SEGMENTATION OF NEURONS FROM ELECTRON MICROSCOPY IMAGES

by

Elizabeth Jurrus

A dissertation submitted to the faculty of
The University of Utah
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in

Computing

School of Computing

The University of Utah

August 2011

The Graduate School
THE UNIVERSITY OF UTAH

STATEMENT OF DISSERTATION APPROVAL

The dissertation of **Elizabeth Jurrus**
has been approved by the following supervisory committee members:

Tolga Tasdizen , Chair	**05/10/2011** Date Approved
Ross T. Whitaker , Member	**05/10/2011** Date Approved
Thomas C. Henderson , Member	**05/10/2011** Date Approved
Hal Daumé III , Member	**05/10/2011** Date Approved
Erik M. Jorgensen , Member	**05/10/2011** Date Approved
Bryan W. Jones , Member	**05/10/2011** Date Approved

and by **Al Davis**,

Chair of the Department of **Computer Science**

Charles A. Wight,
Dean of The Graduate School

ABSTRACT

Neuroscientists are developing new imaging techniques and generating large volumes of data in an effort to understand the complex structure of the nervous system. The complexity and size of this data makes human interpretation a labor intensive task. To aid in the analysis, new segmentation techniques for identifying neurons in these feature rich datasets are required. However, the extremely anisotropic resolution of the data makes segmentation and tracking across slices difficult. Furthermore, the thickness of the slices can make the membranes of the neurons hard to identify. Similarly, structures can change significantly from one section to the next due to slice thickness which makes tracking difficult. This thesis presents a complete method for segmenting many neurons at once in two-dimensional (2D) electron microscopy images and reconstructing and visualizing them in three-dimensions (3D). First, we present an advanced method for identifying neuron membranes in 2D, necessary for whole neuron segmentation, using a machine learning approach. The method described uses a series of artificial neural networks (ANNs) in a framework combined with a feature vector that is composed of image and context intensities sampled over a stencil neighborhood. Several ANNs are applied in series allowing each ANN to use the classification context provided by the previous network to improve detection accuracy. To improve the membrane detection, we use information from a nonlinear alignment of sequential learned membrane images in a final ANN that improves membrane detection in each section. The final output, the detected membranes, are used to obtain 2D segmentations of all the neurons in an image. We also present a method that constructs 3D neuron representations by formulating the problem of finding paths through sets of sections as an optimal path computation, which applies a cost function to the identification of a cell from one section to the next and solves this optimization problem using Dijkstras algorithm. This basic formulation accounts for variability or inconsistencies between sections and prioritizes cells based on the evidence of their connectivity. Finally, we present a tool that combines these techniques with a visual user interface that enables users to quickly segment whole neurons in large volumes.

To Sam.

CONTENTS

LIST OF FIGURES

ACKNOWLEDGEMENTS

Many thanks go out to my committee. In particular, I would like to thank Tolga Tasdizen for his patience and confidence in my potential. I am also grateful for Ross Whitaker's early years as a co-advisor and reminding me on a consistent basis that there is, indeed, a bug in my code. I also want to thank my external committee members, Erik Jorgensen and Bryan Jones, for going above and beyond their duties as advisors to explain basic neurobiolgy and microscopy to me so that I was able to easily understand the problem and the challenges we were working together to solve. Lastly, I appreciate the time Hal Duamé and Tom Henderson took to engage in disucssions and provide much needed feedback on the research we were doing and the process I was going through towards this degree.

No graduate school experience would have been complete without the help of the School of computing and the Scientific Computing and Imaging Institute. The staff from the School of Computing have been tremendously helpful in navigating the Ph.D. process and providing the resources necessary to most effectively complete this degree. The Scientific Computing and Imaging Institute was benneficial in providing outstanding staff support, a comfortable work environment, and endless cups of coffee.

I am also extremely grateful for all the ladies nights I had with friends. To Kristi, Miriah, and Betty... Your support and your late night talks made this process seem sane. I look forward to what life will bring all of us as we continue down our academic paths.

For my parents, thank you for your endless support and encouragement. None of this would have been possible without having you both as my inspiration. The care packages from my Mom helped, too.

To sam, my friend, my husband, and my colleague. You are truly my partner in life, my inspriation for ideas and new techniques, as well as my best critic. I have learned from every discussion we have had and every idea you have ever challenged me to defend.

Finally, I would like to acknowledge my funding support. This research was supported by NIH R01 EB005832 (TT). Data acquisition was supported by NIH EY0015128 (RM), EY002576 (RM), NEI Vision Core EY014800 (RM), HHMI (EMJ), and NIH NINDS 5R37NS34307-15 (EMJ).

CHAPTER 1

INTRODUCTION

The goal of this dissertation is to develop tools and algorithms to segment neurons in two-dimensional (2D) images and assist in the reconstruction of whole neurons from a stack of electron microscopy (EM) images into three-dimensional (3D) volumes. The answers to many biological questions depend on a better understanding of cellular ultrastructure, and microscopic imaging is providing new possibilities for exploring these questions. An important problem in neurobiology is deciphering the patterns of neuronal connections that govern neural computation and ultimately behavior. However, relatively little is known about the physical organization and connectivities of neurons at the subcellular level. Thus, EM has emerged as the primary tool for resolving the 3D structure and connectivity of neurons. The resolution and data quality of EM enable scientists to study neurons at the cellular level, exposing synapses and structures that reveal cellular connectivity. The data also present the scientists with an overwhelming amount of data. Automated methods that can segment thousands of cells across thousands of sections are required, since humans cannot possibly study all these images in a reasonable amount of time. However, automated methods are a challenging problem due to the anisotropic nature of the data, the amount of noise present in some of the imaging modalities, and the presence of intracellular structures. As a result, researchers need to develop techniques that can account for these difficulties and provide scientists with new techniques and tools to annotate and extract patterns from the data.

