substantial information about a neuron's morphology and this is why there exists a number of popular tracers which have been developed specifically for 2D processing [34, 97]. Also, certain categories of neurons (e.g. the cells in the sub cuticle layer of the Drosophila) exhibit flat topology, and for such applications 2D processing is more relevant. Second, with the reduced dimension, one may perform computation at a much faster rate than in 3D, thereby making 2D analysis an attractive choice for an initial, global assessment of the neurites. Finally, the 2D conversion introduces further challenges for image processing, including introduction of intensity inhomogeneity, which occurs due to signal attenuation by the tissues at greater depths. It is a challenge in itself to investigate the applicability of state of the art segmentation algorithms on these datasets and understand the special processing needs for further robust analysis.

5.2 Background and motivation

In this chapter we introduce an edge oblivious segmentation approach *Legendre Level Set* (L2S), which is robust to smooth variations in region intensity levels. State of the

art techniques that tackle inhomogeneity typically require some form of local processing. However, while a global method like Chan-Vese's is insufficient in handling large scale intensity variations, a strictly local approach may lead to undesired segmentation artifacts, especially in presence of noise. We aim to eradicate these issues by proposing a generalized solution for region based segmentation in presence of significant intensity variation and additive noise.

Chan and Vese's region based technique is mathematically summarized in (4.18). The locally optimum level set embedding function ϕ^* that minimizes (4.18) partitions the image in two regions such that the foreground and background are best approximated by the scalars c_1 and c_2, which are computed using alternate minimization.

As mentioned previously, this model is incapable of handling spatially varying illumination. A solution was proposed in [96], where the authors replaced the scalars c_1, c_2 with smooth functions $c_1(\mathbf{x})$ and $c_2(\mathbf{x})$ in (5.1).

$$\begin{aligned}\mathcal{E}_{CV}(\phi, c_1(\mathbf{x}), c_2(\mathbf{x})) &= \int_\Omega |f(\mathbf{x}) - c_1(\mathbf{x})|^2 H_\epsilon(\phi) d\mathbf{x} + \int_\Omega |f(\mathbf{x}) - c_2(\mathbf{x})|^2 H_\epsilon(\phi) d\mathbf{x} \\ &\quad + s_1 \int_\Omega |\nabla c_1(\mathbf{x})| d\mathbf{x} + s_2 \int_\Omega |\nabla c_2(\mathbf{x})| d\mathbf{x} \end{aligned} \tag{5.1}$$

Smoothness of the functions are established by regulating their total variation, controlled by the parameters s_1, s_2. As before, alternating minimization is used to solve (5.1). However, unlike (4.18), the polynomials are computed numerically, by deriving the EL equations and using gradient descent for local minimization.

$$\begin{aligned}\frac{\partial c_1(\mathbf{x})}{\partial t} &= 2(f(\mathbf{x}) - c_1(\mathbf{x}))H_\epsilon(\phi) + s_1 \mathrm{div}\left(\frac{\nabla c_1(\mathbf{x})}{|\nabla c_1(\mathbf{x})|}\right) \\ \frac{\partial c_2(\mathbf{x})}{\partial t} &= 2(f(\mathbf{x}) - c_2(\mathbf{x}))H_\epsilon(\phi) + s_2 \mathrm{div}\left(\frac{\nabla c_2(\mathbf{x})}{|\nabla c_2(\mathbf{x})|}\right) \\ \frac{\partial \phi}{\partial t} &= \left[-(f(\mathbf{x}) - c_1^*(\mathbf{x}))^2 + (f(\mathbf{x}) - c_2^*(\mathbf{x}))^2\right] \delta_\epsilon(\phi)\end{aligned} \tag{5.2}$$

While the solution is attractive, this piecewise smooth model is computationally expensive. This is because, in order to calculate the locally optimum level set function ϕ^*, one needs to iteratively calculate the functions $c_1^*(\mathbf{x})$ and $c_2^*(\mathbf{x})$ by numerically solving (5.2) thereby making computation significantly expensive.

Recently, Li *et al.* [98] introduced a region scalable model to localize the energy functional. The region localization is controlled by the scale of a Gaussian kernel, which is manually tuned for optimal performance. Efforts have been made to incorporate the region statistics for segmentation [93, 99]. These methods are robust to initialization and relatively less sensitive to noise. However, Lankton *et al.* [91] demonstrated that global statistics may not be the best resort for segmenting inhomogeneous objects. Instead, the authors generalize the local region based methods, by proposing a generic energy functional capable of performing segmentation using different region based criteria. They also show that by judiciously tuning the region localizing mask size, one can achieve high quality segmentation, even in presence of noise and inhomogeneity. However, one downside of their approach is that it requires additional local computation, thus increasing the risk of being stuck within local minima.

Feng *et al.* [100] proposed a method for tomographic reconstruction by using a low order parametric model to represent object texture. However, the algorithm is tailored for tomographic reconstruction and is difficult to generalize. Recently a method was proposed to model the foreground and background by a linear function [101]. This approach is an improvement over the model of Chan-Vese, but does not accommodate nonlinear illumination change.

From the above discussion we observe that a majority of these approaches rely on local information only. While localizing the segmentation energy is essential in dealing with inhomegeneity, a generic global framework is also necessary to avoid the local minima problem. We propose to model the foreground and background illumination by

a set of Legendre basis functions [29]. This model allows the region intensities to be represented in a lower dimensional subspace, thereby permitting smooth approximation. Low dimensional signal representation has been used in a slightly different context in the literature, primarily to accommodate shape priors for segmentation [102, 103]. However, although shape based information assists segmentation, such techniques require an atlas of pre-registered objects, which may be unavailable for general purpose segmentation. We further show that the proposed model *Legendre Level Set* (or L2S) is computationally simple, since we achieve a stable, closed form solution at each iteration, allowing faster processing.

5.3 2D segmentation using L2S

The traditional Chan-Vese functional (4.18) can be reformulated and generalized by replacing the scalars c_1 and c_2 by smooth functions $c_1^m(\mathbf{x})$ and $c_2^m(\mathbf{x})$. These functions are used to model the intensity in the two regions separated by the zero level set curves of ϕ. The energy functional corresponding to the L2S data term is expressed as follows:

$$\mathcal{E}_{L2S} = \int_\Omega |f(\mathbf{x}) - c_1^m(\mathbf{x})|^2 H(\phi) d\mathbf{x} + \int_\Omega |f(\mathbf{x}) - c_2^m(\mathbf{x})|^2 \left(1 - H(\phi)\right) d\mathbf{x} \tag{5.3}$$

The essence of our approach is embedded in computing these functions. By allowing the regions to be modeled by flexible (but smooth) functions, we introduce the local information required to tackle the heterogeneous illumination. This is a notable feature of our algorithm. Unlike (5.1), where smoothness is obtained by minimizing the total variation term, we only allow inherently smooth polynomials to approximate the region intensities. As we will show in the next few sections, this formulation allows a suitable framework to model the heterogeneity, without significantly sacrificing on computational time.

To preserve the smoothness and flexibility of the functions, we represent them as a linear combination of a few Legendre basis functions as shown below:

$$c_1^m(\mathbf{x}) = \sum_{k=0}^{m} \alpha_k \mathcal{P}_k(\mathbf{x}) \tag{5.4}$$

$$c_2^m(\mathbf{x}) = \sum_{k=0}^{m} \beta_k \mathcal{P}_k(\mathbf{x}) \tag{5.5}$$

Here $\mathcal{P}_k$ is a multidimensional Legendre polynomial, which can be written as the outer product of the one dimensional counterparts. The 2-D polynomial is computed as $\mathcal{P}_k(x, y) = p_k(x)p_k(y)$, $\mathbf{x} = (x, y) \in \Omega \subset [-1, 1]^2$. p_k is a one dimensional Legendre polynomial of degree k defined as

$$p_k(x) = \frac{1}{2^k} \sum_{i=0}^{k} \binom{k}{i} (x-1)^{k-i}(x+1)^i \tag{5.6}$$

The highest degree of the 1D bases is denoted by m. Hence, for the 2D case, we would represent the regions by a linear combination of a set of $(m+1)^2$ 2D Legendre basis functions.

5.3.1 Optimization of the energy functional

Let us denote $\mathbb{P}(\mathbf{x}) = (\mathcal{P}_0(\mathbf{x}), \ldots, \mathcal{P}_{N-1}(\mathbf{x}))^T$ as the vector of Legendre polynomials. $\mathbf{a} = (\alpha_0, \ldots, \alpha_{N-1})^T$ and $\mathbf{b} = (\beta_0, \ldots, \beta_{N-1})^T$ are the coefficient vectors for the two regions. $N = (m+1)^2$ is the total number of basis functions. We can now rewrite the modified version of (4.18) in matrix form as

$$\begin{aligned} \mathcal{E}(\phi, \mathbf{a}, \mathbf{b}) = & \int_\Omega \left[|f(\mathbf{x}) - \mathbf{a}^T\mathbb{P}(\mathbf{x})|^2 m_1(\mathbf{x}) + |f(\mathbf{x}) - \mathbf{b}^T\mathbb{P}(\mathbf{x})|^2 m_2(\mathbf{x}) \right] d\mathbf{x} \\ & + \lambda_1 ||\mathbf{a}||_2^2 + \lambda_2 ||\mathbf{b}||_2^2 + \nu \int_\Omega |\nabla H_\epsilon(\phi)| d\mathbf{x} \end{aligned} \tag{5.7}$$

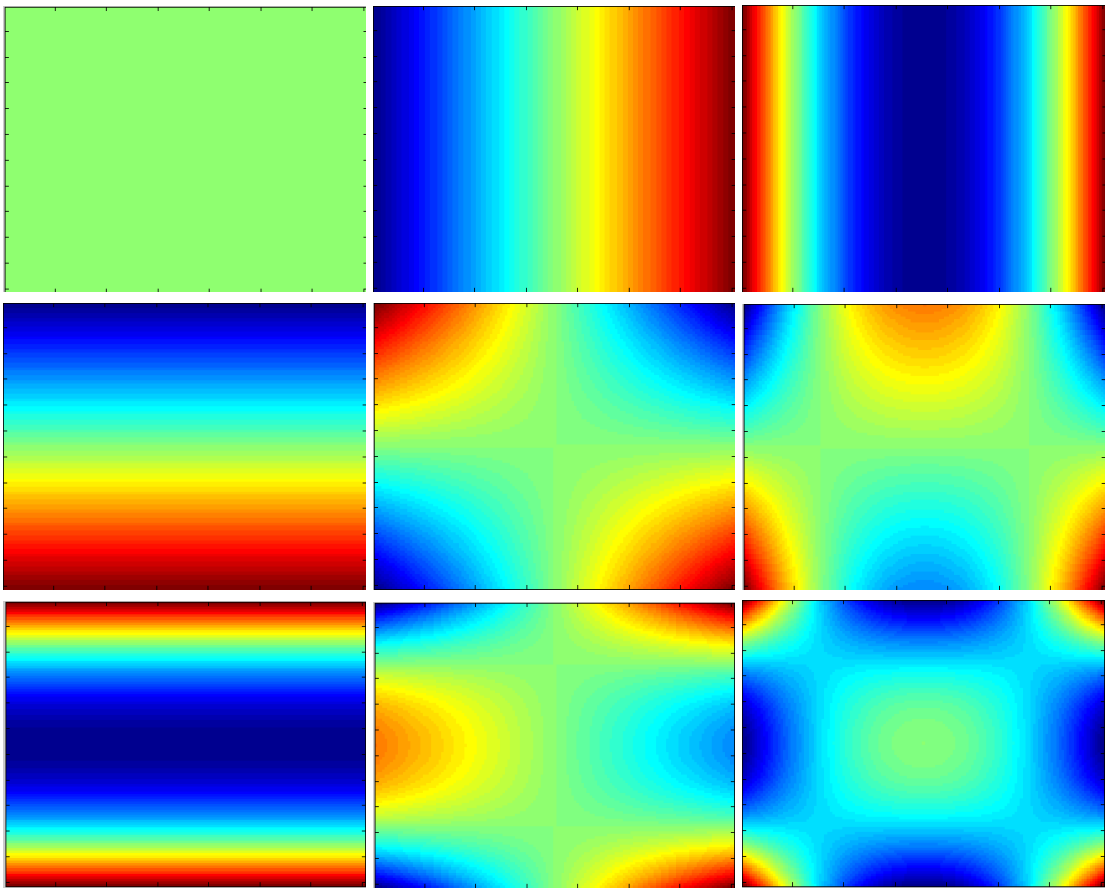


FIGURE 5.1: The set of nine 2D Legendre basis functions. The 2D polynomials are computed from 1D functions of degree 2.

In (5.7) the last term introduces smoothness in the zero level curve, which is regulated by the parameter ν. Let us also denote $m_1(\mathbf{x}) = H_\epsilon(\phi)$ and $m_2(\mathbf{x}) = 1 - m_1(\mathbf{x})$.

The non negative regularizing parameters λ_1, λ_2 can be selected using cross validation techniques to avoid over-fitting. The energy functional (5.7) is optimized using alternating minimization. In the first step, to find the optimal coefficients, we take the partial derivative of (5.7) with respect to $\mathbf{a}$ and $\mathbf{b}$ respectively and setting the result to zero. A closed form solution $\hat{\mathbf{a}}$ and $\hat{\mathbf{b}}$ is obtained as

$$\frac{\partial \mathcal{E}(\phi, \mathbf{a}, \mathbf{b})}{\partial \mathbf{a}} = 0 \Rightarrow \hat{\mathbf{a}} = \left[K + \lambda_1 \mathbb{I}\right]^{-1} \mathbf{p} \tag{5.8}$$

$$\frac{\partial \mathcal{E}(\phi, \mathbf{a}, \mathbf{b})}{\partial \mathbf{b}} = 0 \Rightarrow \hat{\mathbf{b}} = \left[L + \lambda_2 \mathbb{I}\right]^{-1} \mathbf{q} \tag{5.9}$$

[.] denotes a matrix. Here $[K]$ and $[L]$ are Gramian matrices [104] of dimension $N \times N$, whose $(i,j)^{th}$ entry are obtained as follows:

$$[K]_{i,j} = \left\langle \sqrt{m_1(\mathbf{x})}\mathcal{P}_i(\mathbf{x}), \sqrt{m_1(\mathbf{x})}\mathcal{P}_j(\mathbf{x}) \right\rangle = \int_\Omega m_1(\mathbf{x})\mathcal{P}_i(\mathbf{x})\mathcal{P}_j(\mathbf{x})d\mathbf{x} \tag{5.10}$$

$$[L]_{i,j} = \left\langle \sqrt{m_2(\mathbf{x})}\mathcal{P}_i(\mathbf{x}), \sqrt{m_2(\mathbf{x})}\mathcal{P}_j(\mathbf{x}) \right\rangle = \int_\Omega m_2(\mathbf{x})\mathcal{P}_i(\mathbf{x})\mathcal{P}_j(\mathbf{x})d\mathbf{x} \tag{5.11}$$

Here $\langle , \rangle$ denotes the inner product operator for real valued functions, and $0 \leq i,j \leq N$. The individual elements p_j and q_j of the $N \times 1$ vectors $\mathbf{p} = (p_0, \ldots, p_N)^T$ and $\mathbf{q} = (q_0, \ldots, q_N)^T$ are obtained as $p_j = \int_\Omega \mathcal{P}_j(\mathbf{x})f(\mathbf{x})m_1(\mathbf{x})d\mathbf{x}$ and $q_j = \int_\Omega \mathcal{P}_j(\mathbf{x})f(\mathbf{x})m_2(\mathbf{x})d\mathbf{x}$. This can be written in a more compact form using vector notations as follows:

$$\mathbf{p} = \int_\Omega \mathbb{P}(\mathbf{x})f(\mathbf{x})m_1(\mathbf{x})d\mathbf{x} \tag{5.12}$$

$$\mathbf{q} = \int_\Omega \mathbb{P}(\mathbf{x})f(\mathbf{x})m_2(\mathbf{x})d\mathbf{x} \tag{5.13}$$

With the updated coefficient vectors, we can now locally minimize (5.7) with respect to ϕ borrowing techniques from variational calculus. The curve evolution is performed by numerically solving the following partial differential equation:

$$\frac{\partial \phi}{\partial t} = \left[-|f(\mathbf{x}) - \hat{\mathbf{a}}^T\mathbb{P}(\mathbf{x})|^2 + |f(\mathbf{x}) - \hat{\mathbf{b}}^T\mathbb{P}(\mathbf{x})|^2 + \nu\,\nabla \cdot \left(\frac{\nabla\phi}{|\nabla\phi|} \right) \right] \delta_\epsilon(\phi) \tag{5.14}$$

We solve (5.14) using gradient descent and initializing $\phi|_{t=0} = \phi_0$ and $\frac{\delta_\epsilon(\phi)}{|\nabla\phi|}\frac{\partial\phi}{\partial\hat{n}} = 0$ at the domain boundary. See Appendix C for derivation of (5.14).

5.3.2 Analysis of L2S

The surface approximate for foreground and background are obtained by computing $\hat{\mathbf{a}}^T\mathbb{P}(\mathbf{x})$ and $\hat{\mathbf{b}}^T\mathbb{P}(\mathbf{x})$. Since the coefficient vectors are available in closed form, it makes

our algorithm fast and effective. The amount of intensity variation is governed by the coefficient vectors which are computed automatically. However, computing the coefficient vectors require a matrix inversion step. Here we show that the matrices $[K]$ and $[L]$ are invertible when the heaviside function is suitably regularized.

Since $[K]$ is a Gramian matrix, it is full rank iff the polynomials $\sqrt{m_1(\mathbf{x})}\mathcal{P}_i(\mathbf{x})$, $(i = 1, \ldots, N)$ are linearly independent [104]. Since the polynomials $\mathcal{P}_i(\mathbf{x})$ are linearly independent themselves, it is easy to show that the linear independence holds if $0 < m_1(\mathbf{x}) < 1$. A similar argument holds for analyzing the invertibility of $[L]$. In [89], the authors propose a regularized version of the heaviside function which is given by (4.14) By this definition, the functions $m_1(\mathbf{x})$ and $m_2(\mathbf{x})$ are bounded in $[0, 1]$, which make the matrices invertible.

However, inverting the above mentioned matrices may still be prone to numerical error when $\sqrt{m_i(\mathbf{x})}$ is small. The regularizing constants λ_1 and λ_2 contribute to make these matrices well conditioned. Furthermore, the regularization terms are necessary to avoid over-fitting. In most situations, we find that only a few (typically 16) 2-D Legendre functions are sufficient to model the region intensity. However, image noise may lead to over-fitting of the polynomials to the image segments, which may disrupt segmentation as the propagating level set may settle at a local minima. The scalars λ_1, λ_2 produce a damping effect by constraining the $\mathbb{L}_2$ norm of the bases coefficients, thereby favoring interior regions approximated by smooth functions.

5.3.3 Parameter selection for L2S

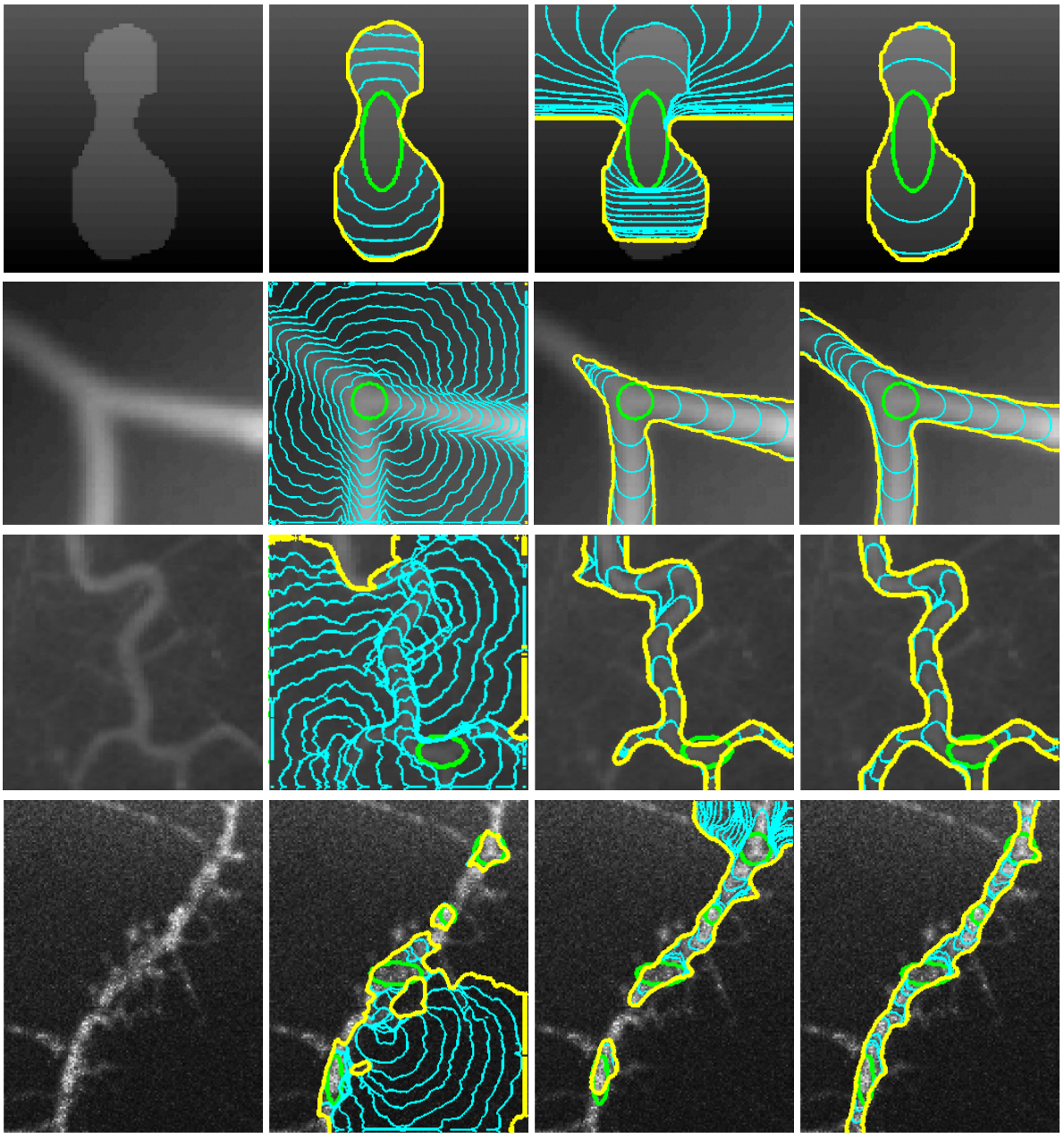
Our algorithm requires specification of a few parameters, namely the Legendre polynomial degree m and the regularizing constants λ_1 and λ_2 in (5.7). We experimentally verified that the intensity variation in the images can be adequately modeled by using 1-D Legendre polynomials of (highest) degree three. We found that the algorithm is relatively

robust to the selection of this value, but a higher degree polynomial typically requires inversion of a larger matrix, which makes computation significantly more expensive. To estimate the value of λ_1 and λ_2, we perform a *leave one out* cross validation on each of the four categories in our dataset. The cross validation is performed over the values of $\{0, 1, \ldots, 100\}$ in multiples of 2. For simplicity, we have chosen $\lambda_1 = \lambda_2$ for every experiment. The particular value which yields the highest average Dice coefficient for each dataset is chosen for experimentation.

Automated selection of the contour smoothness parameter ν in (5.7) is non-trivial. Typically, $0 < \nu < 1$, where a higher value produces smoother contour. As a rule of thumb, one may wish to set ν to a relatively higher value if the noise level in the image is high. For our experiments, we observe that the set of ultrasound images and the simulated noisy images require larger values of ν. For all these images, we select $\nu = 0.6$. For the less noisy images, ν is typically set in the range 0.05 to 0.2.

5.3.4 Comparison with GAC and Chan-Vese

In Chapter 4, we introduced the edge based geodesic active contour [63] model and region based technique due to Chan and Vese [89]. Fig. 5.2 shows the performance of L2S versus GAC and Chan-Vese's method. To maintain fairness of comparison, we have initialized the level set at the same positions for each methods (shown by the green contour). All the five images used for this demonstration are characterized by low contrast, weak edges and significant variation in region illumination levels. GAC and Chan-Vese's algorithm's performance is limited due to these artifacts. GAC is prone to error due to weak edges causing contour leakage, whereas the piecewise constant model due to Chan and Vese is unable to accomodate the intensity inhomogeneities. However, we observe that L2S exhibits significantly superior qualitative performance since it is (a) not dependent on



(a) original image (b) GAC (c) Chan Vese (d) L2S

FIGURE 5.2: (a) A 2D image. Segmentation via (b) Geodesic active contour, (c)Chan-Vese and (d)L2S. The initial contour is plotted in green and the final contour in yellow. Intermediate steps of curve evolution are shown in cyan. Best viewed in color.