Segmenting whole neurons in this dissertation is performed in two steps. First, we introduce a machine learning method for automatically detecting neuron membranes in EM images. The accuracy and the improvement of this method compared to similar work is demonstrated visually with examples and quantitatively using receiver operator characteristic curves. The detected membranes, combined with a simple segmentation algorithm, such as watersheds, result in whole segmented 2D regions that represent neurons. We introduce a second method that links 2D neuron regions across sections to produce a full

3D neuron reconstruction and rendering. Scientists can then use the 3D model to identify the physical organization of neurons and begin to understand connectivity. Compared to early methods for segmenting neurons and mapping connectivity, which were all performed by hand, the work in this proposal aims at turning what used to take years for a human to perform to what will take only a matter of weeks of mostly computer time.

1.1 Motivation for Neural Circuit Reconstruction in EM Images

While neural circuits are central to the study of the nervous system, relatively little is known about their actual arrangement, including differences in existing neuronal classes, patterns, and connections. The answers to many biological questions depend on a better understanding of cellular ultrastructure. The impact of these findings covers a wide range of disciplines, beginning with a better understanding of retinal degenerative diseases to drug discovery and new electronic devices. Microscopic imaging is providing new possibilities for exploring these questions, making it possible to produce large datasets containing millions of cellular processes. However, because of its size and resolution, challenges exist in the analysis and interpretation of this data.

Medical imaging modalities such as MRI provide three-dimensional (3D) measurements of the brain with resolutions on the order of 1 mm [117]. This resolution provides macroscopic information about brain organization, but does not allow analysis of individual neurons. Scanning confocal [70] and two photon [25] light microscopy have the ability to visualize live specimens, but are limited to 200 nm lateral resolution and 500 nm z resolution, which are insufficient to reconstruct connections of individual neurons. Newer light microscopic methods such as 4Pi, STORM, and PALM [10, 27, 86] promise higher resolution. Unfortunately, 4Pi still cannot resolve closely bundled axons, while STORM and PALM, at present, are 2D methods requiring very long imaging times, making large image acquisition too time consuming. Thus electron microscopy, which provides much higher resolution, remains the primary tool for resolving the 3D structure and connectivity of neurons.

Electron microscopy (EM) is an unique modality for scientists attempting to map the anatomy of individual neurons and their connectivity because it has a resolution that is high enough to identify synaptic contacts and gap junctions, which span 16-20nm [2]. These are important indicators for types of neuron topology and are required for neural circuit reconstruction. Several researchers have undertaken extensive EM imaging projects in order

to create detailed maps of neuronal structure and connectivity [13, 29]. Early work in this area, by White *et al.* [115], includes the complete mapping of the nematode *C. elegans* nervous system. This is a simple organism, containing 302 neurons and just over 6000 synapses, yet it took nearly a decade to identify all the relevant structures and reconstruct the connectivity [91]. In comparison, newer imaging techniques are producing much larger volumes of very complex organisms, with thousands of neurons and millions of synapses [2, 14]. Thus, automating the reconstruction process is of paramount importance.

The ability to reconstruct neural circuitry at ultrastructural resolution is also of substantial clinical importance. For instance, in the retina, degenerative diseases, including pigmentosa and macular degeneration, result from a loss of photoreceptors. Photoreceptor cell stress and death induces subsequent changes in the neural circuitry of the retina resulting in corruption of the surviving retinal cell class circuitry. Ultrastructural examination of the cell identity and circuitry reveals substantial changes to retinal circuitry with implications for vision rescue strategies [48, 50, 49, 67, 65, 66, 76]. These findings in retinal degenerative disease mirror findings in epilepsy where neural circuits also undergo remodeling in presumed response to abnormal electrical activity clinically manifested as seizures. Scientists are interested in examining normal and pathological synaptic connectivities and how neuronal remodeling contributes to neuronal pathophysiology [57, 79, 95]. Examination of synaptic and dendritic spine formation during development provides insight into the adaptivity of neural circuits [22, 93]. Ultrastructural evaluation of multiple canonical volumes of neural tissue is critical to evaluate differences in connectivity between wild type and mutants. The complexity and size of the these datasets, which currently are around 120k×120k×400 pixels and contain over 1000 cellular structures that wind, twist, split, and merge, makes human segmentation of the complex textural information of electron microscopic imagery a difficult task. Moreover, population or screening studies become infeasible since fully manual segmentation and analysis would require multiple years of manual effort per specimen. As a result, better image processing techniques are needed to help with automated segmentation of EM data including identification of neurons and the connections.

Information gained from neural circuit reconstruction also greatly benefits one of the grand challenges by the National Academy of Engineering which is to reverse engineer the brain [103]. The impact from this type of information spans many biological disciplines. Biologists will be able to simulate protein interactions and build new ways to test drugs. Scientists will also be able to study and test solutions for brain disorders using new comput-

ing methods. Finally, patients suffering from neuological disorders or damage would benefit from an electronic device that would replace the lost tissues [8, 35, 60].

To summarize, deciphering the patterns of neuronal connections that govern neural computation and ultimately behavior is an important problem in biology and engineering. New image acquisition methods provide scientists with the data needed to do this analysis. However, image processing techniques are also needed to aid in the discovery of patterns and structures.

1.2 Data Acquisition

Electron microscopes produce a magnified image of a tissue by illuminating the specimen with a particle beam of electrons. There are two general types of EM used for neuron reconstruction, transmission and reflection EM. In transmission EM, the electron beam is transmitted through the specimen forming an image that is magnified by an objective lens. Reflection EM creates an image of a specimen by capturing the electrons reflected from a block of tissue. These two methods are used to acquire the data presented in this thesis that provides scientists with high resolution images that capture the relevant structures. Serial section transmission electron microscopy (ssTEM) is type of transmission EM and offers a relatively wide field of view to identify large sets of cells that may wander significantly as they progress through the sections. However, it has a disadvantage in that the images are captured as a series of fields that require registration in order to form a complete volume. It has an in plane resolution that is high enough for identifying synapses in tissues that have been stained to highlight them. In collecting images through ssTEM, sections are cut from a specimen and suspended so that an electron beam can pass through it creating a projection (Figure 1.1(a)). The projection can be captured on a piece of film and scanned or captured directly as a digital image. Two examples of images captured using this method are shown in Figure 1.2(a). Serial block face scanning electron microscopy (SBFSEM) [23, 24] is a type of reflection EM. This method uses an electron beam to scan the top of a block of tissue, producing an electron backscattering image, shown in Figure 1.1(b). Successive sections are removed and discarded between each scan, cutting perpendicular to the cell membrane, producing a volume of images with very little deformation and no need for section to section registration. In contrast to ssTEM, images from SBFSEM have a lower signal to noise ratio and do not have as large of a field of view. SBFSEM often uses a specimen preparation which highlights extracellular spaces, removing much of the contrast from intracellular

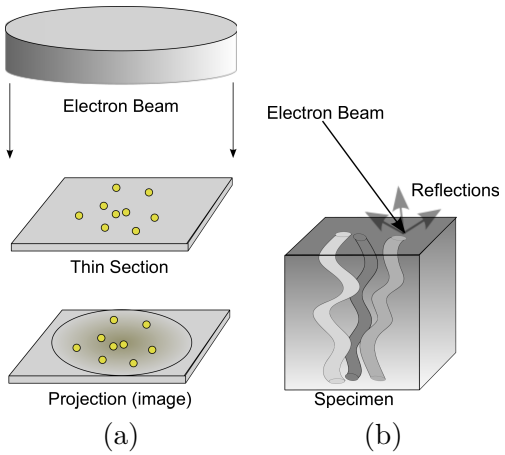
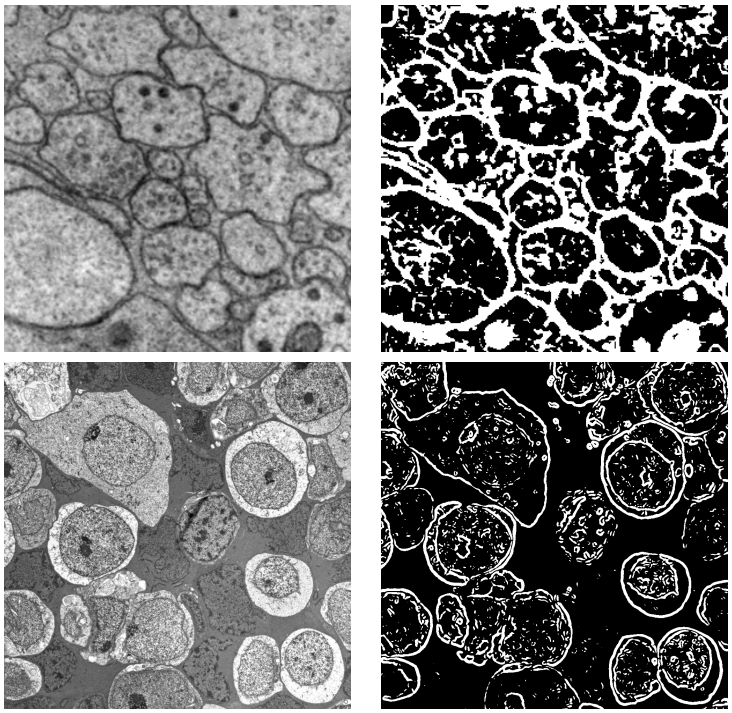


Figure 1.1. Image acquisition diagrams for (a) serial section TEM and (b) serial block face scanning EM.

structures. As a result, it is not possible to identify synapses with this approach but it is useful for extracting structure. If the goal is to identify synaptic connectivity, synapses must be present in the data, because structure does not always determine connectivity [72]. However, A specialized scanning electron microscope equipped with a high precision Gatan 3View ultramicrotome combined with a new novel specimen preparation protocol developed by Deerinck *et al.* [23] that improves the resolution of the imaging several fold, makes it possible to view increased detail of individual cells in the context of their surroundings. An example image using SBFSEM is shown in Section 4.4.2. Imaging techniques not used in this dissertation, such as focused ion beam electron microscopy [56], and the Automatic Tape Collecting Lathe Ultramicrotome [38], are alternative data acquisition technologies. Electron tomography [17, 36, 94] is also an emerging modality for neuron reconstruction because of its ability to capture full 3D image volumes.