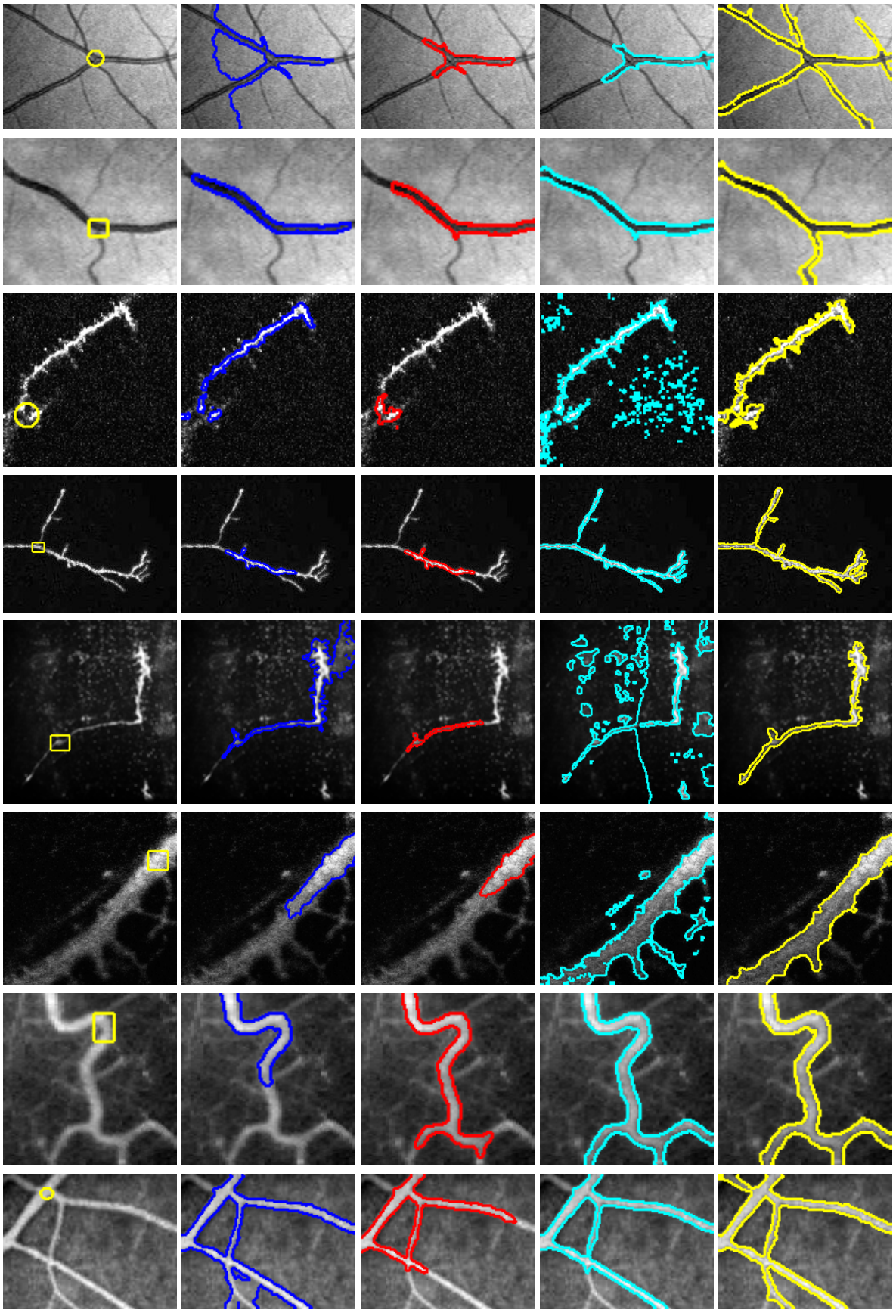
the edge information and (b) capable of handling discontinuities by using polynomial approximation for region intensities.

5.3.5 Comparison with other methods

To demonstrate the efficacy of the proposed method, we perform further experiments on a dataset of 32 images. The dataset consists of a set of synthetic images with added noise and simulated intensity inhomogeneity, a set of biomedical images consisting of blood vessels using magnetic resonance angiogram (MRA), neurons and dendritic spines imaged by confocal microscope and finally, a set of ultrasound images of human blood vessels.

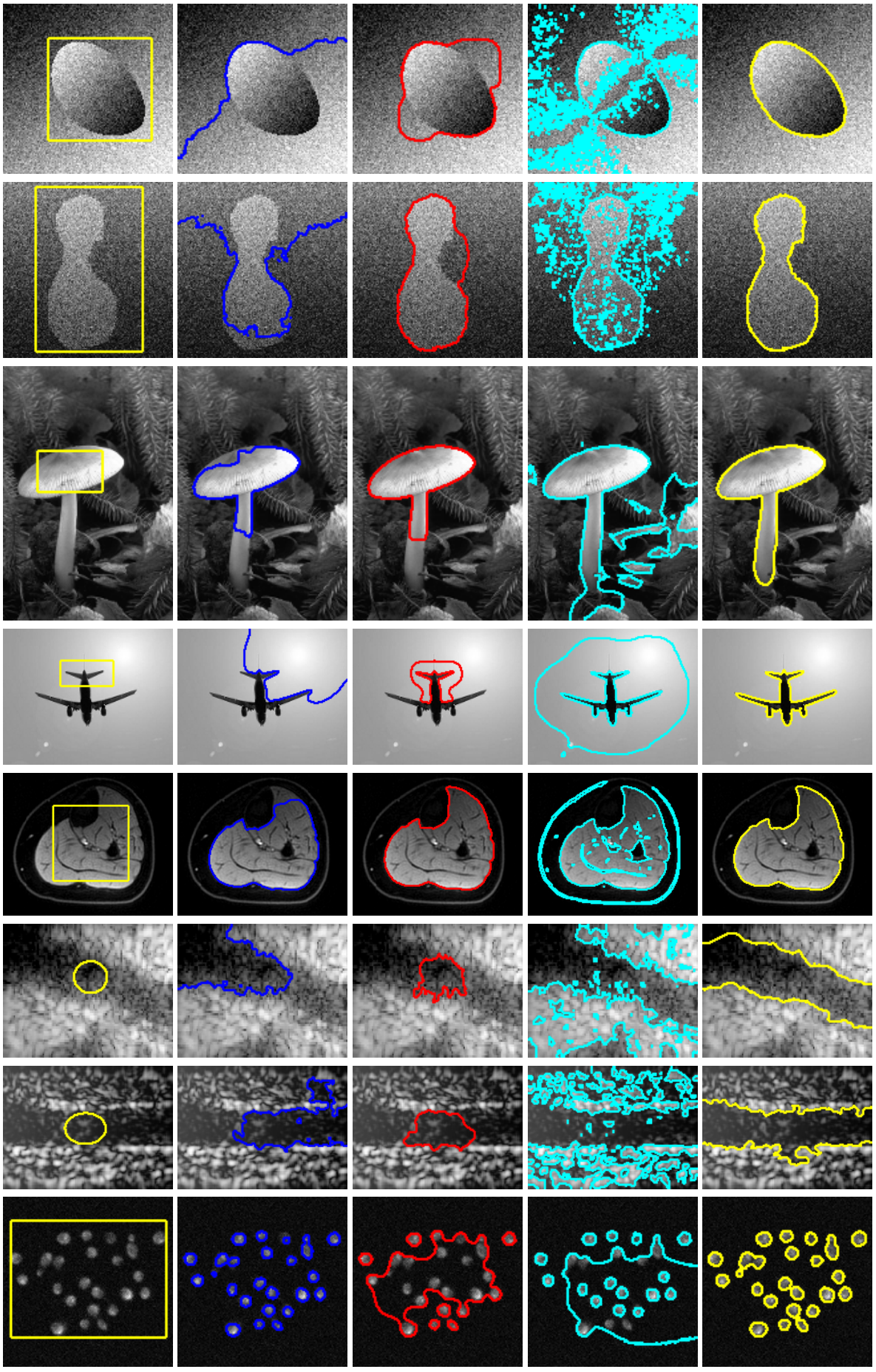
To evaluate the performance of L2S, we compare our approach with three popular and widely used region based segmentation algorithms viz. Chan-Vese [89], Lankton et. al. [91] and Li et. al. [98]. We use the freely available CREASEG [105] tool to evaluate the performance. We choose the above techniques for performance evaluation since all the above models (barring Chan-Vese) were developed to perform region based segmentation with varying object brightness.

To set up the comparative evaluation procedure, we first present the segmentation results on a biomedical image dataset containing vascular structures. This is shown in Fig. 5.3. Fig. 5.3(a) shows the original microscopy images with the initial contour shown in yellow, followed by segmentation results due to (b) Chan-Vese in blue, (c) Lankton *et al.* in red, (d) Li *et al.* in cyan and finally (e) L2S (yellow). The images contains vascular structures and are characterized by noise, non-object clutter and inhomogeneous intensity. To make a fair evaluation, the images were not preprocessed for contrast improvement or noise removal.



(a)2D image (b)Chan-Vese (c)Lankton *et al* [91] (d)Li *et al* [98] (e)L2S

FIGURE 5.3: Qualitative comparison of L2S for vascular images.



(a)2D image (b)Chan-Vese (c)Lankton *et al* [91] (d)Li *et al* [98] (e)L2S

FIGURE 5.4: Qualitative comparison of L2S for non vascular images.

We had mentioned earlier that one of our goals is to develop a segmentation procedure which is reasonably widely applicable. In this chapter we do not assume any structural prior for the objects to be segmented. Although prior information may achieve better results, the algorithm loses its general applicability. In the following chapter we will present a more problem specific solution to neuron segmentation. Since L2S is a general purpose segmentation algorithm, we also present qualitative results on non-vascular structures. However, almost all these images are characterized by inhomogeneous contrast, which is the major issue we try to address in this chapter.

Segmentation results on a few representative images are shown in Fig. 5.4. This set of non vessel images belong to different categories. The first two images are simulated to contain a varying contrast and additive noise. We also include MRI images of human leg muscles, natural images from the Berkeley segmentation database [106], noisy ultrasound images of human arteries and finally microscopy images of yeast cells. As before, L2S results are shown in yellow and qualitative comparison suggest robustness of the method.

5.3.6 Quantitative performance evaluation

The Dice coefficient [90] is used to quantify the results of segmentation. The Dice index $\mathcal{D} \in [0, 1]$ between two regions R_1 and R_2 is given by

$$\mathcal{D}(R_1, R_2) = 2\frac{|R_1 \cap R_2|}{|R_1| + |R_2|} \tag{5.15}$$

Here R_2 is a binary image that denotes ground truth segmentation, and R_1 is the result obtained experimentally. A Dice index of 1 indicates perfect segmentation. The quantitative performance is shown in Fig. 5.5. We observe that over this entire dataset, L2S yields an average Dice score of 0.9, compared to $0.62, 0.57$ and 0.7 for the methods described in [89], [91] and [98] respectively.

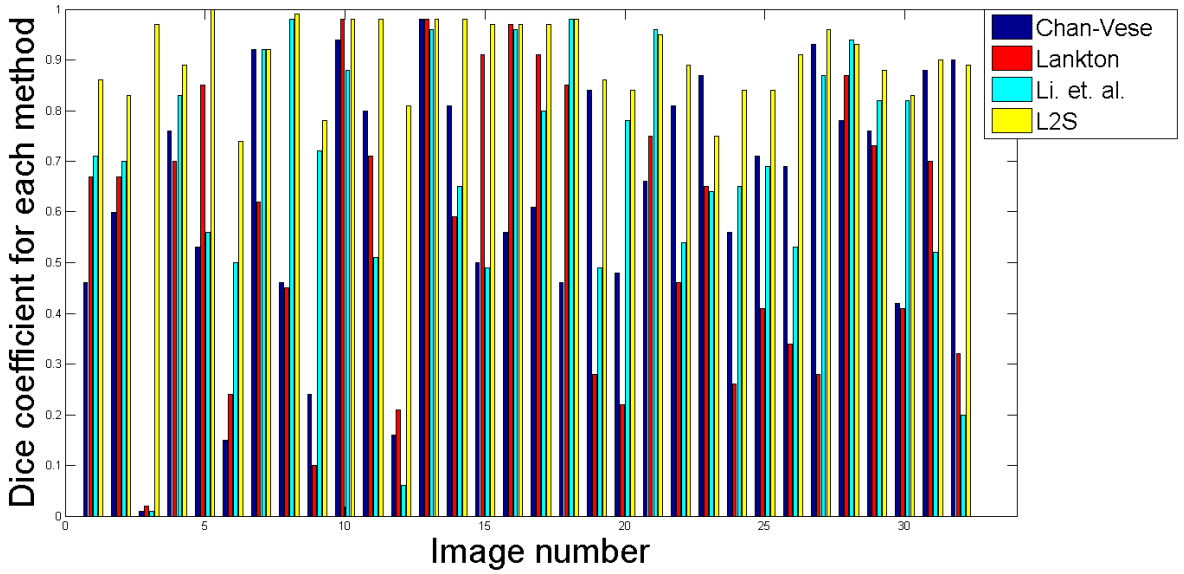


FIGURE 5.5: Dice index for the different algorithms are plotted in this bar chart. L2S results are shown by the yellow bar. Best viewed in color.

5.3.7 Computational comparison

Our algorithm is implemented in Matlab and all experiments are performed on an Intel Pentium processor with 16 GB memory. The convergence times (in seconds) for the four algorithms are presented in Fig. 5.6. Computationally, our method outperforms [91] and [98] on average. It may be noted that the apparent low convergence time of [89] often is a result of convergence at local minima, which do not necessarily correspond to correct

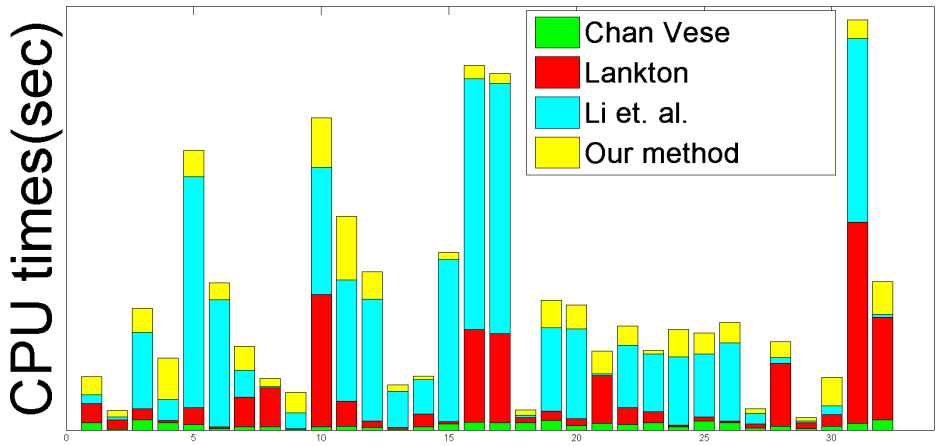


FIGURE 5.6: The CPU running times (sec) for the different algorithms are plotted in this bar chart. L2S results are shown by the yellow bar. Best viewed in color.

object boundaries.

5.4 Discussion

A novel framework for segmentation in presence of significant intra-region illumination variation is presented. Qualitative and quantitative results and comparison with the state of the art techniques suggest robustness of our approach. Here we have focused on bi-level segmentation, although extension to a multi-level framework appears straightforward. Also, our formulation allows easy incorporation of *a priory* shape information, which may enhance performance in select cases. Salient highlights of L2S are presented below:

- L2S uses geometric active contours for segmentation. Therefore, it can adapt to the topological variations in objects via automatic merging and splitting.
- L2S is robust against inhomogeneous intensity levels caused by non uniform signal attenuation or external bias fields, which occurs in many biological imaging applications.
- L2S is a region based method and its performance is robust against noise and weak edges.
- L2S is computationally efficient and numerically stable.

However, like most level set methods, L2S is somewhat biased towards contour initialization. Also, although Legendre polynomials for region intensity approximation provides an elegant solution, it is difficult to comment on the optimality of this choice of bases. Effectiveness of other polynomials such as splines or wavelets [107] needs further investigation. In select cases, it may also be possible to learn a compact set of bases for representation. To address this issue, we identified a scenario where a set of training examples of the object is available. We show that in such applications, the region approximating

polynomials may be learned efficiently, instead of pre specifying them. This segmentation algorithm, *Dictionary Learning Level Sets* (DL2S) [108], leverages the power of dictionary learning [109] to learn the region approximating polynomials. We provide the details of DL2S in Appendix A.

To develop a customized solution specifically for neuron tracing, we hypothesize that robust performance can be achieved by incorporating prior knowledge about the local shape of the neurites. While L2S is a general purpose, region based technique, in the following chapter we design a solution using level sets, that uses the local tubularity of the neurites to perform curve evolution.

Chapter 6

Neuron Segmentation with Tubularity Flow Field

In Chapter 3 and Chapter 5, we have discussed two segmentation algorithms. The first method, Tree2Tree-2 is a graph based tracer, which performs neuron tracing by identifying the correct connections between the neurite fragments after an initial segmentation. We remarked in Chapter 3, that Tree2Tree-2 encounters difficulties when the neurons exhibit complicated morphology, and it predicts false connections. This motivated us to use geometric active contours, so that the connectivity handling could be performed implicitly. This led to the region based segmentation algorithm L2S, which is primarily designed for 2D applications where the region intensities are inhomogeneous.

While L2S overcomes the false connectivity problem of Tree2Tree-2, it is essentially a generalized segmentation technique, and we hypothesize that adopting domain specific knowledge in the framework would make neuron segmentation more robust. This motivates our proposed solution, where level sets are used for segmentation, but the energy functional is designed specifically for segmenting tubular structures, both in 2D and 3D.

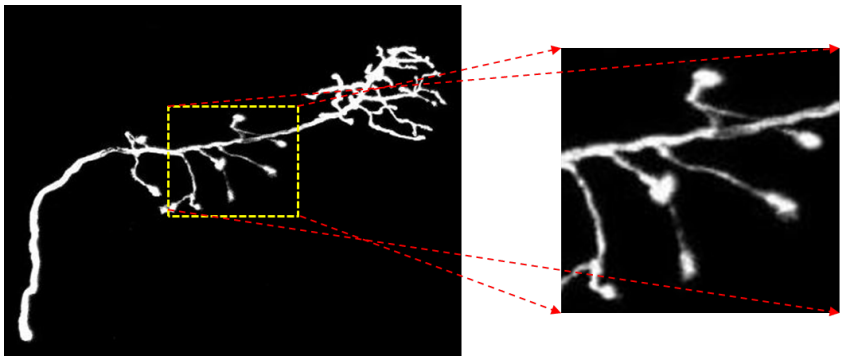


FIGURE 6.1: Maximum intensity projection of a neuron imaged by a confocal microscope. The image suffers from contrast non-uniformity, including gaps that lead to breaks in the segmented neurite structure. The effect is most pronounced in the region bounded by yellow dashed box, magnified here for improved viewing.

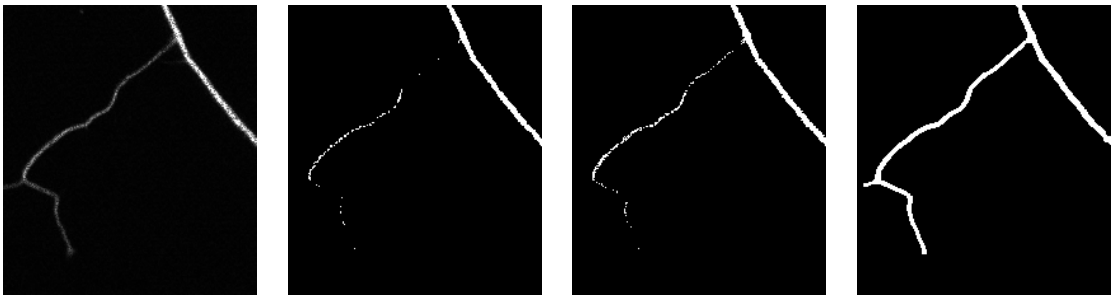
6.1 Introduction

As earlier, we restrict ourselves to reconstructing single neurons from confocal microscopy. A robust neuron segmentation scheme needs to address two primary issues.

- First, the technique should be suited to identify neuron structures from the noisy confocal images. This requires a specialized procedure for clutterand noise removal, while preserving the filamentous structures of the neurites.

- Second, it should be adept at handling the local structure discontinuities (see Fig. 6.1) resulting from imaging artifacts and pre processing errors. While Tree2Tree-2 used explicit schemes for joining such broken branches, we leverage the capabilities of geometric active contours to do the same.

We propose a solution to this segmentation problem using a variational framework driven by geometric active contours. The level set evolution is guided by minimizing an application specific energy functional. A tubularity flow field (TuFF) is computed by utilizing the local tubularity of the neurites which guides the segmentation procedure by encouraging curve evolution along the length (axis) and the thickness of the tubular neurites. A specialized local attraction force is also designed to accommodate the intensity variations in the images of neurite structures, thus presenting an unified framework to naturally link



(a) (b) (c) (d)

FIGURE 6.2: (a) A 2-D neuron subimage. (b) and (c) show segmentation results using Otsu's method and the Chan-Vese variational technique respectively. Fragmented segmentation output is observed in (b) and (c) due to the non-local behavior of the algorithms. (d) Segmentation using TuFF (the proposed method).

the fragmented structures. Our method does not rely on an initial set of seed-points for segmentation; it is automatic. Moreover, it does not require non-trivial post-segmentation analysis to link the disjoint segments. This is performed naturally by using the local attraction force in a level set paradigm. This enables us to connect disunited structures, even if the underlying signal intensity is significantly low. The problem formulation and the design process of the attraction force are discussed in the following sections.

6.2 Tubularity Flow Field for neuron segmentation

Let $f : \Omega \to \mathbb{R}$ be an image defined on the continuous domain $\Omega \subset \mathbb{R}^d$, where d is the dimension of the image. We propose a solution in a variational paradigm, where implicit motion of the zero level set of the embedding function ϕ is obtained by (locally) minimizing an energy functional $\mathcal{E}(\phi)$. For this problem of neuron segmentation, we need to design the energy functional such that it would encourage curve propagation in the filamentous regions of the image, while avoiding the non tubular structures. Also, the segmentation should allow sufficient local processing to avert fragmented segments in the solution, which may appear as a consequence of using global threshold selection schemes like that of Otsu [110] or methods assuming piecewise constant intensity models of [89] (see Fig. 6.2). We avoid this problem by introducing a local shape prior by way of a

specially designed tubularity flow vector field and a local attraction force to link nearby neuronal fragments.

6.2.1 Tubularity Flow Field (TuFF)

In Chapter 3, we defined the concept of tubularity flow field (TuFF). This vector field consists of the set of orthonormal vectors $\{\mathbf{e}_i(\mathbf{x})\}$. The vectors are ordered according to increasing magnitude of curvatures, which are given by the scalars $\{|\lambda_i|\}$. We showed in Chapter 3, that one popular procedure of obtaining these vector fields is via eigen analysis of the hessian matrix of the Gaussian filtered image. Frangi [4] suggested a multiscale procedure to distinguish vascular structures from background by using a multiscale vesselness function $N(\mathbf{x})$. One such choice of a vesselness function is given in (3.5).

The vesselness function $N(\mathbf{x})$ helps distinguishing filamentous structure from noise and clutter, where its value is close to zero. Clutter are present in most confocal microscopy images due to photon emission from non neuronal tissues and are often referred to as *structure noise*. These structure noise may be bright disc shaped non-neuronal segments in 3D images or blob shaped structures. In the following subsections, we show how TuFF can be incorporated in a level set framework to perform neuron segmentation.

6.2.2 Variational formulation with TuFF

Our method performs segmentation via minimization of the energy functional $\mathcal{E}(\phi)$. This energy functional can be mathematically written as:

$$\mathcal{E}(\phi) = \mathcal{E}_{reg}(\phi) + \mathcal{E}_{evolve}(\phi) + \mathcal{E}_{attr}(\phi) \tag{6.1}$$

$$\mathcal{E}_{reg}(\phi) = \nu_1 \int_{\Omega} |\nabla H(\phi)| d\mathbf{x} \tag{6.2}$$

$$\mathcal{E}_{evolve}(\phi) = -\int_{\Omega} \sum_{i=1}^{d} \alpha_i(\mathbf{x}) \langle \mathbf{e}_i(\mathbf{x}), \mathbf{n}(\mathbf{x}) \rangle^2 H(\phi) d\mathbf{x} \tag{6.3}$$

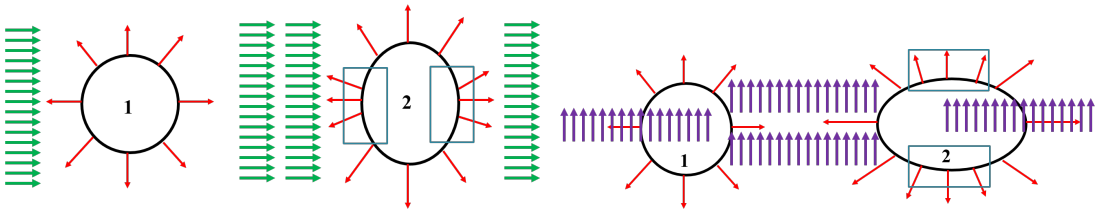
Here $\mathcal{E}_{reg}$ and $\mathcal{E}_{evolve}$ are the energy functionals corresponding to the smoothness of the curve and the curve evolution respectively. The functional $\mathcal{E}_{attr}$ contributes towards creating a local attraction energy. This attraction energy is to be designed in a manner such that minimizing it would result in a force field to join the local, disjoint neuron fragments. For our application, we do not define the attraction energy explicitly; instead, we compute the attraction force resultant from the energy. This is discussed in Sec. 6.3.