1.3 Technical Challenges

Images from EM pose some interesting challenges for data processing. For serial section EM, an important trade off occurs with respect to the section thickness. Thinner sections are preferable from an image analysis point of view because structures are more easily identifiable due to less averaging. However, from an image acquisition point of view, thinner sections from ssTEM are harder to handle and impose a limit on the area of the section that can be cut. For instance, in the rabbit retina, scientists need to study sections with areas as



(a) (b)

Figure 1.2. Example EM images. (a), top, is from the *C. elegans*, bottom is from a rabbit retina. (b) example membrane detection using thresholding after contrast enhancement and anisotropic directional smoothing to enhance membranes (top), and thresholding on the gradient magnitude (bottom). Both methods highlight the membrane boundaries but fail to remove internal structures.

large as $250\mu m$ in diameter to gain a sufficient understanding of neural connectivity patterns. Sections of this size can be reliably cut at $50-90$nm thickness with the current ssTEM technology. This leads to an extremely anisotropic resolution, $2-5$nm in plane compared to $50-90$nm out of plane, and poses two image processing challenges. First, the cell membranes can range from solid dark curves for neurons that run approximately perpendicular to the cutting plane, to grazed gray swaths for others which run more obliquely and suffer more from the averaging effect. Consequently, segmentations of neurons in these 2D images, are difficult given the change in membrane contrast and thickness. Second, due to the large physical separation between sections, shapes and positions of neurons can change significantly between adjacent sections, making it challenging to track important structural differences because they are missing from the data. Finally, to highlight synapses in ssTEM, scientists use stain for EM that also emphasises intracellular structures, such as vesicles

and mitochondria, as well as neuron membranes. The images from SBFSEM are also of anisotropic resolution, with an in plane resolution of 10nm per pixel to $20-70$nm between planes. As a result, cellular processes change considerably between sections and are difficult to segment. Finally, as image acquisition technologies improve larger and larger imaging efforts are taking place, increasing the size of standard datasets. The largest dataset available currently is the retinal connectome which is 16.5T in size [3]. New algorithms need to be able to take advantage of distributed computing,many core processors, and streaming technologies. Therefore, image segmentation techniques must account for these data characteristics in order to identify and successfully track neurons across hundreds of sections.

1.4 Computational Approach for Neuron Reconstruction

Given the challenges outlined in the previous sections, We have developed a series of methods to segment neurons first in 2D and then in 3D by linking together all the regions from the 2D segmentation. Figure 1.2 demonstrates the difficulty in segmenting neuron membranes from two ssTEM images using relatively straight forward image processing techniques, anisotropic smoothing and calculating the gradient magnitude. Both images contain structures that are difficult to isolate. For example, in an effort to strengthen membranes in the top image of Figure 1.2, vesicles are confused with membranes and enhanced. Similarly, in Figure 1.2(a), edges of neurons closely resemble the edges of the nucleus, making it difficult to isolate just the neuron edge features.

There are two general approaches for neuron segmentation. One approach focuses first on the detection of neuron membranes in each 3D volume directly, while other methods choose to perform a segmentation first on each 2D section followed by a linking of the segmented regions between sections. These approaches are discussed in more detail in Chapter 2. Because of the anisotropic nature of this data, we approach this problem by first detecting neurons in 2D sections using a membrane detection machine learning method. Then we link together these neuron profiles into a 3D volume using an optimal path finding algorithm.

1.4.1 2D Neuron Membrane Detection and Neuron Segmentation

The method developed in this dissertation for neuron membrane detection is summarized in Figure 1.3. First, membranes are detected using a series of artificial neural network (ANN) classifiers and image stencil neighborhood feature vectors to detect neuron membranes in

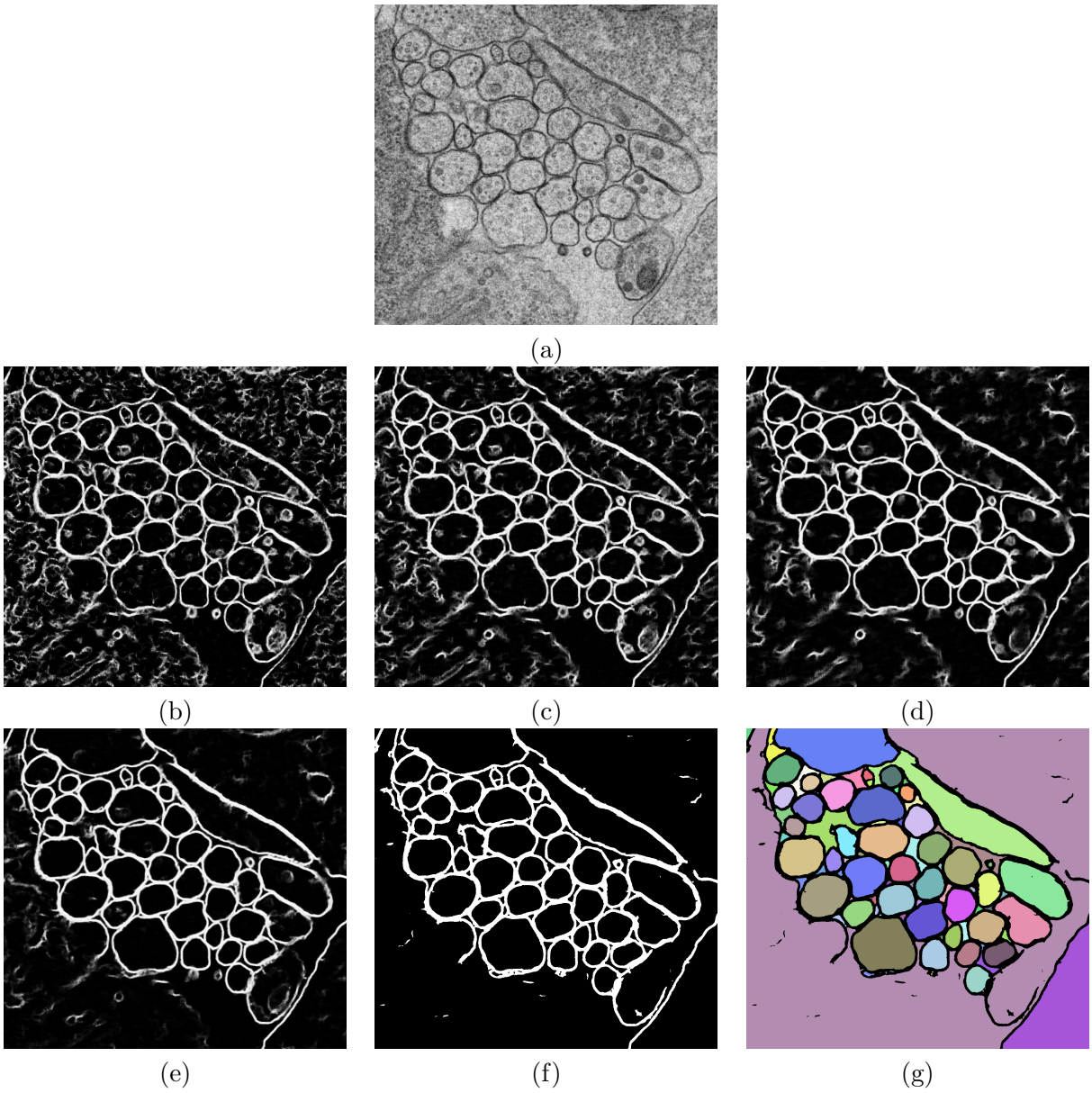


Figure 1.3. Output of the method on a test image from the *C. elegans* dataset. (a) is the raw image, (b) (d) are stages 1, 2 and 5 of the series ANN, (e) is the output from the sequential series ANN, (f) is the output from the tensor voting, and (g) is the region segmentation after using the watershed segmentation algorithm.

2D images [52]. Each ANN is trained in series using information from the previous network as input to the next ANN. The first ANN uses as input the intensity values sampled directly from the image. The input to the subsequent ANNs in the series is comprised of the same set of image values, in addition to the output of the previous ANN sampled on a stencil of nearby pixels. The ANNs in the series, therefore, have different inputs even though they have a common desired output. The advantages of this method are two fold. First, the classifier uses raw data, that is, the image intensities, rather than a constrained version of the image as given by responses to a large filter bank or statistical features that will not scale well for large datasets. Second, the use of the serial ANNs provides context, which is information from nearby pixels that contributes to the learning, providing increasing amounts of relative information at each stage of the network. As a result, the series of ANNs learns to remove vesicles and mitochondria from the membrane detection and close gaps in places where the membrane is weak. Figure 1.3(b) through (d) depict several stages of this process. This is described in detail in Chapter 3, where we demonstrate the improvement from the combined use of stencils and the series of ANNs for three datasets with distinctly different characteristics.

Initially, membrane detection is limited to features within a 2D section. Chapter 4 extends this method to train on learned membranes from neighboring sections. Given the anisotropic nature of the data, sequential sections have very poor membrane correspondence. To account for this, classified results representing the membrane probability image are registered and a 3D stencil that spans 3 sections is formed for training a final ANN. Finally, tensor voting, a method for closing remaining gaps, is used as a post processing step. This provides significantly improved segmentation results by closing gaps and strengthening membranes. This is demonstrated in Figure 1.3(e) and (f).

Finally, regions are segmented representing each neuron profile in a 2D section using a flood fill (shown in Figure 1.3(g)) or a watershed segmentation algorithm. These are discussed in more detail in Section 3.4.

1.4.2 3D Neuron Reconstruction

Two-dimensional neuron regions can now be linked across sections using an optimal path finding algorithm on a graph data structure [55]. This method calculates a normalized correlation between regions in one section with regions in the next section. The neuron regions become nodes and the correlations are used to compute weights for the edges in