The vector $\mathbf{n}(\mathbf{x}) = -\dfrac{\nabla\phi(\mathbf{x})}{|\nabla\phi(\mathbf{x})|}$ denotes the outward unit normal vector to the level sets of ϕ. $\langle\cdot,\cdot\rangle$ is the Euclidean inner product operator. The positive scalar ν_1 in (6.2) contributes to the smoothness of the zero level curve. The weighing parameter α_i determines the contribution of the orthogonal and axial components of the TuFF in curve evolution. Choice of α_i is an important aspect which would be discussed shortly.

In practice, the ideal Dirac delta function $\delta(\phi)$ and the Heaviside function $H(\phi)$ are replaced by their regularized counterparts $\delta_\epsilon(\phi)$ and $H_\epsilon(\phi)$ respectively.

6.2.3 TuFF gradient flow equation

The essence of our technique lies in the design of curve evolution energy $\mathcal{E}_{evolve}$ in (6.3). In absence of the attraction force energy, the level curve evolution (which results from minimizing the energy term (6.3)) depends on the contribution of the axial and orthogonal components of the tubularity flow field. The design of the functional (6.3) is such that the axial vector field component $\mathbf{e}_1$ is responsible for propagating the curve to fill out the vessel thickness. Or in other words, the axial field promotes curve evolution in a direction perpendicular to itself. Identically, the orthogonal components $\mathbf{e}_2, \mathbf{e}_3$ encourage curve propagation in a direction perpendicular to themselves, i.e. along the axis of the neuron filaments. Let us discuss the effect of tubularity flow field on contour propagation in the following subsection. For simplicity, only a 2D case is discussed, but the extension to 3D follows similar arguments.



(a) (b)

FIGURE 6.3: Illustration of curve evolution due to (a) axial component $\mathbf{e}_1$ and orthogonal component (b) $\mathbf{e}_2$. Note how the contour should change to align the surface normals (shown in red arrow) with the vector fields (shown in green and purple arrows respectively) to minimize the evolution energy. The initial curve is marked as 1. The evolution forces create the new curves 2. Note how the curves assume elliptical shape to align the level set normals with the vector fields. The normal vectors are maximally aligned in the region enclosed by the rectangles.

6.2.3.1 Effect of the axial component of TuFF

Maximizing the total squared inner product $\int_{\Omega} \alpha_1(\mathbf{x})\langle \mathbf{e}_1(\mathbf{x}), \mathbf{n}(\mathbf{x})\rangle^2 H_\epsilon(\phi)$ (or minimizing its negative) with respect to the embedding function ϕ results in maximally aligning the outward normal vectors $\mathbf{n}(\mathbf{x})$ of the zero level sets of ϕ and its inner isocontours with the axial flow field $\mathbf{e}_1(\mathbf{x})$. This requires the level sets of ϕ to be re-aligned such that the normal vectors $\mathbf{n}(\mathbf{x})$ aligns itself with the axial field $\mathbf{e}_1(\mathbf{x})$. This results in curve evolution in a direction orthogonal to the vessel axis, causing elongation of the level curves along the vessel width.

6.2.3.2 Effect of the orthogonal component of TuFF

Using a similar argument, maximizing the second term corresponding to the orthogonal component in (6.3) performs alignment of the outward normal vectors with the vector field $\mathbf{e}_2(\mathbf{x})$, creating an elongation force which allows the level curves to propagate along the vessel axis. For an intuitive understanding of the above mentioned phenomenon, Fig. 6.3(a) and (b) is provided to graphically demonstrate how the curve evolution is affected by the axial and the normal components of TuFF.

6.2.3.3 Effect of the vector field weights

Ideally, the parameters $\alpha_i(\mathbf{x}), i = 1, \ldots, d$, should be chosen such that curve propagation is discouraged outside the tubular neurite segments, so as to avoid leakage into the background. i.e. for a voxel $\mathbf{y}$ with low vesselness score, we require $\alpha_i(\mathbf{y}) \approx 0$, for $i = 1, \ldots, d$. Moreover, since the neurites are elongated structures, it is desired that the contour evolution be more pronounced near the filament centerline than at the edges. This can be stated as

$$\frac{\alpha_j(\mathbf{x})}{\alpha_1(\mathbf{x})} \geq 1 \;\; (j = 2, \ldots, d) \quad \text{and} \;\; \alpha_1(\mathbf{x}), \ldots, \alpha_d(\mathbf{x}) > 0 \tag{6.4}$$

Respecting the above constraints, we propose the following functions for choosing the parameters.

$$\alpha_1(\mathbf{x}) = N(\mathbf{x}) \tag{6.5}$$

$$\alpha_j(\mathbf{x}) = N(\mathbf{x}) \left(a_0 + \exp\left(-\frac{|\nabla_\sigma f(\mathbf{x})|}{a_1} \right)^2 \right) \tag{6.6}$$

$\forall \mathbf{x} \in \Omega$ and $j = 2, \ldots, d$. $N(\mathbf{x})$ is the vesselness score which is obtained from (3.5).

6.2.3.4 Isotropic TuFF equation

Let us discuss the isotropic case, when $a_0 = 1$ and $a_1 \to \infty$. Since the unit normal vector $\mathbf{n}(\mathbf{x})$ lies in the vector space spanned by $\{\mathbf{e}_i(\mathbf{x})\}$, it can be written as $\mathbf{n}(\mathbf{x}) = \sum_{i=1}^{d} m_i \mathbf{e}_i(\mathbf{x})$. This reduces (6.3) to

$$\mathcal{E}_{evolve}(\phi) = -\int_\Omega N(\mathbf{x}) \sum_i \langle \mathbf{e}_i(\mathbf{x}), \sum_j m_j \mathbf{e}_j(\mathbf{x}) \rangle^2 H_\epsilon(\phi) d\mathbf{x} \tag{6.7}$$

Since the eigenvectors are orthonormal, $\langle \mathbf{e}_i, \mathbf{e}_j \rangle = 1$ if $\mathbf{e}_i, \mathbf{e}_j \neq \mathbf{0}$ and $i = j$, and 0 otherwise. Also, since $|\mathbf{n}(\mathbf{x})| = 1$, we have $\sum_i m_i^2 = 1$. Using this relation, we obtain

$\sum_i \langle \mathbf{e}_i(\mathbf{x}), \sum_j m_j \mathbf{e}_j(x)) \rangle^2 = 1$. This reduces the evolution equation to

$$\mathcal{E}_{evolve}(\phi) = -\int_{\Omega} N(\mathbf{x}) H_{\epsilon}(\phi) d\mathbf{x} \tag{6.8}$$

The energy functional in (6.8) when minimized performs segmentation via vesselness weighted isotropic region growing along the neuron segments. Leakage of the contour outside vessel boundaries is prohibited by the vessel indicator function $N(\mathbf{x})$ which provides evidence of tubularity by assuming higher value for the tubular objects than non tubular background.

With the discussion of the isotropic case, it is now easier to visualize the effect of the weights on curve evolution. From our previous discussion, we recall that α_1 and $\{\alpha_j, j \neq 1\}$ influence curve propagation along the vessel width and axial direction respectively. $|\nabla_\sigma f(\mathbf{x})|$ denotes the gradient magnitude of the image $f(\mathbf{x})$, which is filtered by a Gaussian kernel with variance σ^2. Since this term is high at the vessel boundaries and end points, the negative exponential term in (6.6) ensures higher response at regions near the vessel centerline. The tuning parameters $a_0 \geq 1$ and a_1 determine the relative influence of the axial curve motion to the motion along the vessel width. In other words, in an anisotropic setting, (6.6) suggests that the level curves evolve with higher curvature near the vessel medial axis than at the edges, which percolates to the isotropic case when $a_0 = 1$ and $a_1 \to \infty$.

Since the neurite filaments are predominantly thin, elongated structures, we observe that the isotropic case yields sufficiently appropriate segmentation results when the initialized zero level set encompasses the filament width. Nevertheless, the proposed framework in (6.1) is general, and is applicable to segmentation problems where vessel thickness is significant and the initialized zero level contour does not fill out the vessel width completely. This is in contrast to the approach in [60], where segmentation of thicker vessels needs separate treatment.

6.2.4 Minimization of the TuFF functional

The energy functional in (6.1) can be minimized using variational calculus techniques [87]. Taking the Gâteaux variation of $\mathcal{E}(\phi)$ with respect to ϕ, we obtain from (6.1)

$$\nabla_\phi \mathcal{E} = \nabla_\phi \mathcal{E}_{reg} + \nabla_\phi \mathcal{E}_{evolve} + \nabla_\phi \mathcal{E}_{attr} \tag{6.9}$$

ϕ can be iteratively updated using gradient descent technique, i.e. setting $\nabla_\phi \mathcal{E} = -\dfrac{\partial \phi}{\partial t}$ with t denoting the pseudo time parameter for the iterative scheme:

$$\frac{\partial \phi}{\partial t} = \mathcal{F}_{reg}(\mathbf{x}) + \mathcal{F}_{evolve}(\mathbf{x}) + \mathcal{F}_{attr}(\mathbf{x}) \tag{6.10}$$

$\mathcal{F}_{reg}$ and $\mathcal{F}_{evolve}$ are scalar force functions, which resulting from locally minimizing the regularizing energy and the evolution energy functionals. These forces are derived by solving the Euler-Lagrange equation for level set evolution in the following manner:

$$\mathcal{F}_{reg}(\mathbf{x}) = \nu_1 \,\mathrm{div}\,(\mathbf{n})\,\delta_\epsilon(\phi) \tag{6.11}$$

$$\begin{aligned}\mathcal{F}_{evolve}(\mathbf{x}) = {} & \delta_\epsilon\,(\phi) \sum_{j=1}^{d} \{\alpha_j(\mathbf{x})\beta_j^2\,(\mathbf{x})\} - \\ & 2\,\mathrm{div}\left(\sum_{j=1}^{d} \eta_j(\mathbf{x})\,(\mathbf{v_j}(\mathbf{x}) - \beta_j\,(\mathbf{x})\,\mathbf{n}(\mathbf{x}))\right)\end{aligned} \tag{6.12}$$

The coefficients β_j and η_j are defined as follows:

$$\beta_j(\mathbf{x}) = \langle \mathbf{v}_j(\mathbf{x}), \mathbf{n}\,(\mathbf{x})\rangle \tag{6.13}$$

$$\eta_j(\mathbf{x}) = \alpha_j(\mathbf{x})\beta_j(\mathbf{x})\frac{H_\epsilon(\phi)}{|\nabla\phi|} \tag{6.14}$$

The derivation details are shown in Appendix C.

6.3 Local attraction force field

The attraction force $\mathcal{F}_{attr}$ in (6.10) is introduced to accommodate the signal intensity variation (and signal loss) across the neurite branches, as shown in Fig. 6.1. Such signal attenuation introduces unwarranted discontinuities in the filamentous objects, resulting in disjoint fragments. Also, discontinuities may be present at the neurite junctions and noisy regions due to the nonlinear response of the vesselness function in (3.5). In such a scenario, the TuFF based evolution energy term in (6.3) is not adequate by itself to perform segmentation. This insufficiency motivates the inclusion of an attraction force component. Designing this attraction force requires analysis of the connected components at each time epoch of level set propagation. At a time t for evolution of the embedding function, the set of connected components $\mathcal{C}(t)$ are obtained as:

$$\mathcal{C}(t) = H\left(\phi(\mathbf{x}, t)\right) \tag{6.15}$$

$$\text{where } H(y) = \begin{cases} 1 & \text{for } y \geq 0 \\ 0 & y < 0 \end{cases}$$

The set of connected components $\mathcal{C}(t) = \{c_1, \ldots, c_p\}$ represents the binary segmentation at time t, which consists of $p \geq 1$ disjoint connected components. Note that this binarization does not require a sophisticated segmentation, since the binary components are obtained by extracting the interior of the zero level sets of the embedding function. Each disjoint component c_j is a potential candidate or a *parent* which has the capability of attracting the remaining *children* c_k, $k \neq j$,$(j, k = 1, \ldots, p)$. This is illustrated in Fig. 6.4(a)-(c), where the component c_1 acts as a parent component and c_2 and c_3 are the children.

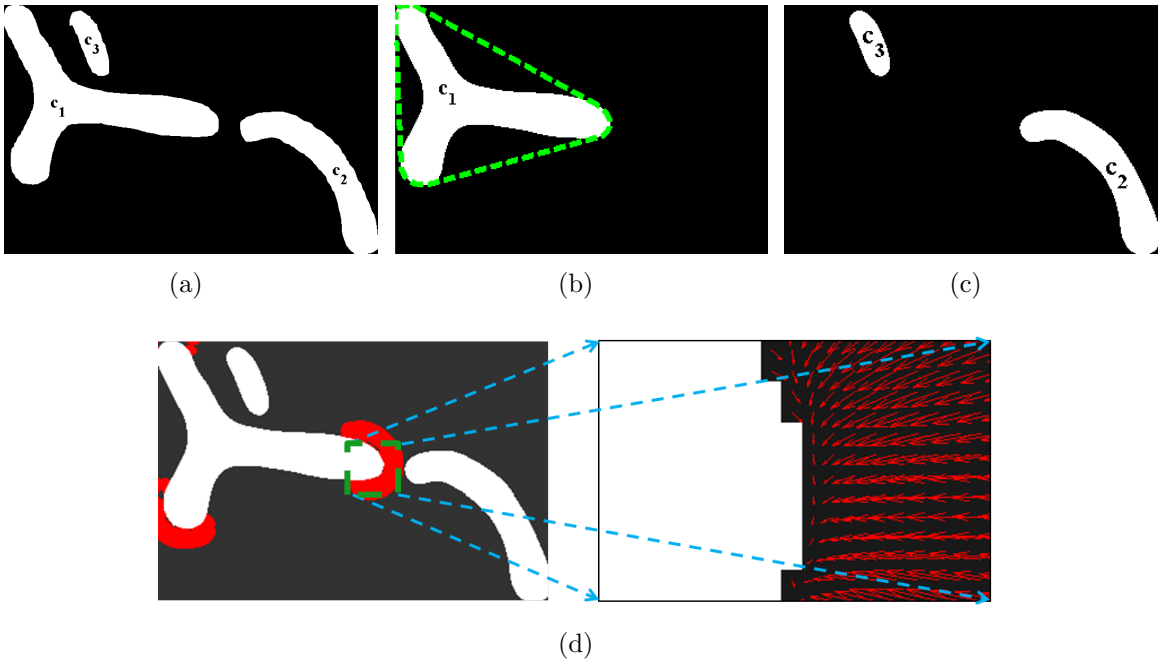


FIGURE 6.4: (a) Set of disjoint connected components $\{c_1, c_2, c_3\}$ at a particular step of iteration. (b) shows a parent component, the green dotted line marking its convex hull. The remaining children are shown in (c). (d) shows the attraction force obtained via (6.18) in red arrows, magnified for visual clarity.

6.3.1 Candidate points for attraction force field

The primary responsibility of the attraction force is to enable the propagating contour surface to attach itself to local disjoint fragments. However, not all points on the connected components are candidates for creating the attraction force. This is because in a majority of the prevalent discontinuities, at least one of the two disconnected portions are likely to be joined via boundary points which represent region of high curvature (see Fig. 6.5). If we denote the boundary of a component c_j by δc_j, to enable a parent to attract a child, we need to design an attraction field which is generated by a set of candidate points lying on the parent boundary. Therefore, for a parent component c_j, a point $\mathbf{y} \in \delta c_j$ belongs to the candidate set if $\mathbf{y}$ is a point the convex hull [111] $\mathcal{H}_j$ of c_j (Fig. 6.4(b)). Formally, the candidate point set $\mathcal{M}_j$ for the connected component c_j is defined as

$$\mathcal{M}_j = \{\mathbf{y} \in \delta c_j : \exists\, \mathbf{x}_j \in \mathcal{H}_j \;\text{ s.t. }\; \|\mathbf{y} - \mathbf{x}_j\|_2 \leq \Delta\} \tag{6.16}$$

Δ is a positive parameter that includes local boundary coordinates of the neighboring points on the convex hull.

6.3.2 Attraction force field design

The candidate set of points for a parent component is responsible for generating a force field capable of attracting the candidate children towards itself for potential merging. This needs to be designed such that the attraction field vectors point toward the region of interest, which is the parent candidate point set for this purpose. We show that an efficient solution may be obtained by using vector field convolution (VFC) to create the attraction force field.

VFC [70] is a technique primarily designed to create smooth external force field for parametric active contours. The specially designed vector field kernel (6.17) generates the desired external force when convolved with the object edge map, with the capability of attracting a contour to the region of interest.

$$\begin{aligned} \mathbf{K}(\mathbf{p}) &= -m(\mathbf{p})\tfrac{\mathbf{p}}{\|\mathbf{p}\|} \\ m(\mathbf{p}) &= \exp(-||\mathbf{p}||^2/\gamma^2) \end{aligned} \tag{6.17}$$

$\mathbf{p} = \mathbf{0}$ denotes the kernel center. The capture range of the vector field is controlled by the parameter γ.

The set of candidate points $\mathcal{M}_j$ for a parent c_j serves as the region of interest to which other components are likely to be attracted. Performing convolution of the candidate set with the kernel in (6.17) results in a vector field where the vectors are directed toward the parent, their magnitude attenuating gradually with distance from the candidate set. If $E_j(\mathbf{x})$ is a binary edge-map which assumes a value of 1 only at points in $\mathcal{M}_j$, we can

obtain the attraction force field Γ_j due to the parent m_j as

$$\Gamma_j(\mathbf{x}) = E_j(\mathbf{x}) * \mathbf{K}(\mathbf{x}), \quad \forall \mathbf{x} \in \Omega. \tag{6.18}$$

The nature of the attraction force field can be intuitively understood from Fig. 6.4. Fig. 6.4(a) shows three connected components and the representative parent c_1 enclosed by its convex hull (shown in (b)). Fig. 6.4(c) illustrates the attraction force field due to the parent as the red arrows which are oriented in the direction of the parent component. The capture range, which is specified by γ, is shown by the red region.

Adopting this policy for designing the attraction field enjoys a few benefits. First, with a specified capture range, we can impose a locality in the approach, by discouraging distant segments to be connected to the parent. As γ increases, effect of the attraction force field gradually diminishes as one moves further from the parent. Moreover, the candidate set is chosen such that only the convex portions of the parent boundary are capable of generating the force field. This ensures not all local structures are potential candidates for linking. For example, in Fig. 6.4 the component c_3 is not in the capture range of the force field of c_1, although it resides in the parent's local neighborhood. To summarize, the attraction force field is designed such that it may attract local connected components which are present in near vicinity of the parent's boundary convexity.

6.3.3 Attraction force

For a parent-child pair c_i and c_j, the parent attracts the child with a force $\mathcal{F}_{attr}^{(i,j)}$ given by

$$\mathcal{F}_{attr}^{(i,j)}(\mathbf{y}) = \mathcal{M}_i \langle \Gamma_i(\mathbf{y}), -\mathbf{n}(\mathbf{y}) \rangle \theta_j(\mathbf{y}) \tag{6.19}$$

The indicator function $\theta_j(\mathbf{y}) = 1$ if $\mathbf{y} \in \delta c_j$ and 0 otherwise. $\mathcal{M}_i$ is the normalized mass of the component c_i which is computed as the ratio of the number of pixels/voxels in c_i

to the total pixels/voxels in $\{c_1, \ldots, c_p\}$. The inner product term in (6.19) suggests that higher force of attraction is experienced by a point on a child's boundary if the outward normal at that point is oriented along the attraction field.

By introducing the factor $\mathcal{M}_i$, we equip heavier connected components with more attractive power. Assuming that the neurites occupy larger volume than the noisy background voxels, we clean the solution of the level set function by performing an area opening operation which eliminates small components with area less than a pre defined threshold [112]. This filtering operation avoids undesired objects to participate in the attraction force field computation. Now, for each parent-child pair in the filtered component space, we can compute the total attraction force $\mathcal{F}_{attr}$ in (6.10) as

$$\mathcal{F}_{attr}(\mathbf{y}) = \nu_2 \sum_{i=1}^{p} \sum_{j \neq i}^{p} \mathcal{F}_{attr}^{(i,j)}(\mathbf{y}), \quad \forall \mathbf{y} \in \Omega. \tag{6.20}$$

The positive scalar ν_2 determines the effect of the attraction force on curve evolution. A finite difference scheme is used to solve the PDE in (6.10) with initial value obtained using Otsu's global segmentation [110] and Neumann boundary condition.

6.4 Handling of discontinuities

Typically, one may encounter two major sources of structure discontinuity arising from initial segmentation. Fig. 6.5 shows three synthetic, disjoint components at an arbitrary stage of level set evolution. The type A discontinuity occurs when connectivity is absent between the end points or leaves of the centerline of the respective objects. Type A discontinuities dominate our application, and connectivity analysis of type A may be performed via Tree2Tree [3], by investigating the geometric orientation and Euclidean distance between the end points . However, end-point analysis algorithms like Tree2Tree are unable to process the type B discontinuities, where the link needs to be established

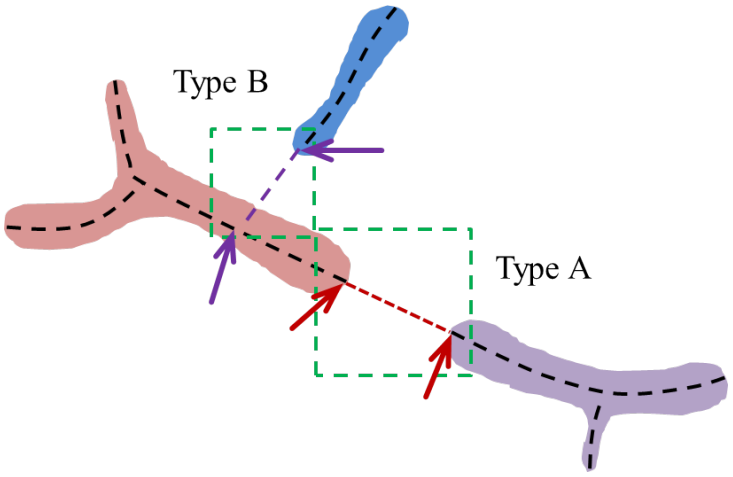
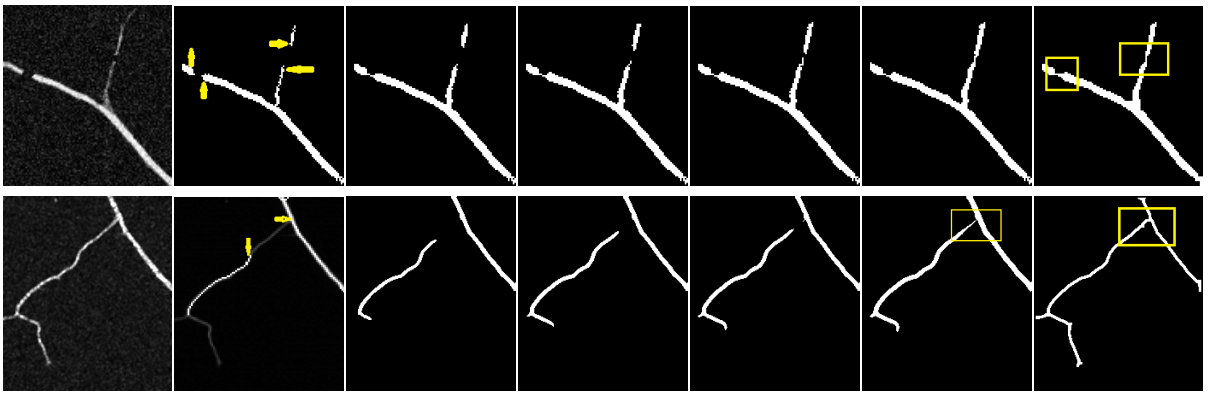


FIGURE 6.5: Two types of discontinuities between the disjoint components. The *Type A* discontinuity can be resolved by joining the end points of the center lines of the respective branches. *Type B* is more difficult, where discontinuity occurs between a branch end point and an intermediate point on the centerline of the other branch.

between the terminal node of one component with a non-terminal point on the other object. This problem is persistent in Tree2Tree-2 also, since the algorithm eventually uses the explicit connectivity determination step to link the neurite subcompartments. This is where the proposed level set framework wins over conventional component linking algorithms since level sets are proficient in handling topological changes of the evolving segmentation.

6.4.1 Type A discontinuities

Type A discontinuities are relatively simpler to analyze. If the neuron filament signal intensity is uniform, then the evolution force component of (6.10) sufficiently propagates the level sets until they are finally merged. However, when the signal drop is substantial, the attraction force term in (6.10) assists the parent and the child component to exert attractive forces on one another, thus propagating the curves till they merge. A demonstration is shown in the first row of Fig. 6.6. The initial segmentation using Otsu's method creates type A gaps, which are ultimately merged. We have intentionally eliminated a portion of the neuron's branch to demonstrate that our methodology works even in complete absence of signal.



(a) (b) (c) (d) (e) (f) (g)

FIGURE 6.6: (a) and (b) shows the original image and the initial global segmentation respectively for two cases demonstrating handling of Type A (top row) and Type B (bottom row) discontinuities. (c)-(f) shows segmentation at subsequent time intervals. (g) shows the final segmentation, where the structure gaps have been closed (the merged portions are enclosed in rectangles).