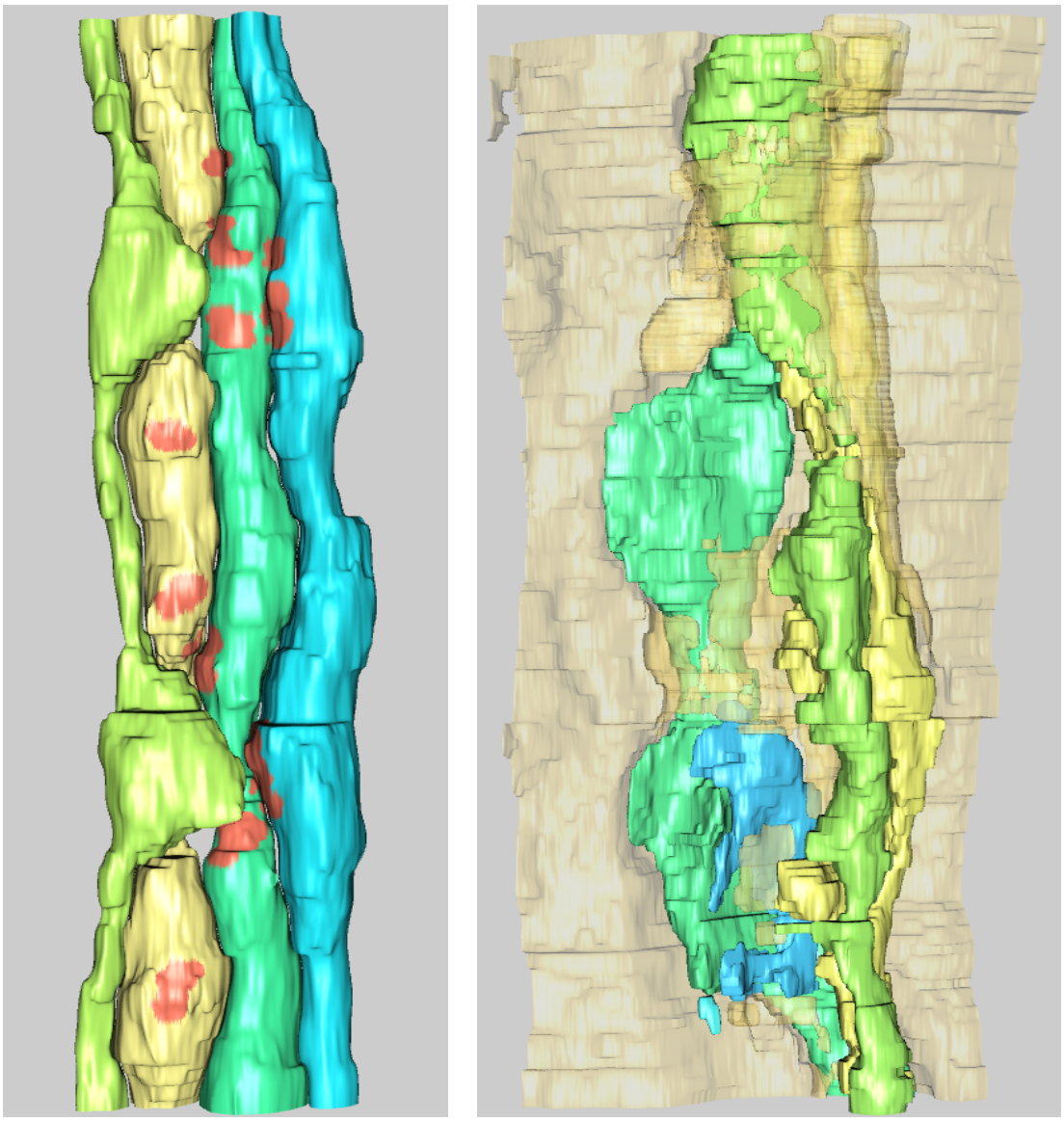
the graph. Dijkstra's algorithm finds the minimum cost path through the graph for each node from the first section of the graph. The regions through the graph become the path the neuron takes through the volume. These regions are used to segment the volume and render each neuron in 3D.

The Neuron Reconstruction Viewer (NeRV) was developed to help with this final segmentation process. NeRV has the ability to edit segmentations, recompute correlations, and merge paths between sequential sections. Most importantly, NeRV gives users the ability to selectively view neurons in 3D (since datasets can contain hundreds of neurons) together with the raw image data. This was a technical challenge due in part to the user interaction and the management of memory and images in a single visualization environment.

Figure 1.4 is an example of a full 3D neuron segmentation for four key neurons and nearby muscles from the *C. elegans* dataset. The neuron mebranes were segmented using machine learning methods and edited by hand to fix incorrectly segmented regions. The muscle structures were segmented entirely by hand, since these structures appear differently than the membranes and would require a different algorithm. From an applications perspective, this 3D model shows the motor neurons in the ventral nerve cord and their processes interdigitating along the lateral edge of the nerve bundle (Figure 1.4(a)) to make contact with the muscles (Figure 1.4(b)). Multiple muscles, in turn, must send processes to these motor neurons to receive input. Areas where this communication is occurring are marked in red. There are three motor neuron inputs into these muscles: the VA neurons release acetylcholine during backwards movement, the VB neurons release acetylcholine during forward movement, and the VD motor neurons release GABA to relax the muscle to allow sinusoidal movement. These data demonstrate that axons do not precisely interweave. GABA neurons run alongside a group of muscle arms and form multiple synapses to differing subsets of muscles before giving way to acetylcholine motor neurons. By contrast, the two types of acetylcholine neurons usually form contacts to the muscles simultaneously. Again, they form 2 to 3 contacts to the muscles for a segment of axon before giving way to the GABA motor neuron. This demonstrates the importance and diagnostic capabilities of full connectivity diagrams and renderings.

1.5 Contributions

This dissertation investigates image processing techniques for extracting neuron profiles from EM data and visualizing them in 3D. Beginning with raw image data, a complete data



(a) (b)

Figure 1.4. Reconstructions of neurons and muscles from the *C. elegans*. (a) Demonstrates renderings of the four neurons competing for information from the muscles. The location of the synapses, which were extracted from user specified locations, are shown in red on the neurons. (b) Similar rendering of the muscles that run alongside the motor neurons.

processing pipeline is provided to segment and view 3D neuron structures. The following list outlines the specific objectives of this dissertation, including published and submitted works:

1. *Develop a machine learning method for accurately segmenting neuron membranes in 2D, removing intracellular structures and closing contours.* The goal of this work is to detect neuron membranes in large images using using supervised machine learning. Experts annotate EM images which are used to train a classifier. The input to the classifier is raw image intensities, making this method easy to apply to different types of datasets. See Chapters 3 and 4 for more detail [53, 52, 75, 98].
2. *Given neuron segmentations in 2D, develop a method for connecting neurons across sections to form 3D neuron models.* This is solved using an algorithm that finds the shortest path in a graph data structure. The neuron regions are nodes in a graph and the connections between regions are edges. The edge weights are determined by the correlations and the distances between regions. Paths are constructed between nodes that represent whole neuron segmentations. More detail on this method is covered in Chapter 5 [55].
3. *Incorporate large EM datasets into existing algorithms.* Handling data produced from current EM efforts means developing flexible algorithms to handle images that will not fit into memory or take extensive time to process. This is handled in part by dividing up embarrassingly parallel problems to run in parallel on many processor, multicore computers. This allows us to simply farm out various tasks on many computers. However, algorithms also need to account for increased processing time and memory storage, so simple techniques for memory management and file I/O must be used. This is discussed in more detail in Section 5.4.
4. *Incorporate existing reconstruction methods into a software tool.* The neuron reconstruction viewer (NeRV) was developed as a prototype to demonstrate how the current image processing tools interact, give the users a method for viewing large image datasets, and edit segmentations. This tool aids researchers in the reconstruction of neurons using the segmentation learned from a classifier trained to detect neuron boundaries and connect neurons across sections using a minimum cost path finding approach to form 3D models of neurons (Section 5.4) [54].

1.6 Overview

This work develops methods to address the challenges outlined in Section 1.3 for several different datasets, including the *C. elegans*, rabbit retina, and the mouse neuropil. These datasets are outlined in Table 1.1. Each dataset has unique qualities to it, while at the same time, the goals are the same. All of the experiments presented in this dissertation were performed on an 8×2.8 GHz shared memory computer with 8G of memory and a 32×2.93GHz shared memory computer with 200G of memory.

Chapter 2 gives an overview on current, state of the art, techniques for image processing of EM image data. Given the complex structure of EM data, several approaches by other researchers are presented, ranging in scale from the segmentation of one neuron at a time to the segmentation of many neurons at once. Supervised machine learning approaches are shown to be useful along with user interactive boundary detection algorithms.

Chapter 3 describes the specific supervised machine learning approach used for the datasets presented in this paper. First, we give an overview of the concept of learned context across a series of ANNs using only image intensity data. This is validated using a standard test dataset and them demonstrated on relevant EM image data.

Chapter 4 improves on the previous segmentation method by using information from learned membranes in sequential sections to close gaps. The learned membranes are registered to sequential sections for use in a 3D stencil in an ANN. Tensor voting is applied to the output of this last step to close small remaining gaps in the membranes.

Finally, Chapter 5 demonstrates how to join all the segmented regions and visualize the neurons in 3D with a software tool. Also, all the software developed to segment and reconstruct neurons are described.

The conclusions, in Chapter 6, give an overview of the methods described in this dissertation and suggestions for future work.

Table 1.1. Table briefly summarizing all the datasets used in this dissertation.

Dataset	Source	Pixels	Resolution (nm)	Section
Weizmann Horses	natural	328 images	N/A	3.5.1
*C. elegans*VNC	ssTEM	662×697×150	6×6×33	3.5.2.1
*C. elegans*VNC	ssTEM	4008×2672×400	6×6×33	4.4.1, 5.5.1
Rat Retina	ssTEM	1000×1000×28	2.5×2.5×90	5.3.1
Mouse Retina	ssTEM	1200×1200×341	2×2×80	3.5.2.2
Mouse Neuropil	SBFSEM	4000×4000×400	10×10×50	4.4.2, 5.5.2

CHAPTER 2

RELATED WORK

Accurate detection of neuron membranes, used for whole neuron reconstruction, in EM is a difficult problem given the presence of intracellular structures. This makes simple thresholding, edge detection (e.g., Canny), and region growing methods ineffective for the detection of neuron membranes. Some example images and results with traditional image processing methods are shown in the previous section, Figure 1.2. There are several general approaches for neuron segmentation. The first approach focuses on the detection of neuron membranes in each 2D section. These boundaries can be used to identify individual neurons, which are then linked across sections to form a complete neuron. A second approach uses all 3 dimensions to perform a segmentation, treating the data as a single 3D volume. Within these categories, some methods prefer to segment neurons based on geometric models and filters while others prefer a machine learning approach, such as random forest and neural network classifiers. Tools that incorporate automated techniques into a user interface are still in the early stages of development and rely on algorithms that cater specifically to the algorithms designed for a particular dataset. The main focus of this thesis is to accurately perform 2D neuron segmentation over large sections and link these segmentations through the sections to form a full 3D reconstruction. These methods are built into a tool to assist users in correcting segmentations and joining paths in large datasets.

One obvious approach for neuron segmentation in this area focuses on emphasising neuron membranes in images using a simple set of convolution kernels [1], radon-like features [59], or anisotropic smoothing based on the Hessian structure tensor [101]. The output from these methods can be used for segmentation either through thresholding or as part of a longer data preprocessing pipeline.

There are several methods that approach the neuron membrane identification process as a 2D image segmentation problem. The most direct segmentation methods, such as active contours, in both parametric and level set forms [51, 63, 111, 9], can provide smooth, accurate segmentations of cells. However, they are very sensitive to initialization, which must be

close to the neuron membrane, and often confuse internal structures for neuron membranes. Graph cut segmentations on EM images produces promising neuron segmentations starting from a manual initialization and compute globally optimal solutions [112, 118]. However, determining the correct combination of data, smoothness, and penalty terms is a challenging task and subject to change with each dataset. Given an edge term that suppresses internal structures, such as one that is derived from the output of the classifier, these methods may be more promising. However, methods requiring an initialization are more appropriate for segmenting only a few cells. As a reminder, the goal of this thesis is the automatic segmentation of thousands of cells which renders manual initialization impractical.