6.4.2 Type B discontinuities

Type B discontinuity involves two segments, for which connectivity needs to be established between one component's end point (or tip) with the other component's body. In presence of adequate signal intensity, TuFF drives the geometric contours toward the participating structure as per the filament orientation. However, when signal intensity drops, the attraction force takes over. An example is shown in the second row of Fig. 6.6(b), where the initial segmentation creates a type B gap. The situation is different from that of type A, where both the components may attract each other. In case of type B, only one component can assume a parent's role. Note that this is the extreme scenario, where the underlying signal strength is so feeble that it renders the evolution force term useless. However, assuming that the parent's mass is not negligible, this attraction force is strong enough to pull the local child connected component for potential merging. It should be noted that only those regions on the child's boundary whose outward normals are maximally aligned with the exerted force field are attracted toward the parent.

6.5 Curve evolution equation

Numerical implementation of (6.10) allows iterative computation of the level set function, which can be expressed as

$$\phi^{(k+1)} = \phi^{(k)} + \Delta t \mathcal{L}^{(k)} \tag{6.21}$$

The learning rate Δt is fixed to a small value (≈ 0.1) to allow stable computation. $\mathcal{L}^{(k)}$ denotes the discretized version of the right hand side of (6.10). $\phi^{(k)}$ is the level set function at iteration k. To initialize the active contour, we require the initialized curve to be inside the neurite structure. The initial level set function may be easily obtained via few mouse clicks to select a region inside the neuron structure. However, to avoid this human involvement, we perform a global thresholding of the scale space vesselness image (3.5) using Otsu's technique [110], followed by noisy binary segment removal using the area open filter [112]. The iterative procedure is halted when no significant change in the length of the zero level curve of ϕ is observed. At convergence, the neuron structure is extracted by selecting the largest binary component in the solution. A cubic spline is then fitted to each branch of the obtained centerline to obtain smooth tracing of neuron centerline.

6.6 Experimental results

To evaluate the performance of TuFF, we set up experiments for segmenting neurons from both 2D and 3D images. As we have mentioned earlier, 2D analysis often serves as the first step for neuron morphological studies. Plus, certain categories of neurons (such as the ones in the cuticle layer) exhibit a flat geometry, and in such cases the extra dimension does not add significant information about their structure. In this section, qualitative segmentation results (for both 2D and 3D images) will be presented first,

with an emphasis on some salient properties of TuFF such as segmentation in presence of severe signal attenuation and robustness against Type B connectivity errors. The quantitative results of neuron tracing will be then furnished, with a suitable similarity metric, calculated against manually obtained results.

6.6.1 Dataset for segmentation

We test the performance of TuFF segmentation algorithm on sets of 2-D and 3-D confocal microscopy images. The 2-D images are primarily used to demonstrate the efficacy of TuFF over component analysis algorithms like Tree2Tree [3]. The 3-D image data set consists of confocal microscopy images of the Drosophila larva, which are genetically tagged with green fluorescence protein (GFP). A majority of the neurons have been imaged in Dr. Barry Condron's laboratory at the University of Virginia, department of Biology. The images are captured using a laser scanning confocal microscope and has a horizontal pixel width of $0.14\mu m$ and vertical pixel width of $0.18\mu m$. These images are characterized by intense background clutter from non neuronal objects (such as the food particles, mildly fluorescing tissues etc.) and considerable contrast and intensity variation. This dataset will be referred to as *Condron dataset*.

The second data set for 3-D analysis consists of olfactory (axonal) projection (OP) image stacks of Drosophila larva. These images were used in the Diadem challenge [113] and like the previous dataset, these neurons are also imaged by a confocal microscope. These *OP-dataset* images are less noisy (due to a noise reduction technique which is inbuilt in the imaging software) and the contrast is better than the images in Condron data set. However, the neurons in this data set exhibit acutely complicated structural appearance in addition to occasional intensity heterogeneity along the neurite filaments.

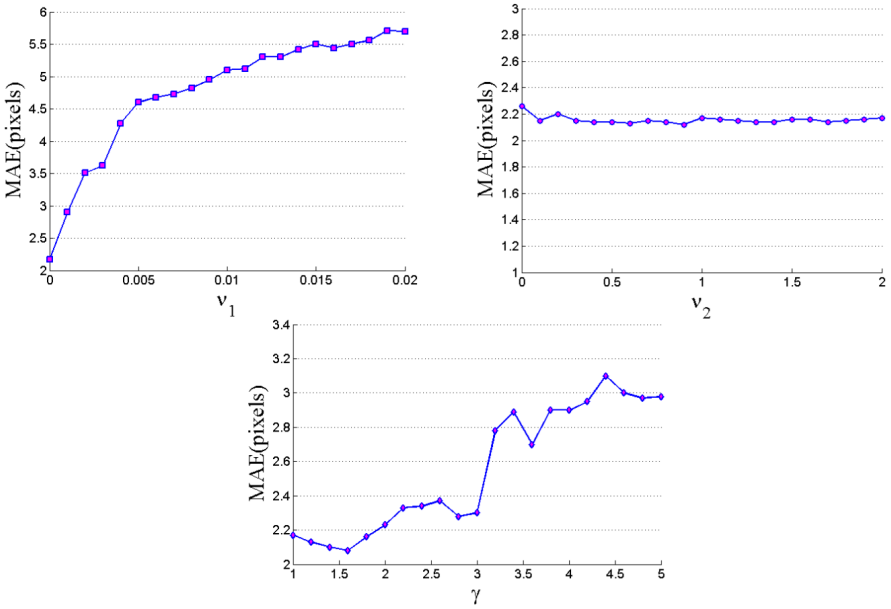


FIGURE 6.7: Sensitivity analysis of the parameters. The mean absolute error of the traced centerline are plotted in the vertical axis for different values of the tuning parameters.

6.6.2 Parameter selection

The level set evolution equation (6.10) depends on a few parameters. The evolution force $\mathcal{F}_{evolve}$ requires specifying the positive scalars a_0 and a_1 in (6.6) which controls the anisotropy of curve evolution. As we have discussed before, since the neurite thickness in our case does not vary considerably, we have adopted the isotropic case, as it requires lesser computation. Therefore, we choose $a_0 = 1$ and a sufficiently high value for a_1.

The smoothness of the evolved curve is controlled by the parameter ν_1 in (6.12). Effect of gradually increasing ν_1, keeping other parameters fixed results in an increased mean absolute error in tracing, as shown in Fig. 6.7. For our experiments, ν_1 is fixed at a value in the range $0 - 0.02$.

The attraction force defined in (6.20) depends on the weighing parameter ν_2 and the parameter γ controlling the local capture range. As we observe in Fig. 6.7 our algorithm is relatively robust to the choice of ν_2. However, we notice that a very low value of ν_2 restricts the attraction force from closing small gaps. For all our experiments, we select $\nu_2 = 1$. The term γ induces locality in the capture range for the attraction force. While a

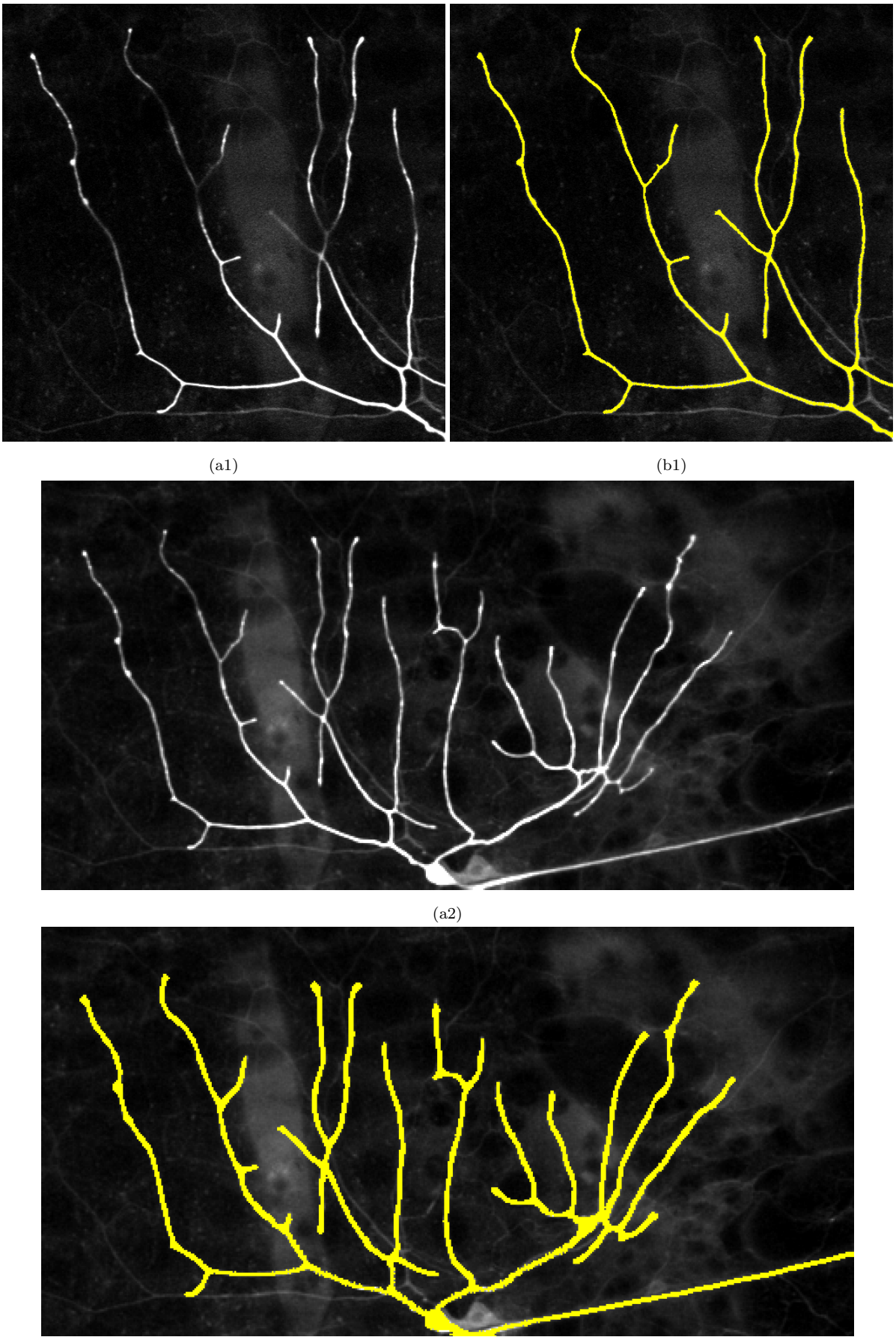
small value of γ can be too restrictive, a relatively high value attracts distant structures to be merged to the attracting component (see Fig. 6.7). Note that we are interested in connecting the disjoint structures over a local neighborhood. Based on our knowledge about the dataset, we observe that typically γ ranges between $0.2 - 1.5\mu m$ ($\approx$ 1–7 pixels) for our data. Setting these biologically inspired bounds on the range of γ, we proceed to select the value in the following manner. First, at any stage of segmentation, we compute the median distance ρ between all the segments, and update the value of γ as $\gamma^* = \rho/3$. If the updated value is beyond the pre selected upper or lower bounds, we select the closest boundary value for γ^*. This is repeated at each iteration to compute the attraction force.

Experimentally we have observed that the parameters Δ and ϵ can be prefixed to a particular value without affecting performance. For all experiments we choose $\Delta = 5$ pixels and $\epsilon = 1$ as suggested by the authors in [89].

6.6.3 2D segmentation via TuFF: qualitative results

Fig. 6.8 demonstrates the efficacy of TuFF in segmenting flat neuron cells from the subcuticle layer of Drosophila larva. The maximum intensity projection images of the neurites are shown in Fig. 6.8(a1) and Fig. 6.8(a2), and the respective segmentation results are shown in Fig. 6.8(b2) and (c2).

This example suggests that using TuFF, segmentation of vascular structures can be performed robustly, without extensive human intervention. The above shown images exhibit low contrast with significant filament discontinuities, and the neuronal processes have significant structural complexity with numerous branches and bifurcations.



(a1) (b1)

(a2)

(b2)

FIGURE 6.8: Segmentation of 2D flat neuron cells using TuFF. The images are courtesy Dr. G. Ascoli's lab, George Mason University, USA.

6.6.4 Efficacious handling of branch connectivity

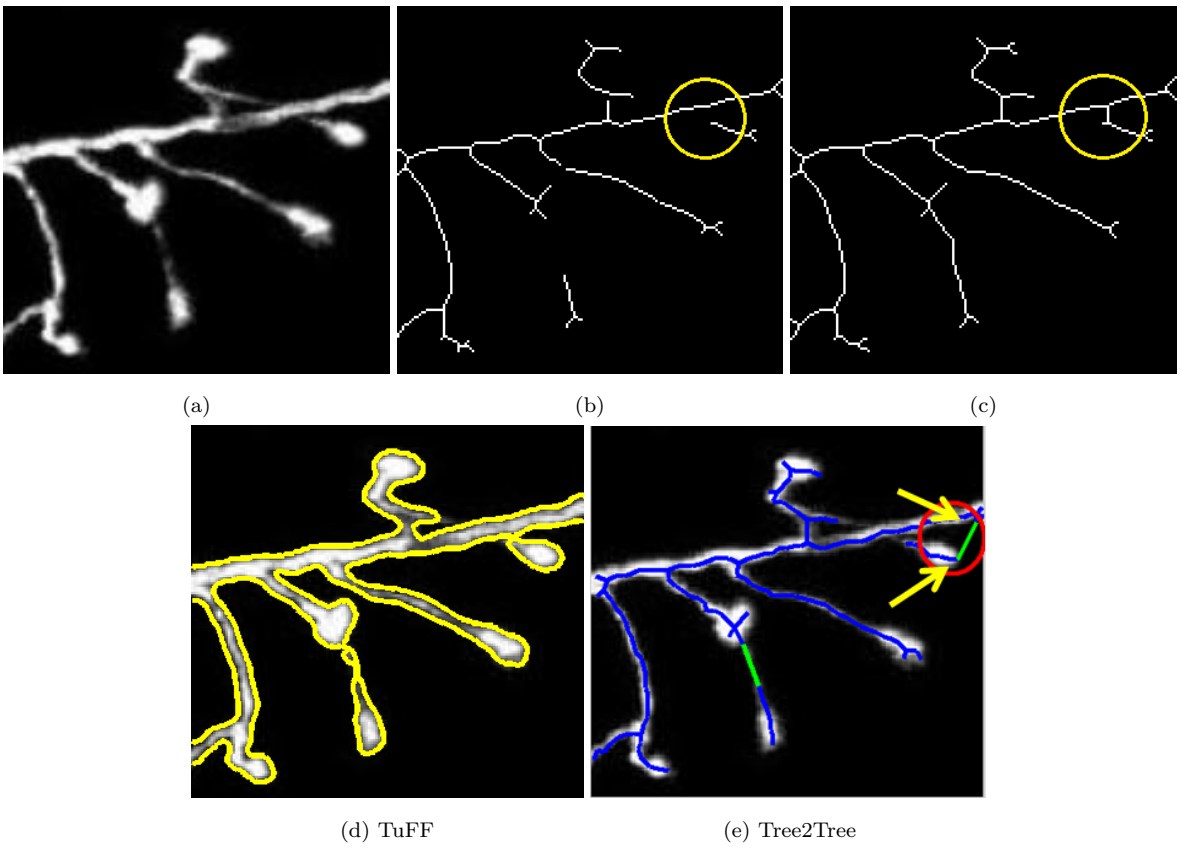
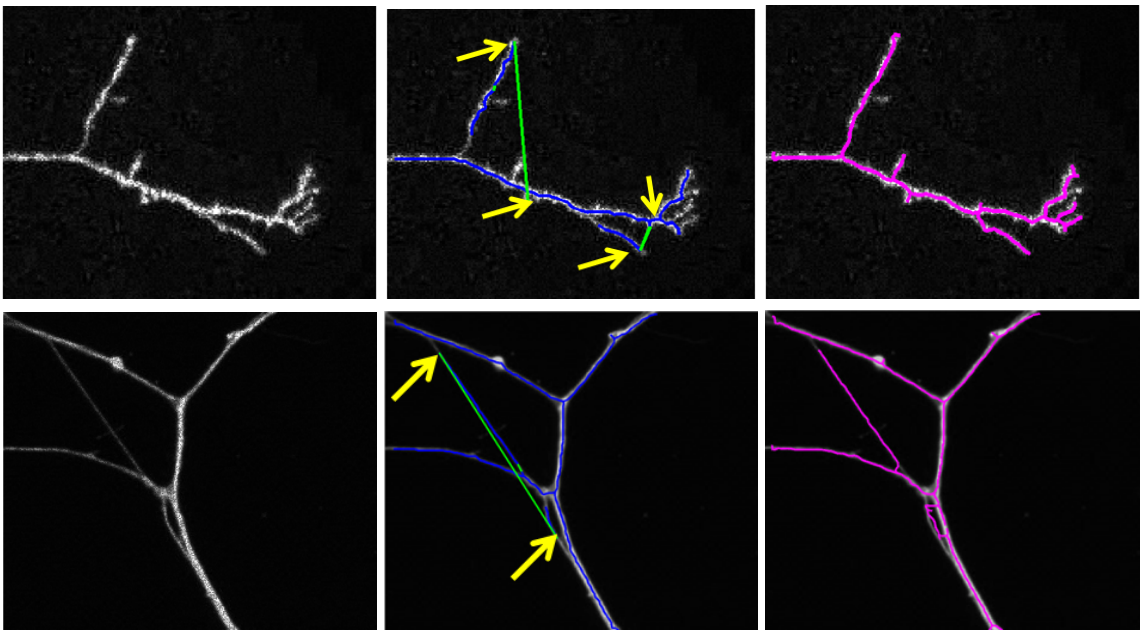


FIGURE 6.9: (a) 2-D neuron sub-image. (b) Centerline of the initial segmentation using [110]. The type B discontinuity is highlighted by the yellow circle. (c) Centerline obtained after segmentation using TuFF. (d) Final segmentation via TuFF. (e) Tracing using Tree2Tree. A typical error in connectivity is indicated by the arrows.

Previously, we have demonstrated the ability of TuFF to handle type A and type B discontinuities. In this section, we demonstrate the advantage of using TuFF over Tree2Tree [3] for determining branch connectivity. For this purpose, we show segmentation results on a few 2-D neuron images. The 2-D images are obtained from a maximum intensity projection of the corresponding 3-D stacks.

To set up Tree2Tree for segmentation, we follow the author's methodology of performing an initial segmentation to obtain a set of binary components. The component analysis stage of Tree2Tree then decides on the connection between the segments by analyzing their relative orientation. To initialize the level set for TuFF, we have used Otsu's segmentation, same as Tree2Tree, and the level set propagates according to (6.10).

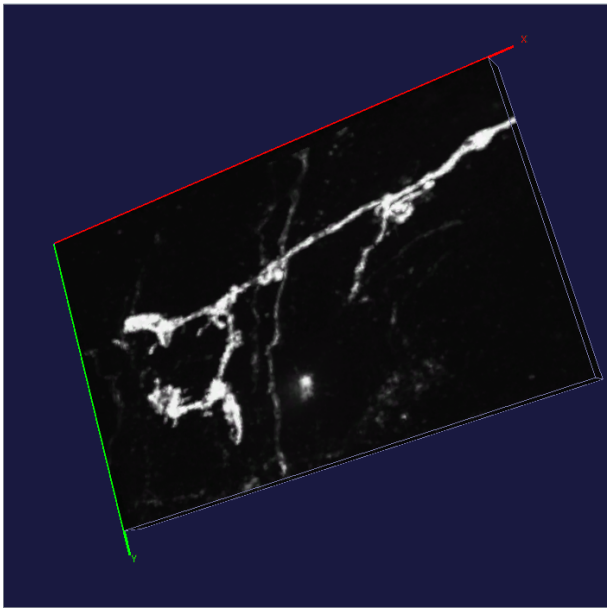


(a) MIP image (b) Tree2Tree (c) TuFF

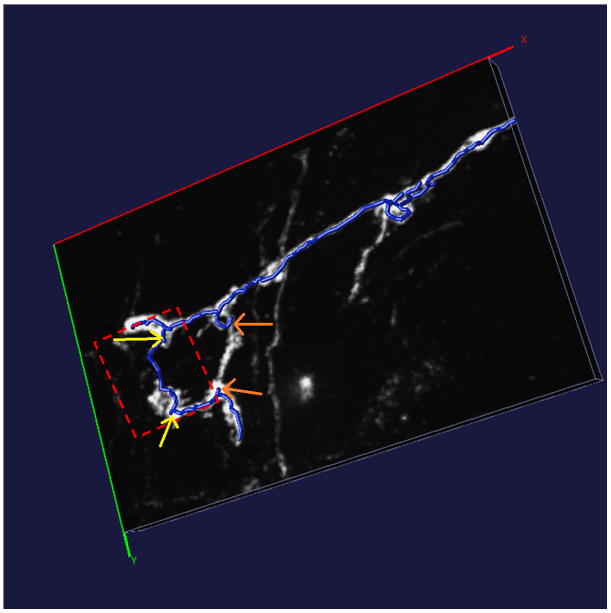
FIGURE 6.10: The first column shows sample 2-D neuron images. Tree2Tree [3] segmentation results are displayed in the second column. The edges linked by Tree2Tree are shown in green and the traced centerline is overlaid on the original image in blue. Excessive clutter restricts the efficiency of Tree2Tree, yielding improper connections, which are highlighted by the yellow arrows. The last column shows tracing output via TuFF (magenta).

Fig. 6.9 demonstrates an example where Tree2Tree creates improper connection, due to its inability to handle type B discontinuity. The level set based methodology in TuFF performs proper segmentation (shown in Fig. 6.9(c),(d)). It is evident that the type B gap is closed by TuFF, where Tree2Tree fails to do so (see Fig. 6.9(c) vs (e)).

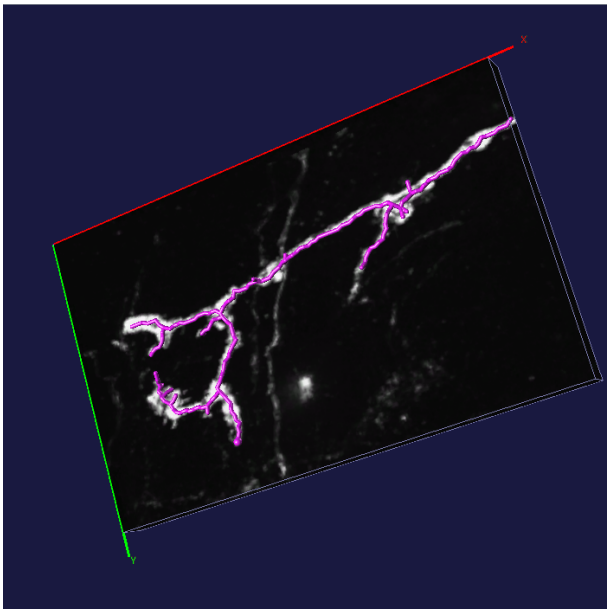
Two more examples are shown in Fig. 6.10 where Tree2Tree's tracing (shown in blue) creates incorrect branch connection as compared to TuFF (shown in magenta). The connection errors are highlighted by the yellow arrows. Tree2Tree segmentation results suggest lack of robustness of the component linking scheme for complex structures embedded in a noisy environment. Furthermore the initial segmentation step in Tree2Tree often fails to detect low contrast objects, which cannot be recovered in future, since the multistage pipeline of Tree2Tree is unable to recover lost neurite portions. Fig. 6.11 demonstrates another example, for the 3D case, where TuFF performs robust segmentation of the neurites compared to Tree2Tree (Fig. 6.11(c)), where connectivity



(a) 3D neuron image



(b) Tree2Tree



(c) TuFF

FIGURE 6.11: An example in 3D where TuFF avoids type B connectivity error. (a) A 3D confocal microscopy image of a neuron is shown. (b) The tracing results of Tree2Tree is shown in blue. Tree2Tree causes type B error, shown by the yellow arrows. The actual connection should have been between the segments marked by orange arrows. (c) Tracing result due to TuFF is shown in magenta. This figure is best viewed in color.

error due to Type B discontinuity is observed (Fig. 6.11(b)).

The above examples suggest that TuFF handles bifurcations and component gaps successfully, since level sets are well equipped in handling topological changes. Also, the specially designed attraction force component of TuFF makes segmentation robust in

cases where structure gaps result from very weak signal intensity (Fig. 6.9).

6.6.5 3D segmentation via TuFF: qualitative results

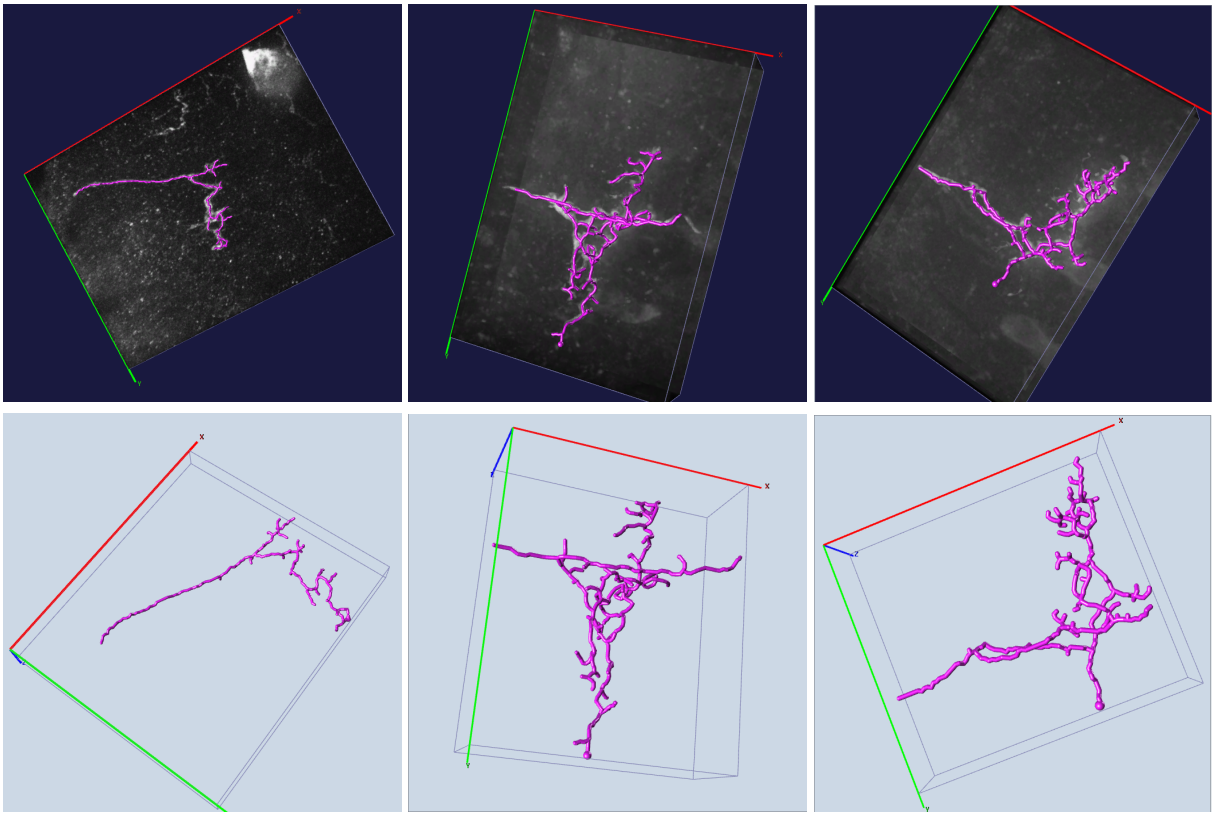


FIGURE 6.12: A few 3D neuron reconstruction examples using TuFF. The images are from the Condron dataset. The first row shows reconstructed neurites, overlaid on the original stack. The 3D reconstructions are shown in the second row.

Fig. 6.12 shows a few 3D neuron reconstructions (tracings) of Drosophila neurons using TuFF for segmentation. Digital reconstruction is obtained by computing the centerline of the segmented neuron, followed by spline fitting to each branch of the resulting skeleton graph. The neuron tracing results are shown in magenta, in the first row. The second row shows the neuronal structure, which is embedded in a *.swc* file format. We have used the Vaa3D [39] toolkit for visualizing these digital reconstructions.

6.6.6 Comparison of segmentation performance

In this section we present a comparative segmentation performance analysis of the proposed method TuFF versus three popularly used neuron tracers. The ground truth data for segmentation is obtained by manually selecting points on the neuron structure and joining them manually in a manner that the morphological structure is preserved. The Vaa3d software [39] is used for creating the ground truth. To evaluate the performance of TuFF, we compare its performance to the following algorithms.

6.6.6.1 Graph Augmented Deformable (GD) model [1]

This semi automatic tool is extensively used for its relatively simple working methodology, which consists of a manual seed selection step followed by automated seed joining process by using graph theoretic techniques. Since the algorithm's efficacy is inversely proportional to the spatial distribution of selected seed points, we only select the neuron terminal points as the set of seeds. As the seed selection is performed manually, a practice which TuFF avoids, we believe that selecting the minimal set of seeds is essential to maintain fairness of comparison. Sample tracing results using this algorithm are shown in yellow.

6.6.6.2 Neuronstudio [2]

Neuronstudio is one of the state of the art publicly available automatic neuron segmentation software which is heavily used by biologists for tracing purpose. We have seen that segmentation accuracy of NeuronStudio is affected by the choice of the initial seed point. For each image in our dataset, we experiment with several initial seed locations and finally choose the one which yields the best visual segmentation result. Neuronstudio segmentation results are shown in orange color.

6.6.6.3 Tree2Tree [3]

As discussed earlier, Tree2Tree belongs to the category of seed independent neron segmentation methods. Setting up Tree2Tree requires an initial segmentation stage, followed by graph-theoretic component linking procedure. The segmentation results of Tree2Tree are shown in blue color.

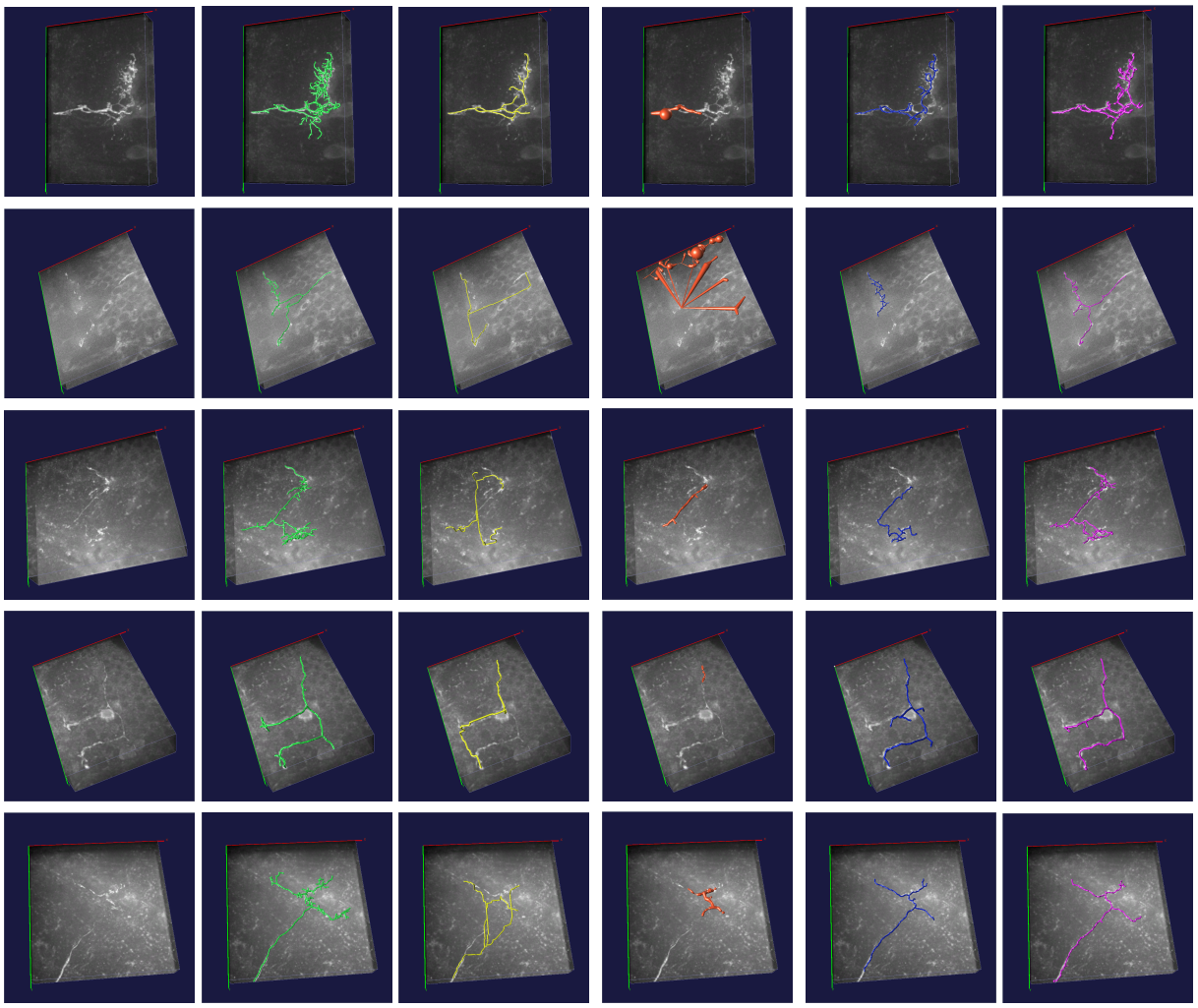
For each of the above mentioned algorithms and TuFF, we first obtain the segmentation followed by neuron centerline detection. A cubic spline is fitted to each branch of the detected centerline. This spline fitted centerline of the neurons represent the tracing results.

6.6.7 Qualitative performance analysis

6.6.7.1 Results on Condron data set

Fig. 6.13 shows the performance of the above mentioned neuron tracers on five representative neurons chosen from the Condron dataset. The 3-D stacks are shown in the first column, followed by manual ground truth segmentation in the second column (shown in green). Tracing results using GD model [1] is plotted in yellow in the third column. The fourth and fifth columns show segmentation output using the automated techniques Neuronstudio and Tree2Tree (plotted in orange and blue color) respectively. Finally, the last column shows the neuron tracing due to TuFF (plotted in magenta).

It may be observed that these images are in general noisy, which makes the segmentation task difficult. Moreover, high structural complexity of the neurons require sophisticated mechanism to preserve the structural morphology. The severity of contrast variation and low SNR pose difficulty for the GD model. Even with manually selected terminal nodes, it is seen that the semi-manual tracer performs incorrect segmentation (Fig. 6.13, second column, rows 2-5). This is primarily due to the inability of the local search based technique fails to identify the actual filamentous path in presence of clutter.



(a) 3D stack (b) Ground truth (c) GD model [1] (d) NeuronStudio [2] (e) Tree2Tree [3] (f) TuFF

FIGURE 6.13: Tracing results on 3D images of the UVA-Condron dataset. First column shows the original images, followed by the tracing outputs of the different algorithms. Tracing results of TuFF are shown in the last column in magenta.

Furthermore, human assisted neurite termination detection proved to be a difficult and time consuming problem in these images owing to the high structural complexity.

Neuronstudio performs particularly poorly in these examples. The major reason can be attributed to the lack of continuity in the neurite structure and high signal variation, which forces the algorithm to converge prematurely. Also, the cluttered environment is detrimental to the performance of the local voxel scooping process of Neuronstudio. This results in under segmentation and sometimes, incorrect segmentation due to leakage of the region growing technique.

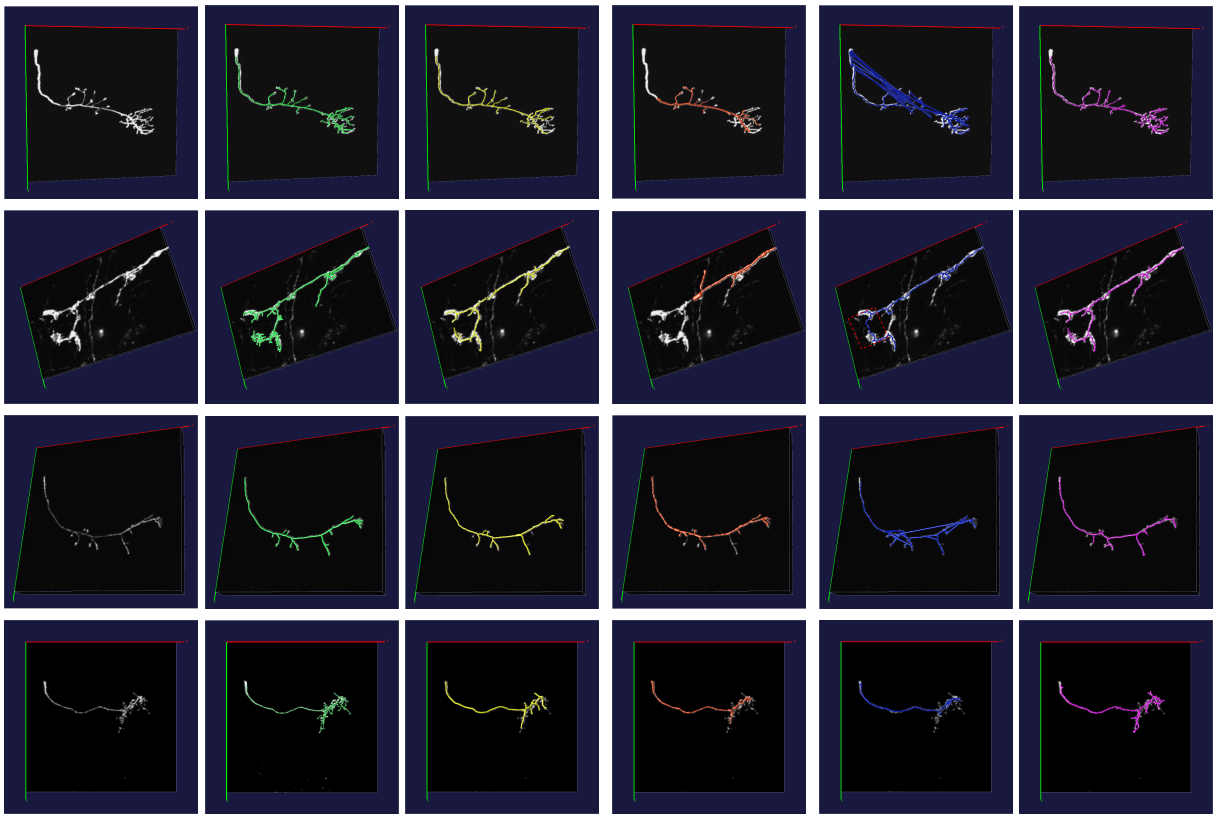
Tree2Tree outperforms Neuronstudio, especially when the component linking algorithm is able to determine proper connectivity. We observe that Tree2Tree performs well if the initial segmentation step is reliable. However, under segmentation is an inherent problem in Tree2Tree due its inability to incorporate additional neuronal structures in its solution after initial thresholding.

On the other hand, TuFF performs segmentation efficiently, even in cluttered environment. A close inspection would reveal that important morphological entities like bifurcation points and branch locations are preserved (see Fig. 6.13 rows 2,3 and 4), while the iterative directional region growing scheme prevents under segmentation of neurons.

6.6.7.2 Segmentation results on OP dataset

These image stacks exhibit relatively higher signal intensity than the Condron data set. However, neuron tracing is still a challenging task owing to their complicated structure and sudden intensity variations in the neurites, creating a fragmented, discontinuous appearance. This often results in type B discontinuity which demands sophisticated analysis. Fig. 6.14 compares the segmentation results for above mentioned algorithms.

Reduction in background clutter and increased signal intensity assists the semi automatic GD-model tracer. Since the images exhibit significant improvement in contrast, manual detection of seeds is less stressful. Still, the complicated structure of a few images (Fig. 6.14, row 1 for example) makes manual seed selection demanding. Performance of Neuronstudio also shows slight improvement in this dataset. However, despite brighter foreground and less noise, this local tracing scheme shows tendency to stop at intensity gaps, which needs to be modified manually at a later stage. On the other hand, it is observed that Tree2Tree's performance degrades significantly for this dataset. This is primarily due to a large number of improper branch connections. This connectivity error occurs mostly due to Tree2Tree's inability to handle type B discontinuities (Fig. 6.14,



(a) 3D stack (b) Ground truth (c) GD model [1] (d) NeuronStudio [2] (e) Tree2Tree [3] (f) TuFF

FIGURE 6.14: Results on the images of the OP dataset. First column shows the original images, followed by the tracing outputs of the different algorithms. Tracing results of TuFF are shown in the last column in magenta.

rows 1-3). In fact, even in relatively high SNR images Tree2Tree under performs significantly by extracting an improper structural morphology of the neurons. TuFF, however demonstrates good performance on these images by virtue of its ability to handle structure gaps automatically. The segmentation results are shown in the last column of Fig. 6.14. A qualitative assessment of the algorithm's performance is presented in the following sections.

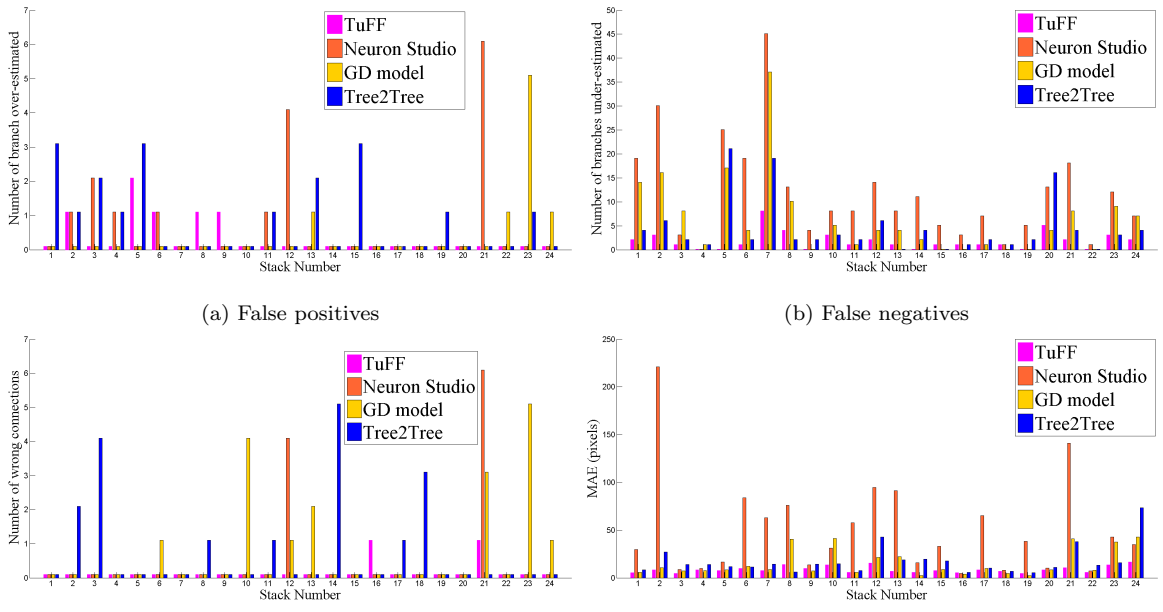
6.6.8 Quantitative Performance Analysis

To quantify the segmentation performance, we identify four measures which reflects the efficiency of a particular neuron tracer. These are as follows: number of over-estimated branches (Fig. 6.15(a)), number of unidentified/missed branches (Fig. 6.15(b)), total

number of incorrect branch connections (see Fig. 6.15(c)) and finally the mean absolute error in the traced centerline with respect to the ground truth.

The number of over determined/missed branches reflect the adequacy of an algorithm in respecting the morphology of the imaged neuronal structure. This quantification of the segmentation quality is performed by a human expert. However, since even the ground truth data is susceptible to subtle errors in computing the 3D skeleton, we have disregarded small branches (less than 5 units in length) from the analysis. The graphs in Fig. 6.15(a) and (b) suggests that over the whole data set, TuFF outperforms the competing algorithms in a majority of cases. It is observed in a few cases that Neuronstudio in particular misses a large number of branches, due to its inability to deal with fragmented structure.

The number of incorrect branch connections (Fig. 6.15(c)) indicate an algorithm's ability to tackle discontinuities. Indeed, improper connections often result when signal



(a) False positives

(b) False negatives

(c) Incorrect connection

(d) MAE

FIGURE 6.15: (a)-(c): Quantitative performance of the four neuron tracers TuFF (pink), Neuron Studio (orange), GD model [1] (yellow) and Tree2Tree (blue) in terms of number of over-estimated branches, number of under-estimated branches and total number of wrong connections respectively. (d) quantifies the tracing accuracy in terms of mean absolute error (6.22).

heterogeneity is significant. Apart from a few occasions, TuFF demonstrates its superiority in handling discontinuities better than other automated methods.

To perform quantitative analysis of the traced neuron centerline, we compute the mean absolute error (MAE) of the obtained trace against the manually acquired ground truth. If $\mathcal{P} = \{p_1, \ldots, p_n\}$ and $\mathcal{Q} = \{q_1, \ldots, q_m\}$ denote the set of traced coordinates for a neuron, the mean absolute error (in pixels) between the traces is given by

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n} \min_{j} |p_i - q_j| + \frac{1}{m}\sum_{i=1}^{m} \min_{k} |q_i - p_k| \tag{6.22}$$

$\forall j \in \{1, .., m\}$, $\forall k \in \{1, .., n\}$. Mean absolute errors for the 3-D images are plotted for each algorithm in Fig. 6.15(d). It is observed that TuFF outperforms the automated tracers Tree2Tree and Neuronstudio in almost all of the cases, except for the 8^{th} and 16^{th} stack, where Tree2Tree and Neuronstudio perform marginally better. Also, TuFF successfully competes with the semi-automatic GD-model, even outperforming it in some images in the Condron dataset.

The mean, median and standard deviation MAE of the four algorithms are reported in Table 6.1. This suggests that on a whole TuFF outperforms its competitors with a mean and median MAE of 8.81 (pixels) and 7.95 (pixels) respectively. TuFF also exhibits 75% improvement of mean error over the second best performer, which is the semi-automatic tracer of Peng *et al.* If we compare its efficacy against the fully automated techniques, we obtain an improvement of over 98% over Tree2Tree, while Neuron Studio is outperformed with an improvement of greater than 400%. Also, the error standard deviation of TuFF is only 3.4 as compared to 50.6, 14.03 and 15.08 for Neuronstudio, GD-model and Tree2Tree.

Table 6.1: Comparison of MAE

	TuFF	**Neuron Studio**	**GD model**	**Tree2Tree**
Avg. MAE	8.81	79.98	15.41	17.62
Median MAE	7.95	34.06	8.54	13.98
Std. Dev	3.4	50.6	14.03	15.08

The visual segmentation results and the quantitative results presented here suggests the efficiency of TuFF in segmenting structurally complex neurons from cluttered confocal microscope images.

6.7 Further improvements

Until now, we have described the application of TuFF for the purpose of segmenting neurons from 2D and 3D imagery. However, the basic formulation of TuFF is applicable to other segmentation problems, which involve tubular objects. As a result, the mathematical formulation of TuFF involves a technique for identifying the tubular structures from the noisy images. The hessian based methodology of Frangi [4] is used here, as well as in Tree2Tree-2, to get a vesselness function and the vessel orientations. However, this popular tool has a severe drawback; it is unable to detect key locations such as filament bifurcations and end points. Furthermore, the obtained vesselness response diminishes directly with the vessel signal intensity. Therefore, in many cases, the Frangi filter produces nonlinear vesselness maps with significant false negatives at vessel junctions, terminals or at locations with relatively lower foreground intensity. This creates major hurdles in Tree2Tree/Tree2Tree-2 [3, 21], and while the local attraction force in TuFF tries to mitigate this artifact, we hypothesize that a robust vessel identification scheme would result in refined segmentation. The proposed method uses specialized steerable filters for ridge detection that incorporates a local processing scheme to enhance vessel detection. Finally, we show that in the variational paradigm, it is also possible to incorporate edge based energy term in the TuFF equation, which can assist segmentation in some applications. The extensions of TuFF and a few non biological applications are discussed in Appendix B.

6.8 Discussion

In this paper we have presented an automated neuron segmentation algorithm which can segment neurons from both 2-D and 3-D images. The proposed framework is suitable for tracing highly fragmented neurite images, and is capable of processing the structure discontinuities automatically, while respecting the overall neuron morphology. Connectivity analysis is performed in a level set framework which presents a nice and simple alternative to graph based techniques which may introduce undesired branches in segmentation. The efficiency of TuFF is further demonstrated by its superior overall quantitative performance where it outperforms peer algorithms, including a semi manual tracer. The salient highlights of this proposed method is discussed below.

First, we avoid human intervention in terms of seed point selection. Automated initialization of the level set is performed by Otsu's global thresholding [110] followed by noise removal using morphological area open operators [112]. The level set function is computed from this initialized segments using binary distance transform.

Second, TuFF presents a natural framework to process both type A and type B discontinuities (Fig. 6.6). This is a major improvement over Tree2Tree [3], where the inability to handle type B discontinuity introduces several false connections in the solution.

Finally, TuFF is capable of joining broken neurite fragments even in complete absence of signal. The proposed attraction force field is independent of the local signal intensity and depends only on the morphology and relative positioning of the connected components. This feature improves on the widely used local intensity seeking neuron tracers [2], which are susceptible to illumination variation in the images of neural structure. The TuFF guided evolution energy is combined with the attraction force component in a mathematically elegant, integrated fashion as opposed to a multistage sequential processing pipeline.

Chapter 7

Conclusion

This is the concluding chapter

Appendix A

Dictionary Learning Level Set

A.1 Dictionary Learning Level Sets (DL2S)

The primary motivation for DL2S is similar to that of L2S– performing segmentation in presence of intensity inhomogeneity. In L2S [29], we generalized the Chan-Vese model by approximating the foreground and background regions as a piecewise polynomial function computed via linear combination of a few Legendre basis functions. This can be viewed from the perspective of low dimensional approximation of a signal. While Chan-Vese's method is a form of extreme dimensionality reduction (due to the piecewise constant assumption), L2S achieves a balance between reduction of dimensionality and accurate intensity modeling.

Despite its merits, L2S suffers from certain issues. First, the segmentation quality relies heavily on the number of chosen basis functions. Second, L2S suffers from scalability issues since the pre-specified bases cannot represent any arbitrary intensity variation. As it turns out, recent research in the field of sparse modeling and dictionary learning [109,114–117] have shown that for a given set of training data, one can obtain an optimal set of basis elements (atoms) to represent a signal. This is the main highlight of DL2S –*instead of explicitly specifying the set of basis elements, we estimate an optimal set of*

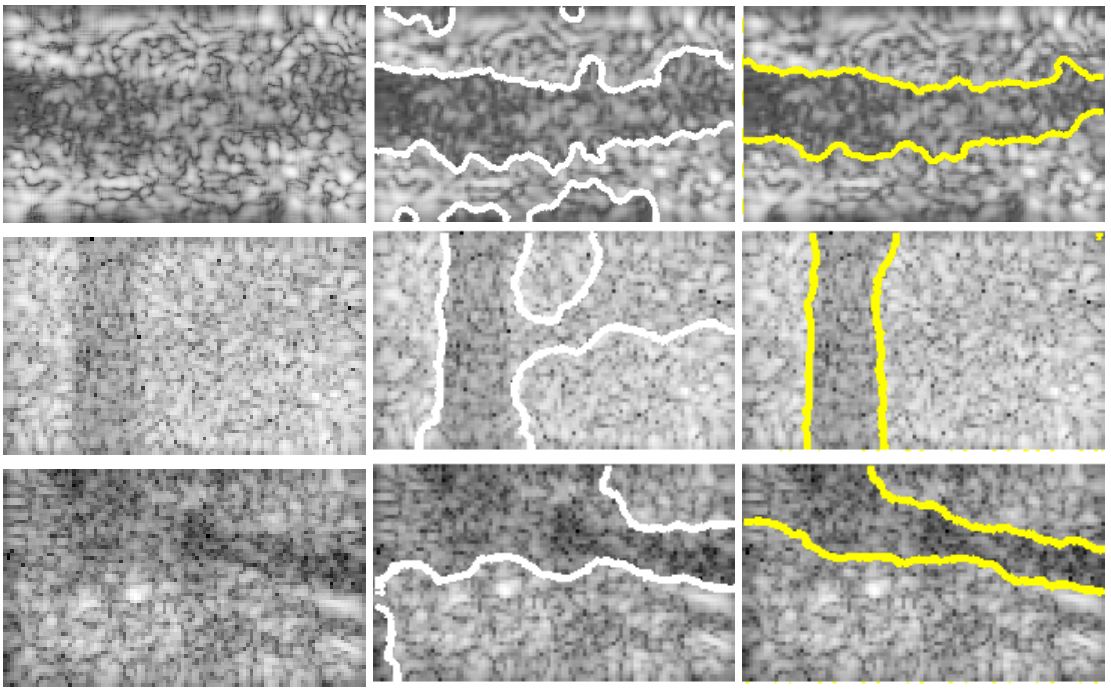


FIGURE A.1: Segmentation results of Chan-Vese [89] (white) and DL2S (yellow) on three C-mode ultrasound images captured with a portable scanner.

bases from the set of training images using dictionary learning. To demonstrate our technique, we choose an important segmentation problem for ultrasound imaging. Blood vessels are imaged in C-mode using a portable, low cost, battery operated ultrasound device. Our objective is to segment the vessel boundary to assist medical practitioners for performing phlebotomy application such as intravenous needle placement (see Fig. A.1). Images captured using these portable devices suffer from low contrast, noise and speckle in addition to non-linear illumination of the objects which makes segmentation challenging.

A.1.1 Methodology

A generalized version of Chan-Vese's model can be formulated as follows:

$$\begin{aligned}\mathcal{E}(\phi, A, B) \;= &\int_{\Omega} |f(\mathbf{x}) - \sum_{i=0}^{k} a_i d_i(\mathbf{x})|^2 m_1(\mathbf{x}) d\mathbf{x} + \int_{\Omega} |f(\mathbf{x}) - \sum_{i=0}^{k} b_i d_i(\mathbf{x})|^2 m_2(\mathbf{x}) d\mathbf{x} \\ &+\nu \int_{\Omega} |\nabla H_\epsilon(\phi)| d\mathbf{x} + \lambda \left(||A||_2^2 + ||B||_2^2 \right) \end{aligned} \tag{A.1}$$

Here $\mathbb{D}_k(\mathbf{x}) = [d_1(\mathbf{x}), \ldots, d_k(\mathbf{x})]^T$ is a dictionary which will be discussed in detail shortly. $d_0(\mathbf{x}) = \mathbf{1}$. $d_1, \ldots, d_k$ are dictionary elements or atoms which are used to model the non-linearity in the intra-region intensities of the images. The third term in (A.1) introduces smoothness in the solution, which is controlled using the parameter ν. $\mathbf{a} = [a_0, \ldots, a_k]^T, \mathbf{b} = [b_0, \ldots, b_k]^T$ are $(k+1)$ dimension real valued coefficient vectors. The parameter λ reduces over-fitting, by constraining the ℓ_2 norm of the coefficient vectors.

With $k = 0$, (A.1) reduces to the piecewise constant model in (4.18). In other words, (A.1) generalizes the traditional Chan-Vese technique by introducing capability to handle heterogeneous image regions. Here $d_1, \ldots, d_k$ can be interpreted as *detail functions* to model the intensity variation in conjunction to the constant illumination term d_0. As earlier, (A.1) can be optimized with respect to ϕ, A and B using alternating minimization.

Naturally, a question arises– how to select $d_1(\mathbf{x}), \ldots, d_k(\mathbf{x})$? It was shown in [29] that high quality segmentation results can be obtained by using a few Legendre basis functions. However, we hypothesize that if a dataset of example images is available, we can enhance the segmentation performance by learning an optimal set of basis functions (dictionary elements) for region intensity approximation instead of using a predefined set of basis. For the application described in this paper, we are concerned with sets of ultrasound images, imaged using similar type of devices. The multi-depth images are captured at the same scale, and are preregistered. As a result, we have the provision to learn these functions $d_i(\mathbf{x})$ directly from the dataset, rather than relying on *ad hoc* procedures for selecting the same.

A.1.2 Intensity modeling with dictionary learning

Sparse coding techniques have gained popularity recently. Such algorithms have been used for a multitude of applications ranging from image denoising, inpainting, restoration, classification, retrieval etc [114–117]. Given a set of training data, the goal of dictionary

learning is to compute a set of basis elements, also called *atoms*, such that each training data can be represented as a linear combination of only a few of these atoms. The key idea is to utilize the underlying sparsity of the training data, while minimizing the reconstruction error. Mathematically, if $\mathbb{F} = [\mathbf{f}_1, \ldots, \mathbf{f}_N]$ denotes the set of N discretized, vectorized and mean subtracted training images, we can use dictionary learning technique to compute the dictionary $\mathbb{D}_k = [\mathbf{d}_1, \ldots, \mathbf{d}_k]^T$ mentioned in (A.1) by solving the following optimization problem

$$\begin{aligned} \mathbb{D}_k &= \arg\min_{\mathbb{D}, \mathbf{y}_i} \sum_{i=1}^{N} ||\mathbf{f}_i - \mathbb{D}^T \mathbf{y}_i||_2^2 \\ \text{such that } &||\mathbf{y}_i||_0 \leq \theta, \quad \forall i = 1, ..., N. \end{aligned} \tag{A.2}$$

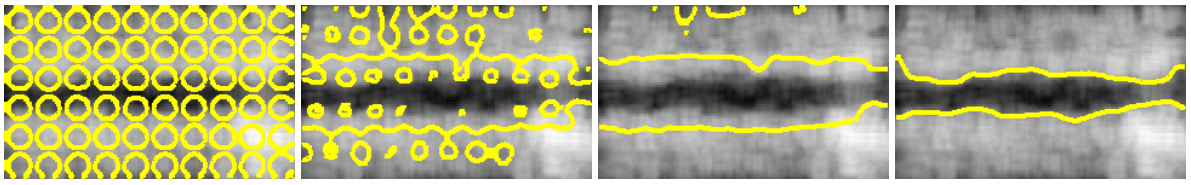
where y_i is a coefficient vector corresponding to the i^{th} training image and θ is a scalar which dictates the level of sparsity. There are a number of methods in the literature that use some approximation to solve the hard optimization problem (A.2). For example, k-SVD [109] combines a greedy methodology using orthogonal matching pursuit algorithm to provide a fast solution to this problem. Dictionary learning exploits sparsity in the data (A.2) by constraining ℓ_0 norm of the coefficients.

A.1.3 DL2S curve evolution

Let us denote $\hat{\mathbb{D}}_k = [d_0(\mathbf{x})^T \ \mathbb{D}_k(\mathbf{x})]^T$. We first try to minimize (A.1) with respect to A and B, by taking derivatives and setting the result to zero. A closed form solution is obtained as follows:

$$\hat{\mathbf{a}} = \ [K + \lambda \mathbb{I}]^{-1} \int_{\Omega} \hat{\mathbb{D}}(\mathbf{x}) f(\mathbf{x}) m_1(\mathbf{x}) d\mathbf{x} \tag{A.3}$$

$$\hat{\mathbf{b}} = \ [L + \lambda \mathbb{I}]^{-1} \int_{\Omega} \hat{\mathbb{D}}(\mathbf{x}) f(\mathbf{x}) m_2(\mathbf{x}) d\mathbf{x} \tag{A.4}$$



(a) (b) (c) (d)

FIGURE A.2: Evolution steps shown for our algorithm (a) initialization, (b) iteration=20, (c) iteration=60, (d) final contour.

where [.] denotes a matrix. $[K]$ and $[L]$ are $k \times k$ Gramian matrices [104], in which $(i,j)^{th}$ entries are obtained as

$$[K]_{i,j} = m_1(\mathbf{x}) \langle d_i, d_j \rangle \text{ and } [L]_{i,j} = m_2(\mathbf{x}) \langle d_i, d_j \rangle \tag{A.5}$$

$0 \leq i,j \leq k$ and $\langle , \rangle$ denotes the Euclidean inner product operator. With the updated coefficient vectors, we can now minimize (A.1) with respect to ϕ using variational calculus. We obtain the following partial differential equation using gradient descent technique for minimization.

$$\frac{\partial \phi}{\partial t} = \left[-|f(\mathbf{x}) - \hat{\mathbf{a}}^T \hat{\mathbb{D}}_k(\mathbf{x})|^2 + |f(\mathbf{x}) - \hat{\mathbf{b}}^T \hat{\mathbb{D}}_k(\mathbf{x})|^2 \right] \delta_\epsilon(\phi) + \nu \delta_\epsilon(\phi) \nabla \cdot \left(\frac{\nabla \phi}{|\nabla \phi|} \right) \tag{A.6}$$

We initialize $\phi|_{t=0} = \phi_0$ and $\frac{\delta_\epsilon(\phi)}{|\nabla \phi|} \frac{\partial \phi}{\partial \hat{n}} = 0$ at the domain boundary. The gradient flow of DL2S is computed iteratively by discretizing (A.6) using a finite difference scheme. Fig. A.2 shows different steps of the curve evolution. Fig. A.2(a) shows the initialization Fig. A.2(b) and (c) shows two intermediate steps and Fig. A.2(d) shows the finally evolved curve.

A.1.4 Analysis of DL2S

The Chan-Vese method performs segmentation by approximating an image $f(\mathbf{x})$ by a piecewise constant image $g(\mathbf{x})$. To make the model more flexible, we add higher order

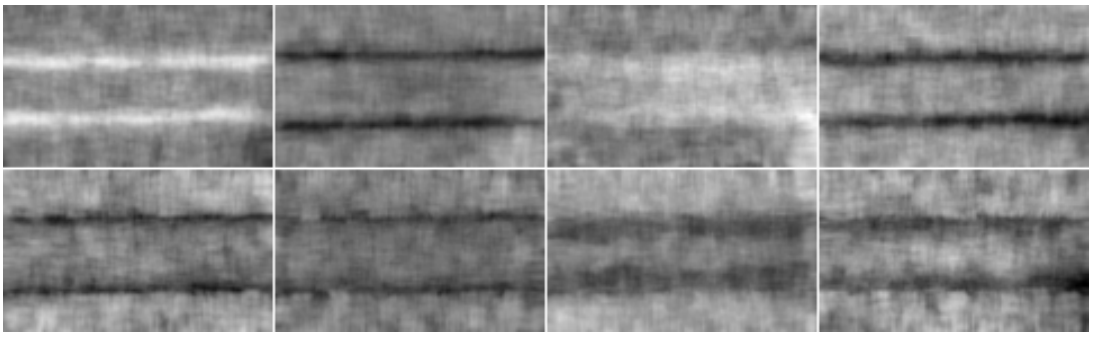


FIGURE A.3: Eight dictionary atoms learned from the mean subtracted images for a phantom image category.

terms which can capture the intensity variations in the regions. Going by the intuition of Chan and Vese, it is fair to approximate the mean image of a dataset as a piecewise constant image.

Assuming a mean image which is approximately piecewise constant, the dictionary atoms learned from the mean subtracted dataset can be utilized to provide the non-linear variation necessary to model the intensity inhomogeneity. The energy functional in (A.1) essentially incorporates this idea in a mathematical framework. One can also think of the dictionary atoms as incorporating higher order details, learned to suit our dataset. The dictionary atoms computed for a particular ultrasound image dataset is shown in Fig. A.3. The dictionary atoms aid in retaining the more significant image properties and compactly represent the dataset.

DL2S is applicable where a set of pre-registered training data is available, for example multi-depth ultrasound images of blood vessels, in temporal image sequences of biomedical objects such as carotid artery, heart videos. In applications involving a temporal image sequence, the first few frames of the can be treated as the training data to learn the dictionary, which can be exploited to segment the subsequent frames.

A.1.5 Experimental Results

We use five different sets of images to evaluate the performance of our algorithm. Out of them, three datasets contain images of medical phantoms which mimic human veins. These phantoms are generally used by medical practitioners for device calibration. The remaining two datasets consists of human vein images, captured *in vivo*. Each dataset contains approximately 18 to 60 images, captured in C-mode using a portable, battery operated ultrasound scanner. The different images in a given set correspond to the image of a vein at various depths. Note that each dataset consists of registered blood vessel images. The vessel orientation and scale are also consistent. A separate dictionary is computed using the mean subtracted images for each of the datasets.

Dependency on contour initialization: We show the performance of our algorithm using both manual and automatic initialization methods. The segmentation results with manual and automatic initialization for Chan-Vese [89], L2S [29] and DL2S are shown in Fig. A.4 for the same image. We observe that the segmentation performance of L2S drops significantly for automatic initialization, which is also true for Chan-Vese method. In comparison DL2S has similar segmentation results for both initialization technique. Quantitative evaluation of performance based on initialization is provided in Table I.

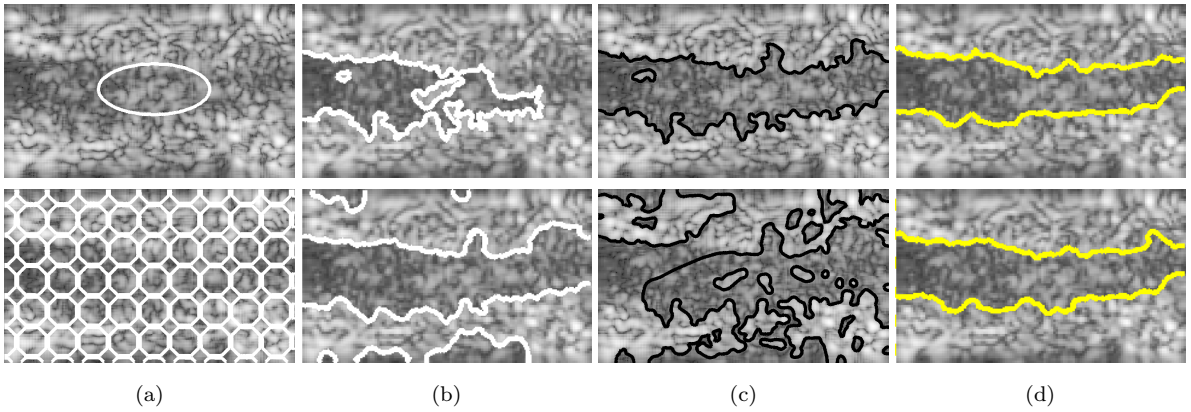


FIGURE A.4: Comparison of segmentation results using manual and automatic initialization methods. (a) initialized contour (b) segmentation results of Chan-Vese (white), (c) segmentation via L2S (black) and (d) segmentation via DL2S model (yellow)

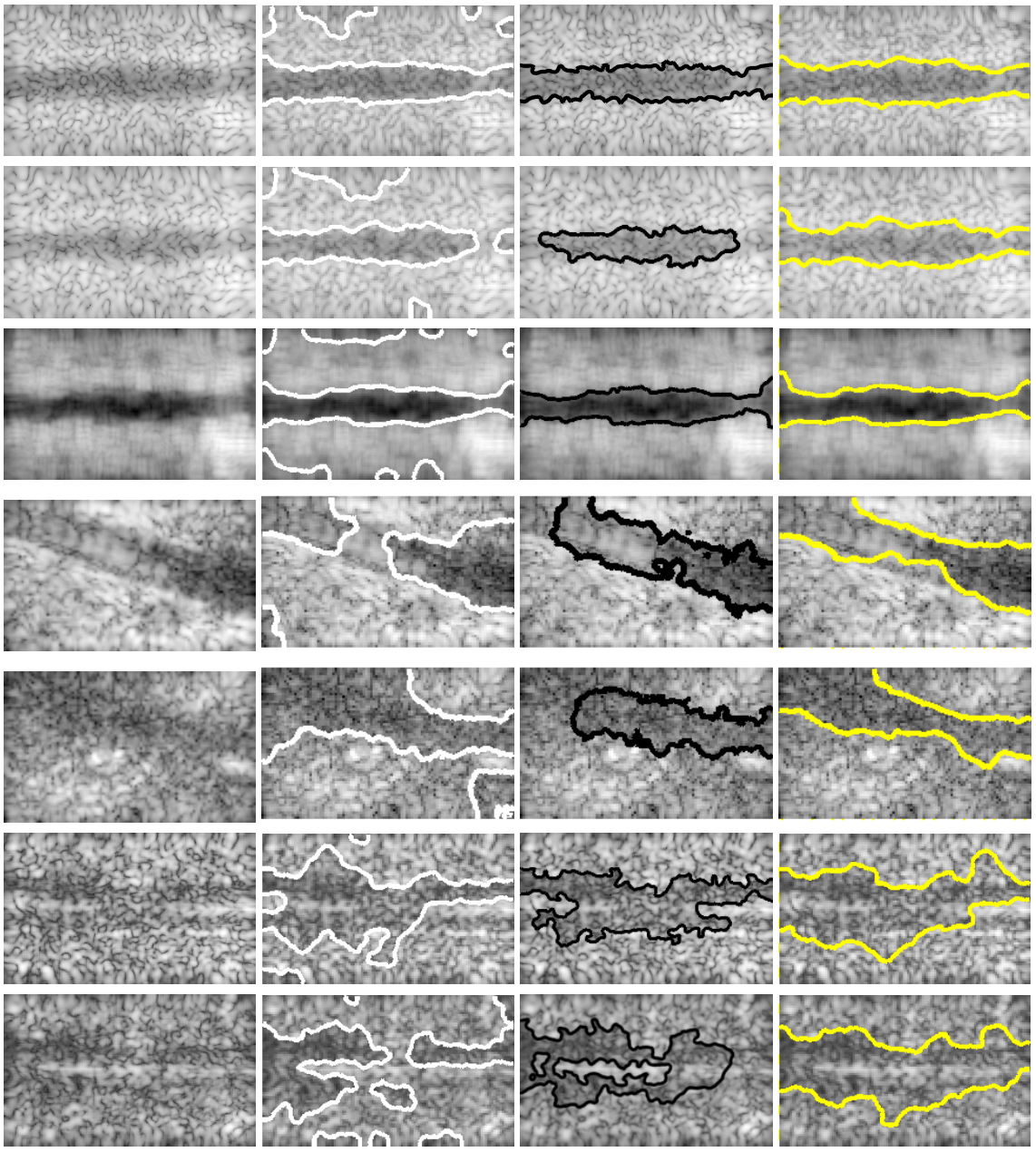
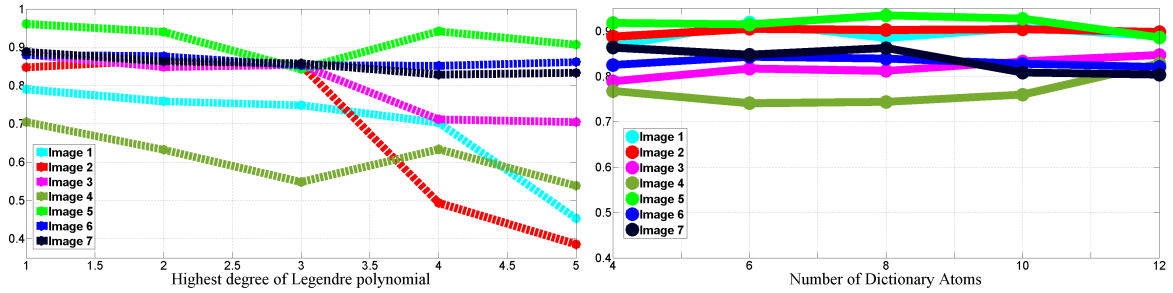


FIGURE A.5: Segmentation comparison of DL2S with Chan-Vese and L2S is shown here. The original C-mode ultrasound images captured with a portable scanner are shown in the first column. Segmentation results of Chan-Vese (white), L2S (black) and DL2S model (yellow) on these images are shown in columns 2, 3 and 4 respectively.

Dependency on dictionary size: We perform sensitivity analysis experiment to study the performance of the segmentation algorithm with changing dictionary size. The Dice coefficient are plotted (along Y-axis) for L2S [29] (Fig. A.6 (a)) ans DL2S (Fig. A.6 (b)) to show the performance with changing basis / dictionary size (along X-axis) for 7 randomly chosen images. In comparison to L2S, where performance decreases with



(a) (b)

FIGURE A.6: (a) Dice coefficient for L2S with changing number of basis functions, (b) Dice index for the same images plotted for DL2S with changing size of dictionary

increasing number of basis functions, DL2S exhibits a more stable performance. Based on experiment evaluation, we fix the number of dictionary elements $k = 8$ which is at most 50% of the size of the smallest dataset. We choose sparsity inducing parameter $\theta = 3$ such that about 30% or less number of atoms can be used for representing the training images.

Quantitative comparison of segmentation: Fig. A.5 shows the segmentation performance of Chan-Vese (white) [89]), L2S (black) [29] and DL2S (yellow) Fig. A.5 shows that DL2S is able to capture the blood vessels more appropriately in presence of severe contrast and intensity inhomogeneity. A quantitative comparison for five datasets as shown in Table A.1. The Dice index is evaluated for the three algorithms. Here s_g denotes the ground truth segmentation and s_t is the segmentation result for DL2S, Chan-Vese or L2S. It is observed that for each dataset, DL2S demonstrates significantly

TABLE A.1:
Quantitative comparison of the three methods

DL2S		*Chan-Vese* [89]		*L2S* [29]	
Manual	*Auto*	*Manual*	*Auto*	*Manual*	*Auto*
0.93±0.02	0.92±0.04	0.91± 0.07	0.86±0.11	0.89±0.09	0.55±0.17
0.90±0.04	**0.90±0.07**	0.88± 0.05	0.88±0.12	**0.90±0.06**	0.88±0.12
0.85±0.08	**0.86±0.08**	0.80± 0.08	0.85±0.11	0.85±0.12	0.84±0.09
0.80±0.10	**0.83±0.06**	0.69± 0.21	0.73±0.12	0.70±0.19	0.60±0.14
0.76±0.16	**0.76±0.10**	0.75± 0.14	0.72±0.11	0.72±0.16	0.62±0.13

better performance than L2S or CV. Dl2S achieves highest improvement in performance of above 65% in one dataset and 42% in an in-vivo dataset. On average, we observe increase in segmentation accuracy by more thab 12% for the all the datasets.

The mean Dice coefficient for each of the dataset is provided in the table. Results obtained using DL2S remain significantly consistent in compariosn to L2S ans Chan-vese, for both manual and automatic initialization methods and for all the five different datasets.

A.2 Discussion

We have proposed a novel segmentation method which combines the idea of dictionary learning and region based variational segmentation algorithm in presence of significant clutter and heterogeneous intensity. Furthermore, DL2S outperforms the state of the art in terms of contour initialization and demonstrates accurate segmentation in cluttered images without the use of explicit shape prior. The results presented here show significant improvement in segmentation accuracy using basis functions that are computed from the data in comparison to using a fixed number of basis functions.

Appendix B

Extensions of Tubularity Flow Field

Tubularity Flow Field is designed specifically for neuron segmentation, both in 2D and 3D. However, since the basic formulation of TuFF (see (TuFF formula)) can be extended for segmenting any vascular structure, and is not particularly restricted to neurons only. Segmenting vascular structures is a salient problem in the biomedical imaging literature. Sample applications include, but are not limited to, segmenting arteries and veins from magnetic resonance angiography (MRA) images of different organs such as brain, liver, retina etc.

For neuron segmentation, the major hurdle was to accommodate the complicated filamentous architectures, along with the ability to handle structure gaps due to heterogeneous florescence. However, it should be noted that our neuron segmentation example is restricted to single cell analysis only, and the attraction force term in TuFF may produce undesirable effects when multiple filamentous structures are present. Therefore, to generalize TuFF for applications other than neuron tracing, we need replace the attraction force term with other means to handle filament bifurcations and intensity attenuation.

Also, in applications where the filament edges are prominent (i.e. if the gradient magnitude at the boundaries are significant), segmentation accuracy can be further improved by incorporating a edge based energy in TuFF energy equation. This results in

better adherence to object edges and reduces the leakage phenomenon. The modifications suggested in this appendix are focused at the two above mentioned points:

- A robust method to enhance heterogeneously illuminated filaments, and other structure patterns such as sharp bends and bifurcations.
- A solution to incorporate edge information in the TuFF energy functional to avoid leakage in low contrast images.

B.1 Vessel Contrast Enhancement With Local Directional Evidence

Previously, we have discussed the vessel enhancement method due to Frangi [4], which performs eigen analysis of the multiscale hessian matrix of the image to identify and enhance filamentous objects. However, this popular vessel enhancing algorithm suffers from some deficiencies. First, intensity variation across the vessels compromises the enhancement result. This is primarily because the algorithm is inherently local and does not utilize neighboring vessel evidence. Furthermore, the method of [4] assumes that the vessel filaments can be approximated as an ellipsoid, an assumption which is violated at bifurcation and vessel bends (see Fig. B.1). As a result, the output of this vesselness filter sometimes requires an additional postprocessing such as vessel diffusion [31], tensor voting [47] etc.

B.1.1 Vessel detection with oriented filters

Vessel detection problem can be posed as finding the maximum response when the image is convolved with a suitable template (see Fig. B.2(b-c)) at multiple orientations. Freeman and Adelson [118] introduced a class of filters, namely steerable filters, which can be

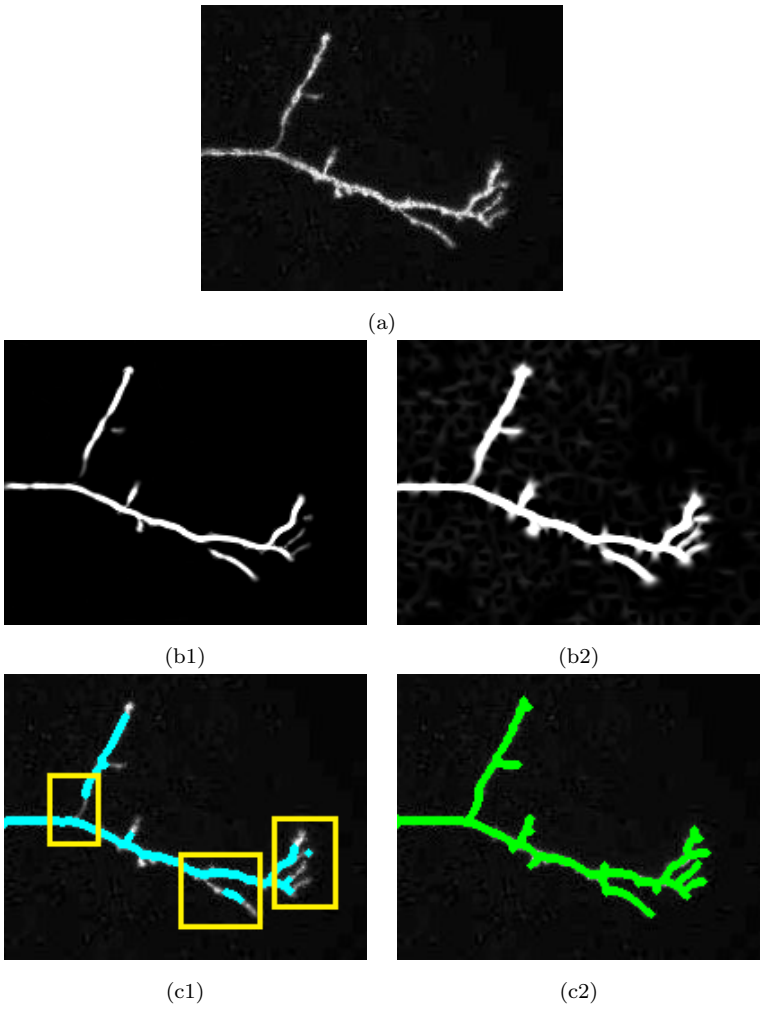


FIGURE B.1: (a) A confocal microscopy image of a dendrite is shown in the first row. (b1) Shows the results of vessel enhancement due to Frangi's filter [4] and (b2) shows the enhancement via LDE. (c1) The segmented centerline using [4] is displayed in cyan and (c2) the tracing via LDE is shown in green. Segmentation errors using [4] are highlighted by the yellow rectangles.

rotated by taking a linear combination of a few basis filters. For ridge detection, steerable filters based on second order derivatives of the gaussian function are popular due to computational simplicity. Jacob and Unser [32] showed that efficient ridge templates can be computed using directional gaussian second derivatives. The hessian $H_\sigma(p)$ of a 2-d image $f(p)$, convolved with a gaussian kernel $g(p;\sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2+y^2}{2\sigma^2}}$ at a position $p = (x, y)$ is given by

$$H_\sigma(p) = \begin{pmatrix} g_{xx}(p) & g_{xy}(p) \\ g_{xy}(p) & g_{yy}(p) \end{pmatrix} * f(p) \tag{B.1}$$

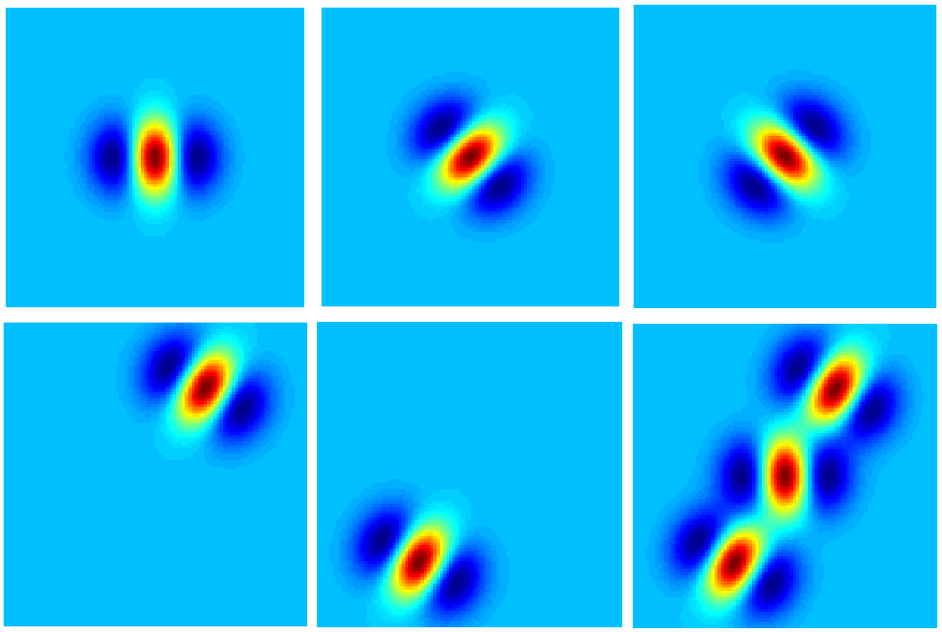


FIGURE B.2: The top row shows oriented vessel detector kernels r_d for $\theta = 0, \frac{\pi}{4}, -\frac{\pi}{4}$ from left to right. The bottom row shows from left to right: forward evidence kernel r_f, backward evidence kernel r_b and the superimposed LDE kernels for $\theta = 0$ and $\psi_1 = \psi_2 = \frac{\pi}{6}$. We choose the offset to an exaggerated value $d = 4\sigma$ for improved visual clarity.

The second derivative of the smoothed image in a direction $\mathbf{u}_\theta = (\cos\theta, \sin\theta)^T$ is obtained as $\mathcal{R}_d(p, \theta; \sigma) = \mathbf{u}_\theta^T H_\sigma(x, y)\mathbf{u}_\theta$. Simplifying this, we obtain the directional vessel response as follows:

$$\mathcal{R}_d(p, \theta; \sigma) = r_d(p, \theta; \sigma) * f(p) \tag{B.2}$$

$$r_d(p; \theta, \sigma) = g_{xx}\cos^2\theta + g_{yy}\sin^2\theta + g_{xy}\sin 2\theta \tag{B.3}$$

The vesselness score at scale σ is computed as

$$\mathcal{R}_d^*(p; \sigma) = \max_\theta \mathcal{R}_d(p, \theta; \sigma) \tag{B.4}$$

Here $r_d(p; \sigma, \theta)$ denotes a local vessel detector template, oriented at an angle $\pi/2 + \theta$. A higher value of the inner product of the shifted and rotated version of the template $r_d(p, \theta; \sigma)$ at each image pixel provides evidence for presence of a vascular object orientated at an angle θ and scale σ. A set of three detector kernels are shown in Fig. B.2, top row.

Despite the property of steerability of the local detection template r_d, such detectors are prone to local intensity variation across the vascular structures. Moreover, the designed steerable kernel in (B.3) is suited for detecting homogeneous vessels and is incapable of handling complicated structures such as vessel bifurcations.

The motivation for this work comes from the fact that the widely used vessel enhancing filters [4, 30, 119, 120] are less adept at detecting vascular regions of low intensity and bifurcation points. Our proposed method, called Local Directional Evidence (LDE), uses a set of oriented filters to determine local evidence of vessels. This set of oriented filters, in addition to the local detector filter shown in (B.4) is used to enhance low contrast vascular structures with complicated morphology.

B.1.2 Vessel enhancement with local directional evidence (LDE)

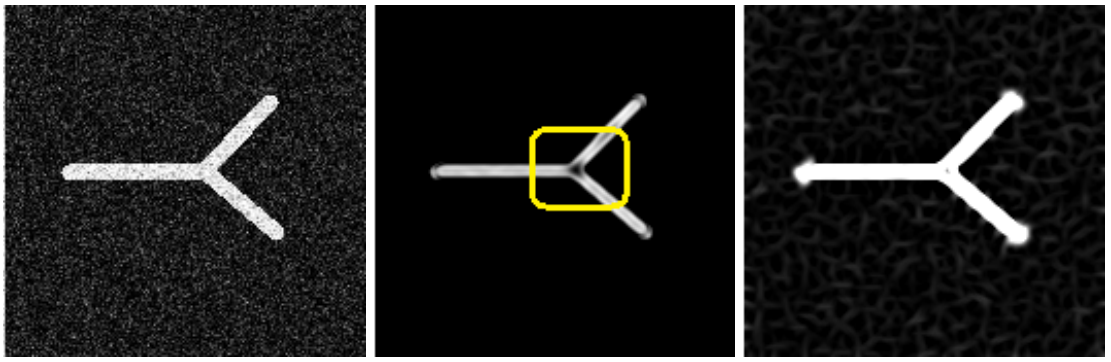
As mentioned before, a major defect with current vessel enhancement methods is the inability to tackle local intensity variations and complex morphology. This creates further artifacts during segmentation, where multiple fragments are created which again require further connectivity analysis [3, 21]. We hypothesize that one possible way to handle this problem is to boost the local detector response (B.3) with evidence of a vessel in a given local neighborhood.

The local directional evidence is provided by convolving the image with another set of oriented filters– forward filters $r_f(p;\sigma,\psi_1)$ and backward filters $r_b(p;\sigma,\psi_2)$. We compute the set of local evidence filters in the following manner:

$$r_f(p;\sigma,\psi_1) = r_d(x + d\cos(\theta+\psi_1), y + d\sin(\theta+\psi_1)) \tag{B.5}$$

$$r_b(p;\sigma,\psi_2) = r_d(x - d\cos(\theta+\psi_2), y - d\sin(\theta+\psi_2)) \tag{B.6}$$

Here $\theta - \alpha \leq \psi_1, \psi_2 \leq \theta + \alpha$ is a set of orientations used to detect evidence of vessels in a local angular region near the detection point. The offset parameter $d > 0$ controls



(a) (b) (c)

FIGURE B.3: A simulated image of a Y-junction is shown in (a). The enhancement result using [4] is shown in (b) and the response at the junction highlighted by the yellow rectangle. (c) The LDE response.

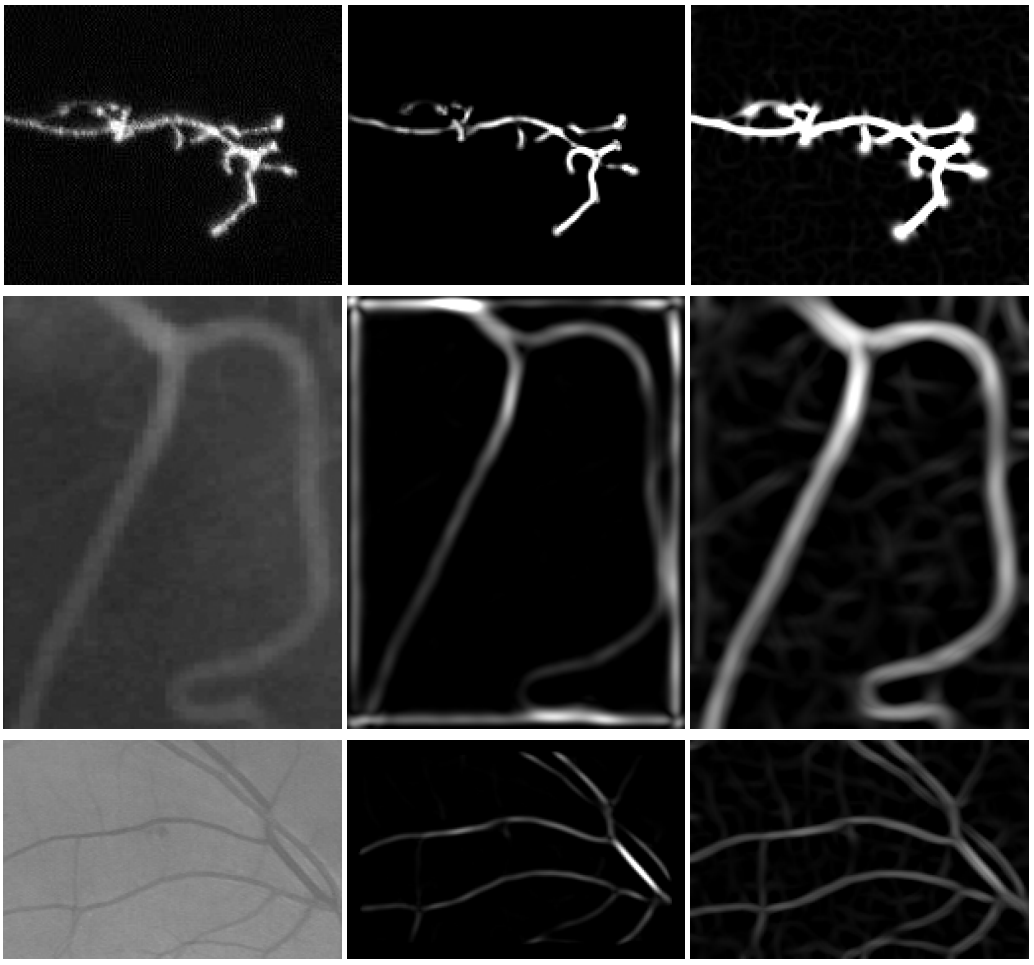
the locality of the evidence filters. While a low value of d does not contribute enough, a significantly larger value may introduce false positives by predicting vessels which do not belong to the same structure. For experimental evaluation, we choose $\alpha = \frac{\pi}{3}$ since abrupt bending in vessels is rare in our applications (neurons and retinal blood vessels). The value of $d \leq \sqrt{2}\sigma$ has been observed to give good balance between localization and accuracy of the evidence filters. A set of evidence kernels is shown in the second row of Fig. B.2 with $\theta = 0$ and $\psi_1 = \psi_2 = \frac{\pi}{6}$.

The response due to the evidence kernels are given by $\mathcal{R}_f(p;\sigma,\psi_1) = r_f(p;\sigma,\psi_1) * f(p)$ and $\mathcal{R}_b(p;\sigma,\psi_2) = r_b(p;\sigma,\psi_2) * f(p)$. The overall vessel enhancement response is calculated in the following manner:

$$\mathcal{R}^*(p;\sigma) = \max_{\theta} \mathcal{R}_d(p) + \mathcal{R}_b^*(p) + \mathcal{R}_f^*(p) \tag{B.7}$$

$$\mathcal{R}_b^*(p) = \max_{\psi_1} \mathcal{R}_b(p) \text{ and } \mathcal{R}_f^*(p) = \max_{\psi_2} \mathcal{R}_f(p)$$

The dependence on the variables θ, ψ_1, ψ_2 is implied. To incorporate vessels with varying thickness in our solution, a multiscale approach is desirable. Over a range of scales



(a) (b) (c)

FIGURE B.4: (a) Images of filamentous objects. (b) Enhancement results due to Frangi and (c) vessel enhancement using LDE

$\mathcal{S} = \{\sigma_{min}, \ldots, \sigma_{max}\}$ scale space vesselness response is calculated as follows:

$$\mathcal{V}(p) = \max_{\sigma \in \mathcal{S}} \mathcal{R}^*(p; \sigma) \tag{B.8}$$

The range of scales is problem specific and is determined from prior biological knowledge about the thickness of the vascular structures.

B.1.3 Discussion of LDE

The most significant contribution of this work is in the design of the directional vessel evidence templates. Unlike the standard ridge detector template shown in the first row

of Fig. B.2, the forward and backward evidence templates in (B.6), together with the detector template (B.3), provide an effective way to approximate the vessels as flexible oriented structures, as opposed to the local rigid structure which results from using the detector template only. A set of vessel templates are shown in Fig. B.2, second row for illustration. The advantage of using the oriented evidence filters is that complex vessel geometry such as sharp bends and bifurcations can also be handled. While the traditional eigenvalue based method proposed by Frangi [4] rejects branch points as candidate vessel point (see Fig. B.3(b)), LDE is able to tackle the bifurcation by virtue of the local evidence filters (see Fig. B.3(c)).

B.2 Edge assisted TuFF

Adding edge based information to the TuFF variational formulation is quite trivial. The effect of the edge term is to make the contour adhere to the object boundaries, thereby restricting the curve to leak across the boundary. The variational form of geodesic active contour [68] is one way to introduce the edge based energy. This is stated as follows:

$$\begin{aligned} \mathcal{E}_{edge} &= \int_{\Omega} g(\mathbf{x}) |\nabla H(\phi)| d\mathbf{x} \\ \text{where} \quad g(\mathbf{x}) &= \frac{1}{1 + |\nabla f(\mathbf{x})|^p} \end{aligned} \tag{B.9}$$

Such edge assistance is particularly useful when the vessels have considerably thicker widths. For narrow vessels, one needs to be conservative while designing the edge attraction term, since it can cause the curve to collapse on to itself. The edge based term replaces the contour regularizer part in (TuFF eq). The scalar variable p is manually adjusted to control the influence of the gradient field. As mentioned in Chapter 4, the effect of this term is to minimize the geodesic length of the curve, as opposed to the euclidean length minimizer proposed in the original formulation.

B.3 Application: automatic crack detection

To demonstrate the wide applicability of TuFF, we conclude this appendix with some results on a non biological dataset. The problem that we aim to solve here has relevance in the field of civil engineering and involves automated structural health monitoring of concrete structures such as bridges, pavements etc.

Appendix C

Derivations of the mathematical results

C.1 Derivation of L2S equation (5.14)

The data term in (5.7) is written as

$$\mathcal{E}_d(\phi, \alpha_i, \beta_i) = \int_\Omega |f(\mathbf{x}) - \sum_i \alpha_i \mathcal{P}_i(\mathbf{x})|^2 m_1(\mathbf{x}) + \int_\Omega |f(\mathbf{x}) - \sum_i \beta_i \mathcal{P}_i(\mathbf{x})|^2 m_2(\mathbf{x}) d\mathbf{x} + \lambda_1 ||\mathbf{a}||^2 + \lambda_2 ||\mathbf{b}||^2$$

We use alternate minimization to compute the locally optimum vectors $\mathbf{a} = (\alpha_0, \ldots, \alpha_{N-1})^T$ and $\mathbf{b} = (\beta_0, \ldots, \beta_{N-1})^T$ by setting $\frac{d\mathcal{E}}{d\alpha_j} = 0$ and $\frac{d\mathcal{E}}{d\alpha_j} = 0$ respectively and solving for α_j and β_j, $\forall i, j \in [0, \ldots, N-1]$. Taking the derivatives we obtain:

$$\frac{d\mathcal{E}}{d\alpha_j} = 2 \int \left(f(\mathbf{x}) - \sum_i \alpha_i \mathcal{P}_i(\mathbf{x}) \right) \mathcal{P}_j(\mathbf{x}) m_1(\mathbf{x}) d\mathbf{x} + 2\lambda_1 \alpha_j = 0$$

$$\text{or,} \quad \int \left(\sum_i \alpha_i \mathcal{P}_i(\mathbf{x}) \right) \mathcal{P}_j(\mathbf{x}) m_1(\mathbf{x}) d\mathbf{x} + \lambda_1 \alpha_j = \int f(\mathbf{x}) \mathcal{P}_j(\mathbf{x}) m_1(\mathbf{x}) d\mathbf{x}$$

$$\text{or,} \quad \sum_i \alpha_i \underbrace{\int \mathcal{P}_i(\mathbf{x}) \mathcal{P}_j(\mathbf{x}) m_1(\mathbf{x}) d\mathbf{x}}_{[K]_{i,j}} + \lambda_1 \alpha_j = \underbrace{\int \mathcal{P}_j(\mathbf{x}) f(\mathbf{x}) m_1(\mathbf{x}) d\mathbf{x}}_{p_j}$$

The above equation can be written in a matrix vector form as

$$[K + \lambda_1 \mathbb{I}]\, \mathbf{a} = \mathbf{p}$$

$$\text{or,} \quad \hat{\mathbf{a}} = [K + \lambda_1 \mathbb{I}]^{-1}\, \mathbf{p}$$

Here $\mathbf{p} = (p_0, \ldots, p_{N-1})^T$. Using, similar arguments, we can also derive $\hat{\mathbf{b}} = [L + \lambda_1 \mathbb{I}]^{-1}\, \mathbf{q}$, where $[L]_{i,j} = \int \mathcal{P}_i(\mathbf{x})\mathcal{P}_j(\mathbf{x})m_2(\mathbf{x})d\mathbf{x}$ and $q_j = \int f(\mathbf{x})\mathcal{P}_j(\mathbf{x})m_2(\mathbf{x})d\mathbf{x}$ respectively.

Once the coefficient vectors are obtained, we proceed to solve for the locally optimum level set function $\phi^* = \nabla_\phi \mathcal{E}_d(\phi, \hat{\mathbf{a}}, \hat{\mathbf{b}})$ using variational calculus. The derivation is trivial, and we refer the reader to [89] for the details.

C.2 Derivation of TuFF equation (6.12)

We provide the derivation of (6.12) for 2D, ie. $\mathbf{x} = (x, y)^T$. The TuFF vector fields are given by $\mathbf{v}_1 = (v_{11}, v_{12})^T$ and $\mathbf{v}_2 = (v_{21}, v_{22})^T$; the dependency on $\mathbf{x}$ implied. The extension to 3-D is simple and follows from this derivation. We can rewrite $\mathcal{E}_{reg}(\phi) = \int E_1(\phi)d\mathbf{x}$, where $E_1(\phi) = \nu_1|\nabla\phi(\mathbf{x})|\delta_\epsilon(\phi)$. Then by calculus of variation, the Gateaux variation of $\mathcal{E}_{reg}$ can be obtained as:

$$\frac{\delta \mathcal{E}_{reg}}{\delta \phi} = \frac{\partial E_1}{\partial \phi} - \frac{\partial}{\partial x}\left(\frac{\partial E_1}{\partial \phi_x}\right) - \frac{\partial}{\partial y}\left(\frac{\partial E_1}{\partial \phi_y}\right) \tag{C.1}$$

Since the proof is already shown in [89], we merely state the result as follows:

$$\frac{\delta \mathcal{E}_{reg}}{\delta \phi} = -\nu_1 \text{div}\left(\frac{\nabla\phi}{|\nabla\phi|}\right)\delta_\epsilon(\phi) \tag{C.2}$$

Similarly, we can write the evolution energy as $\mathcal{E}_{evolve}(\phi) = \int E_2(\phi)d\mathbf{x}$. This can be expanded as $E_2(\phi) = A_1(\phi) + A_2(\phi)$, where $A_j(\phi) = \alpha_j \langle \mathbf{v}_j, \frac{\nabla\phi}{|\nabla\phi|}\rangle^2 H_\epsilon(\phi)$. The dependency of α, ϕ and $\mathbf{v}_j$ on $\mathbf{x}$ is implied, and hence not mentioned explicitly.

We can further decompose A_1 as

$$A_1(\phi) = -\alpha_1 \frac{(v_{11}\phi_x + v_{12}\phi_y)^2}{\phi_x^2 + \phi_y^2} H_\epsilon(\phi)$$

Let us denote $\beta_j = \langle \mathbf{v}_j, \mathbf{n} \rangle$, where the unit normal vector $\mathbf{n} = \frac{\nabla\phi}{|\nabla\phi|}$. Therefore, we can write $A_1(\phi) = -\alpha_1\beta_1^2 H_\epsilon(\phi)$.

As earlier, we compute the Gateaux derivative as follows:

$$\frac{\partial A_1}{\partial \phi} = -\alpha_1\beta_1^2 \delta_\epsilon(\phi) \tag{C.3}$$

Also, by simple algebraic manipulation, we obtain

$$\frac{\partial A_1}{\partial \phi_x} = -2\left[\frac{\alpha_1\beta_1}{|\nabla\phi|}v_{11} - \alpha_1\left(\frac{\beta_1}{|\nabla\phi|}\right)^2 \phi_x\right] H_\epsilon(\phi)$$
$$\frac{\partial A_1}{\partial \phi_y} = -2\left[\frac{\alpha_1\beta_1}{|\nabla\phi|}v_{12} - \alpha_1\left(\frac{\beta_1}{|\nabla\phi|}\right)^2 \phi_y\right] H_\epsilon(\phi)$$

Therefore, we have

$$\frac{\partial}{\partial x}\left(\frac{\partial A_1}{\partial \phi_x}\right) = -2\left[\frac{\partial}{\partial x}\left(\eta_1 v_{11}\right) - \frac{\partial}{\partial x}\left(\eta_1\beta_1\frac{\phi_x}{|\nabla\phi|}\right)\right] \tag{C.4}$$
$$\frac{\partial}{\partial y}\left(\frac{\partial A_1}{\partial \phi_y}\right) = -2\left[\frac{\partial}{\partial y}\left(\eta_1 v_{12}\right) - \frac{\partial}{\partial y}\left(\eta_1\beta_1\frac{\phi_y}{|\nabla\phi|}\right)\right] \tag{C.5}$$

Where $\eta_j = \frac{\alpha_j\beta_j}{|\nabla\phi|}H_\epsilon(\phi)$. Therefore, by symmetry we compute

$$\frac{\partial}{\partial x}\left(\frac{\partial A_j}{\partial \phi_x}\right) + \frac{\partial}{\partial y}\left(\frac{\partial A_j}{\partial \phi_y}\right) = -2\mathrm{div}\left[\left(\eta_j\right)\left(\mathbf{v}_j - \beta_j\mathbf{n}\right)\right] \tag{C.6}$$

The Gateaux variation of $\mathcal{E}_{evolve}$ can be obtained as:

$$\frac{\delta \mathcal{E}_{evolve}}{\delta \phi} = \frac{\partial E_2}{\partial \phi} - \frac{\partial}{\partial x}\left(\frac{\partial E_2}{\partial \phi_x}\right) - \frac{\partial}{\partial y}\left(\frac{\partial E_2}{\partial \phi_y}\right) \tag{C.7}$$

We now use gradient descent to find the local minima of the functionals. The regularizer force and evolution forces are given by $\mathcal{F}_{reg} = -\frac{\delta \mathcal{E}_{reg}}{\delta \phi}$ and $\mathcal{F}_{evolve} = -\frac{\delta \mathcal{E}_{evolve}}{\delta \phi}$ which leads to the following equations:

$$\mathcal{F}_{reg} = \nu_1 \text{div}\left(\frac{\nabla \phi}{|\nabla \phi|}\right) \tag{C.8}$$

and

$$\mathcal{F}_{evolve} = \sum_{j=1}^{d} \left(\alpha_j \beta_j^2 \delta_\epsilon(\phi) - 2\text{div}\left[\eta_j\left(\mathbf{v}_j - \beta_j \mathbf{n}\right)\right]\right) \tag{C.9}$$

Bibliography

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

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

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

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

[5] H. Peng, "Bioimage informatics: a new area of engineering biology," *Bioinformatics*, vol. 24, no. 17, pp. 1827–1836, 2008.

[6] F. De Chaumont, S. Dallongeville, N. Chenouard, N. Hervé, S. Pop, T. Provoost, V. Meas-Yedid, P. Pankajakshan, T. Lecomte, Y. Le Montagner *et al.*, "Icy: an open bioimage informatics platform for extended reproducible research," *Nature Methods*, vol. 9, no. 7, pp. 690–696, 2012.

[7] J. Schindelin, I. Arganda-Carreras, E. Frise, V. Kaynig, M. Longair, T. Pietzsch, S. Preibisch, C. Rueden, S. Saalfeld, B. Schmid *et al.*, "Fiji: an open-source platform for biological-image analysis," *Nature Methods*, vol. 9, no. 7, pp. 676–682, 2012.

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

[9] J. G. White, E. Southgate, J. N. Thomson, and S. Brenner, "The structure of the nervous system of the nematode caenorhabditis elegans," *Philosophical Transactions of the Royal Society of London. B, Biological Sciences*, vol. 314, no. 1165, pp. 1–340, 1986.

[10] P. V. Belichenko, A. Oldfors, B. Hagberg, and A. Dahlström, "Rett syndrome: 3-d confocal microscopy of cortical pyramidal dendrites and afferents," *Neuroreport*, vol. 5, no. 12, pp. 1509–1513, 1994.

[11] C. Koch and I. Segev, "The role of single neurons in information processing," *Nature Neuroscience*, vol. 3, pp. 1171–1177, 2000.

[12] E. A. Daubert, D. S. Heffron, J. W. Mandell, and B. G. Condron, "Serotonergic dystrophy induced by excess serotonin," *Molecular and Cellular Neuroscience*, vol. 44, no. 3, pp. 297–306, 2010.

[13] J. Chen and B. G. Condron, "Branch architecture of the fly larval abdominal serotonergic neurons," *Developmental biology*, vol. 320, no. 1, pp. 30–38, 2008.

[14] H. Cuntz, F. Forstner, J. Haag, and A. Borst, "The morphological identity of insect dendrites," *PLoS Computational Biology*, vol. 4, no. 12, p. e1000251, 2008.

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

[16] S. Nanda, M. Allaham, M. Bergamino, S. Polavaram, R. Armañanzas, G. Ascoli, and R. Parekh, "Doubling up on the fly: Neuromorpho. org meets big data," *Neuroinformatics*, vol. 1, no. 13, pp. 127–129, 2015.

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

[18] M. Oberlaender, R. M. Bruno, B. Sakmann, and P. J. Broser, "Transmitted light brightfield mosaic microscopy for three-dimensional tracing of single neuron morphology," *Journal of biomedical optics*, vol. 12, no. 6, pp. 064 029–064 029, 2007.

[19] A. Santamaría-Pang, C. Colbert, P. Saggau, and I. A. Kakadiaris, "Automatic centerline extraction of irregular tubular structures using probability volumes from multiphoton imaging," in *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2007*. Springer, 2007, pp. 486–494.

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

[21] S. Mukherjee, S. Basu, B. Condron, and S. T. Acton, "Tree2tree2: Neuron tracing in 3d," in *IEEE Intl. Symp. on Biomedical Imaging (ISBI)*, 2012, pp. 448–451.

[22] S. Mukherjee, B. Condron, and S. T. Acton, "Tubularity flow field: A technique for automatic neuron segmentation," *IEEE Trans. Image Process.*, vol. 24, no. 1, pp. 374–389, 2015.

[23] Q. Li, Z. Deng, Y. Zhang, X. Zhou, U. Nagerl, and S. Wong, "A global spatial similarity optimization scheme to track large numbers of dendritic spines in time-lapse confocal microscopy," *IEEE Trans. Med. Imag.*, vol. 30, no. 3, pp. 632–641, March 2011.

[24] Q. Li and Z. Deng, "A surface-based 3-d dendritic spine detection approach from confocal microscopy images," *IEEE Trans. Image Process.*, vol. 21, no. 3, pp. 1223–1230, March 2012.

[25] C. Becker, C. Christoudias, and P. Fua, "Domain adaptation for microscopy imaging," *IEEE Trans. Med. Imag.*, vol. PP, no. 99, pp. 1–1, 2014.

[26] H. Nguyen and Q. Ji, "Shape-driven three-dimensional watersnake segmentation of biological membranes in electron tomography," *IEEE Trans. Med. Imag.*, vol. 27, no. 5, pp. 616–628, 2008.

[27] A. Vaccari, K. Gamage, S. Nachum, B. Condron, C. Deppmann, and S. T. Acton, "Assessment of wallerian degeneration by automated image analysis," in *Asilomar Conf. on Signals, Systems and Computers (ASILOMAR)*. IEEE, 2012, pp. 1583–1587.

[28] S. Mukherjee and S. T. Acton, "Vector field convolution medialness applied to neuron tracing," in *IEEE Intl. Conf. on Image Processing (ICIP)*, 2013, pp. 665–669.

[29] S. Mukherjee and S. Acton, "Region based segmentation in presence of intensity inhomogeneity using legendre polynomials," *IEEE Signal Processing Letters*, vol. 22, no. 3, pp. 298–302, March 2015.

[30] Y. S. *et al.*, "Three-dimensional multi-scale line filter for segmentation and visualization of curvilinear structures in medical images," *Medical Image Analysis*, vol. 2, no. 2, pp. 143–168, 1998.

[31] R. M. *et al.*, "Vessel enhancing diffusion: A scale space representation of vessel structures," *Medical Image Analysis*, vol. 10, no. 6, pp. 815–825, 2006.

[32] M. Jacob and M. Unser, "Design of steerable filters for feature detection using canny-like criteria," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 26, no. 8, pp. 1007–1019, 2004.

[33] A. C. Bovik, *Handbook of Image and Video processing.* Academic press, 2010.

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

[35] A. X. Falcão, J. K. Udupa, S. Samarasekera, S. Sharma, B. E. Hirsch, and R. d. A. Lotufo, "User-steered image segmentation paradigms: Live wire and live lane," *Graphical Models and Image Processing*, vol. 60, no. 4, pp. 233–260, 1998.

[36] M. D. Abràmoff, P. J. Magalhães, and S. J. Ram, "Image processing with imagej," *Biophotonics International*, vol. 11, no. 7, pp. 36–42, 2004.

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

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

[39] H. Peng, Z. Ruan, F. Long, J. H. Simpson, and E. W. Myers, "V3d enables real-time 3d visualization and quantitative analysis of large-scale biological image data sets," *Nature Biotechnology*, vol. 28, no. 4, pp. 348–353, 2010.

[40] J. Xie, T. Zhao, T. Lee, E. W. Myers, and H. Peng, "Anisotropic path searching for automatic neuron reconstruction," *Medical Image Analysis*, vol. 15, no. 5, pp. 680–689, 2011.

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

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

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

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

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

[46] H. Cai, X. Xu, J. Lu, J. Lichtman, S. Yung, and S. T. Wong, "Shape-constrained repulsive snake method to segment and track neurons in 3d microscopy images," in *IEEE Intl. Symp. on Biomedical Imaging (ISBI)*, 2006, pp. 538–541.

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

[48] Y. Y. Boykov and M.-P. Jolly, "Interactive graph cuts for optimal boundary & region segmentation of objects in nd images," in *IEEE Intl. Conf. on Computer Vision (ICCV)*, vol. 1. IEEE, 2001, pp. 105–112.

[49] K. A. Al-Kofahi, A. Can, S. Lasek, D. H. Szarowski, N. Dowell-Mesfin, W. Shain, J. N. Turner, and B. Roysam, "Median-based robust algorithms for tracing neurons from noisy confocal microscope images," *IEEE Trans. Info. Tech. in Biomed.*, vol. 7, no. 4, pp. 302–317, 2003.

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

[51] R. Srinivasan, X. Zhou, E. Miller, J. Lu, J. Litchman, and S. T. Wong, "Automated axon tracking of 3d confocal laser scanning microscopy images using guided probabilistic region merging," *Neuroinformatics*, vol. 5, no. 3, pp. 189–203, 2007.

[52] J. Wang, X. Zhou, J. Lu, J. Lichtman, S.-F. Chang, and S. T. Wong, "Dynamic local tracing for 3d axon curvilinear structure detection from microscopic image stack," in *IEEE Intl. Symp. on Biomedical Imaging (ISBI)*. IEEE, 2007, pp. 81–84.

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

[54] Y. Wang, A. Narayanaswamy, and B. Roysam, "Novel 4-d open-curve active contour and curve completion approach for automated tree structure extraction," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2011, pp. 1105–1112.

[55] D. Lesage, E. D. Angelini, I. Bloch, and G. Funka-Lea, "A review of 3d vessel lumen segmentation techniques: Models, features and extraction schemes," *Medical Image Analysis*, vol. 13, no. 6, pp. 819–845, 2009.

[56] L. M. Lorigo, O. D. Faugeras, W. E. L. Grimson, R. Keriven, R. Kikinis, A. Nabavi, and C.-F. Westin, "Curves: Curve evolution for vessel segmentation," *Medical Image Analysis*, vol. 5, no. 3, pp. 195–206, 2001.

[57] A. Gooya, H. Liao, K. Matsumiya, K. Masamune, Y. Masutani, and T. Dohi, "A variational method for geometric regularization of vascular segmentation in medical images," *IEEE Trans. Image Process.*, vol. 17, no. 8, pp. 1295–1312, 2008.

[58] A. Gooya, H. Liao, and I. Sakuma, "Generalization of geometrical flux maximizing flow on riemannian manifolds for improved volumetric blood vessel segmentation," *Computerized Medical Imaging and Graphics*, vol. 36, no. 6, pp. 474–483, 2012.

[59] A. Vasilevskiy and K. Siddiqi, "Flux maximizing geometric flows," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 24, no. 12, pp. 1565–1578, 2002.

[60] Y. Shang, R. Deklerck, E. Nyssen, A. Markova, J. de Mey, X. Yang, and K. Sun, "Vascular active contour for vessel tree segmentation," *IEEE Transactions on Biomedical Engineering*, vol. 58, no. 4, pp. 1023–1032, 2011.

[61] B. N. Saha and N. Ray, "Image thresholding by variational minimax optimization," *Pattern Recognition*, vol. 42, no. 5, pp. 843–856, 2009.

[62] S. Osher and J. A. Sethian, "Fronts propagating with curvature-dependent speed: algorithms based on hamilton-jacobi formulations," *Journal of Computational Physics*, vol. 79, no. 1, pp. 12–49, 1988.

[63] V. Caselles, R. Kimmel, and G. Sapiro, "Geodesic active contours," *International Journal of Computer Vision*, vol. 22, no. 1, pp. 61–79, 1997.

[64] M. A. Grayson, "The heat equation shrinks embedded plane curves to round points," *Journal of Differential Geometry*, vol. 26, no. 2, pp. 285–314, 1987.

[65] V. Caselles, R. Kimmel, G. Sapiro, and C. Sbert, "Minimal surfaces based object segmentation," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 19, no. 4, pp. 394–398, 1997.

[66] A. Tannenbaum, G. Sapiro, and P. J. Olver, "Invariant geometric evolutions of surfaces and volumetric smoothing," *SIAM Journal on Applied Mathematics*, vol. 57, no. 1, pp. 176–194, 1997.

[67] R. Malladi, J. A. Sethian, and B. C. Vemuri, "Shape modeling with front propagation: A level set approach," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 17, no. 2, pp. 158–175, 1995.

[68] V. Caselles, R. Kimmel, and G. Sapiro, "Geodesic active contours," *International Journal of Computer Vision*, vol. 22, no. 1, pp. 61–79, 1997.

[69] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active contour models," *International Journal of Computer Vision*, vol. 1, no. 4, pp. 321–331, 1988.

[70] B. Li and S. T. Acton, "Active contour external force using vector field convolution for image segmentation," *IEEE Trans. Image Process.*, vol. 16, no. 8, pp. 2096–2106, 2007.

[71] C. Xu and J. L. Prince, "Snakes, shapes, and gradient vector flow," *IEEE Trans. Image Process.*, vol. 7, no. 3, pp. 359–369, 1998.

[72] B. Li and S. T. Acton, "Automatic active model initialization via poisson inverse gradient," *IEEE Trans. Image Process.*, vol. 17, no. 8, pp. 1406–1420, 2008.

[73] M. Jacob, T. Blu, and M. Unser, "Efficient energies and algorithms for parametric snakes," *IEEE Trans. Image Process.*, vol. 13, no. 9, pp. 1231–1244, 2004.

[74] P. Brigger, J. Hoeg, and M. Unser, "B-spline snakes: a flexible tool for parametric contour detection," *IEEE Trans. Image Process.*, vol. 9, no. 9, pp. 1484–1496, 2000.

[75] A. K. Mishra, P. W. Fieguth, and D. A. Clausi, "Decoupled active contour (dac) for boundary detection," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 33, no. 2, pp. 310–324, 2011.

[76] D. P. Mukherjee, N. Ray, and S. T. Acton, "Level set analysis for leukocyte detection and tracking," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 562–572, 2004.

[77] N. Ray, S. T. Acton, T. Altes, E. E. De Lange, and J. R. Brookeman, "Merging parametric active contours within homogeneous image regions for mri-based lung segmentation," *IEEE Trans. Med. Imag.*, vol. 22, no. 2, pp. 189–199, 2003.

[78] N. Ray, S. T. Acton, and K. Ley, "Tracking leukocytes in vivo with shape and size constrained active contours," *IEEE Trans. Med. Imag.*, vol. 21, no. 10, pp. 1222–1235, 2002.

[79] L. Zhu, Y. Gao, V. Appia, A. Yezzi, C. Arepalli, T. Faber, A. Stillman, and A. Tannenbaum, "A complete system for automatic extraction of left ventricular myocardium from ct images using shape segmentation and contour evolution," *IEEE Trans. Image Process.*, vol. 23, no. 3, pp. 1340–1351, 2014.

[80] C. Li, C. Xu, C. Gui, and M. D. Fox, "Level set evolution without re-initialization: a new variational formulation," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, vol. 1. IEEE, 2005, pp. 430–436.

[81] ——, "Distance regularized level set evolution and its application to image segmentation," *IEEE Trans. Image Process.*, vol. 19, no. 12, pp. 3243–3254, 2010.

[82] G. Papandreou and P. Maragos, "Multigrid geometric active contour models," *IEEE Trans. Image Process.*, vol. 16, no. 1, pp. 229–240, 2007.

[83] R. Goldenberg, R. Kimmel, E. Rivlin, and M. Rudzsky, "Fast geodesic active contours," *IEEE Trans. Image Process.*, vol. 10, no. 10, pp. 1467–1475, 2001.

[84] J. A. Sethian, "Fast marching methods," *SIAM review*, vol. 41, no. 2, pp. 199–235, 1999.

[85] Y. Shi and W. C. Karl, "A real-time algorithm for the approximation of level-set-based curve evolution," *IEEE Trans. Image Process.*, vol. 17, no. 5, pp. 645–656, 2008.

[86] D. Mumford and J. Shah, "Optimal approximations by piecewise smooth functions and associated variational problems," *Communications on Pure and Applied Mathematics*, vol. 42, no. 5, pp. 577–685, 1989.

[87] J. L. Troutman, *Variational calculus and optimal control: optimization with elementary convexity*. Springer-Verlag New York, Inc., 1995.

[88] H.-K. Zhao, T. Chan, B. Merriman, and S. Osher, "A variational level set approach to multiphase motion," *Journal of Computational Physics*, vol. 127, no. 1, pp. 179–195, 1996.

[89] T. F. Chan and L. A. Vese, "Active contours without edges," *IEEE Trans. Image Process.*, vol. 10, no. 2, pp. 266–277, 2001.

[90] O. Bernard, D. Friboulet, P. Thévenaz, and M. Unser, "Variational b-spline level-set: a linear filtering approach for fast deformable model evolution," *IEEE Trans. Image Process.*, vol. 18, no. 6, pp. 1179–1191, 2009.

[91] S. Lankton and A. Tannenbaum, "Localizing region-based active contours," *IEEE Trans. Image Process.*, vol. 17, no. 11, pp. 2029–2039, 2008.

[92] T. Chan and W. Zhu, "Level set based shape prior segmentation," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, vol. 2, 2005, pp. 1164–1170.

[93] D. Cremers, M. Rousson, and R. Deriche, "A review of statistical approaches to level set segmentation: integrating color, texture, motion and shape," *International Journal of Computer Vision*, vol. 72, no. 2, pp. 195–215, 2007.

[94] A. Foulonneau, P. Charbonnier, and F. Heitz, "Affine-invariant geometric shape priors for region-based active contours," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 28, no. 8, pp. 1352–1357, 2006.

[95] D. Nain, A. Yezzi, and G. Turk, "Vessel segmentation using a shape driven flow," in *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2004*. Springer, 2004, pp. 51–59.

[96] L. A. Vese and T. F. Chan, "A multiphase level set framework for image segmentation using the mumford and shah model," *International Journal of Computer Vision*, vol. 50, no. 3, pp. 271–293, 2002.

[97] S. Basu, A. Aksel, B. Condron, and S. T. Acton, "Tree2tree: Neuron segmentation for generation of neuronal morphology," in *IEEE Intl. Symp. on Biomedical Imaging (ISBI)*, 2010, pp. 548–551.

[98] C. Li, C.-Y. Kao, J. C. Gore, and Z. Ding, "Minimization of region-scalable fitting energy for image segmentation," *IEEE Trans. Image Process.*, vol. 17, no. 10, pp. 1940–1949, 2008.

[99] J. Kim, J. W. Fisher, A. Yezzi, M. Çetin, and A. S. Willsky, "A nonparametric statistical method for image segmentation using information theory and curve evolution," *IEEE Trans. Image Process.*, vol. 14, no. 10, pp. 1486–1502, 2005.

[100] H. Feng, D. A. Castanon, and W. C. Karl, "Tomographic reconstruction using curve evolution," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, 2000, pp. 361–366.

[101] X. Du and T. D. Bui, "A new model for image segmentation," *IEEE Signal Processing Letters*, vol. 15, pp. 182–185, 2008.

[102] K. D. Fritscher, A. Grünerbl, and R. Schubert, "3d image segmentation using combined shape-intensity prior models," *International Journal of Computer Assisted Radiology and Surgery*, vol. 1, no. 6, pp. 341–350, 2007.

[103] X. Huang and D. N. Metaxas, "Metamorphs: deformable shape and appearance models," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 30, no. 8, pp. 1444–1459, 2008.

[104] N. Barth, "The gramian and k-volume in n-space: some classical results in linear algebra," *J Young Investig*, vol. 2, 1999.

[105] T. Dietenbeck, M. Alessandrini, D. Friboulet, and O. Bernard, "Creaseg: a free software for the evaluation of image segmentation algorithms based on level-set," in *IEEE Intl. Conf. on Image Processing (ICIP)*, 2010, pp. 665–668.

[106] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *IEEE Intl. Conf. on Computer Vision (ICCV)*, vol. 2, July 2001, pp. 416–423.

[107] A. Achuthan, M. Rajeswari, D. Ramachandram, M. E. Aziz, and I. L. Shuaib, "Wavelet energy-guided level set-based active contour: A segmentation method to segment highly similar regions," *Computers in biology and medicine*, vol. 40, no. 7, pp. 608–620, 2010.

[108] R. Sarkar, S. Mukherjee, and S. Acton, "Dictionary learning level sets (*accepted*)," *IEEE Signal Processing Letters*, 2015.

[109] M. Aharon, M. Elad, and A. Bruckstein, "-svd: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Trans. Signal Process.*, vol. 54, no. 11, pp. 4311–4322, 2006.

[110] N. Otsu, "A threshold selection method from gray-level histograms," *Automatica*, vol. 11, no. 285-296, pp. 23–27, 1975.

[111] R. L. Graham and F. Frances Yao, "Finding the convex hull of a simple polygon," *Journal of Algorithms*, vol. 4, no. 4, pp. 324–331, 1983.

[112] S. T. Acton, "Fast algorithms for area morphology," *Digital Signal Processing*, vol. 11, no. 3, pp. 187–203, 2001.

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

[114] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 31, no. 2, pp. 210–227, 2009.

[115] K. Kreutz-Delgado, J. F. Murray, B. D. Rao, K. Engan, T.-W. Lee, and T. J. Sejnowski, "Dictionary learning algorithms for sparse representation," *Neural Computation*, vol. 15, no. 2, pp. 349–396, 2003.

[116] M. Elad and M. Aharon, "Image denoising via sparse and redundant representations over learned dictionaries," *IEEE Trans. Image Process.*, vol. 15, no. 12, pp. 3736–3745, 2006.

[117] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman, "Non-local sparse models for image restoration," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2009, pp. 2272–2279.

[118] W. T. Freeman and E. H. Adelson, "The design and use of steerable filters," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 13, no. 9, pp. 891–906, 1991.

[119] M. Wilkinson and M. Westenberg, "Shape preserving filament enhancement filtering," in *MICCAI*. Springer, 2001, pp. 770–777.

[120] K. K. *et al.*, "Model-based detection of tubular structures in 3d images," *Computer Vision and Image Understanding*, vol. 80, no. 2, pp. 130–171, 2000.