More generally, the detection of complete membranes, even when portions of the membrane are low in contrast, is closely related to the contour completion and salient contour extraction problems which have been studied extensively in the computer vision literature. Various approaches have been proposed including spectral clustering and, graph analysis [89, 64, 31, 119], tensor voting [96], probabilistic models [84] and conditional random fields [83]. Hierarchical methods for grouping contours together can be formulated using a graph [119] or watershed transform [5]. Some related work in this area also uses supervised classification that combines features across different scales to detect edges and close contours [26, 90].

More direct 3D approaches, such as Jeong *et al.* transform 2D active ribbon segmentations of membranes to more robustly segment neurons in 3D with the use of an arbitrary cutting plane [46]. Graph cuts have also been extended to 3D to include an energy term that accounts for probabilities of membranes in adjacent sections [118].

Supervised machine learning methods have proved to be useful for detecting membranes in EM images [44]. For example, Jain *et al.* use a multilayer convolutional ANN to classify pixels as membrane or nonmembrane in specimens prepared with an extracellular stain [43]. The convolutional ANN has two important characteristics: it learns the filters for classification directly from data, and the multiple convolutions throughout the layers of the network account for an increasing (indirect) filter support region. Andres *et al.* propose a multipart segmentation process that use statistical learning and watersheds to segment neural tissue [4]. Machine learning methods that minimize the rand index [106, 107] or the warping error [42], as opposed to the mean squared error, demonstrate an improvement in the segmentation, rather than the pixel wise classification. These methods produce clear segmentations of the membranes, however, they are aimed at more isotropic datasets in

which the stain used on the specimen suppresses the contrast of intracellular structures leaving only the cell membranes visible [14]. This preparation technique potentially simplifies the segmentation task and operates under the assumption that connectivity can be determined from structure [12, 77]. However, the other philosophy regarding this processes requires the detection and staining of synapses for full neural circuit reconstruction, which are characterized by intracellular structures [72]. These structures, as seen in the data presented in this paper, complicate the segmentation process.

In other work based on supervised learning, simple classifiers such as a single perceptron applied to a carefully chosen set of features has been shown to provide promising results in identifying membranes in EM images [71]. Nevertheless, this method still needs significant post processing to connect membranes and remove internal cellular structures. Similarly, Venkataraju *et al.* propose using local context features computed from the Hessian matrix to train a boosted classifier to detect membranes, which highlights the importance of context for membrane detection [108]. The results obtained with these methods demonstrate not only the complexity of the problem, but also the potential of supervised machine learning for neuron segmentation.

There are several existing software efforts that incorporate many of the above algorithms specifically for reconstructing neural circuits from biological volumetric images. These tools provide an interface to the data and contour tools to segment structures in a stack of EM images. One of most widely used software tools is Reconstruct [28, 30], which enables users to view and outline structures of interest and then render as three dimensional volumes. Combining Reconstruct with automated methods, such as the ones proposed by Mishchenko [72] has resulted in scientific discoveries regarding the predicted location of synapses within a neuron. IMOD has also proved to be another useful tool for segmenting and rendering structures from a variety of biological volumetric image data [58]. A more comprehensive set of tools for reconstruction is the Cell Centered Database [68] which performs not only annotation on EM images, but also provides data management and protein knowledge base interfaces. While these tools are critical in segmentation and reconstruction of EM data, the goal of this paper is to segment many structures from large sets of data and design of a tool that can use automated algorithms and handle large sets of images that may not fit into memory. Towards this goal, two software programs, the Serial Section Reconstruction and Tracing Tool (SSECRETT) and NeuroTrace segment large image databases [46, 45]. SSECRETT is an interface for slice based viewing of

large volumes using a client server architecture to request only the data needed by the user. NeuroTrace incorporates 2D level set segmentation tools for segmenting individual sections, and then using those segmentations to identify long neuronal processes. Tools that combine fast machine learning algorithms with user annotation include ilastik [92]. Most relevant to the work in this thesis is Raveler, a new unique tool containing advanced editing capabilities for proofreading segmentations and linking together regions to form 3D neuron representations [18]. These combined tools produce vital reconstruction data, but the interface to the data is specific to the implemented algorithms and still requires the user to initialize each neuron for segmentation. The software program designed for this dissertation manages memory for large datasets. It is designed to incorporate automated segmentation algorithms and user selected segmentations. Also, it provides an interface specific to the segmentation method presented in this paper to make corrections and most importantly, view the raw image data with its 3D segmentation.

The success of machine learning methods applied to EM data and the anisotropic nature of the data in this thesis leads to a solution that incorporates a machine learning method for 2D membrane detection followed by a linking of neurons in 3D. The complicated nature of the data gives rise to a tool that aids users in building full 3D models.

CHAPTER 3

2D MEMBRANE DETECTION AND NEURON SEGMENTATION

3.1 Introduction

The goal of the work presented in this chapter is the 2D segmentation of individual neurons in EM images. Segmentation of neurons in these images, an essential step of the reconstruction pipeline, is challenging because of noise, anisotropic resolution, variable shapes, and brightness, and the presence of confounding structures. The method developed to best handle these challenges uses a series of artificial neural networks (ANNs) in a framework combined with a feature vector that is composed of image intensities sampled over a stencil neighborhood. Several ANNs are applied in series allowing each ANN to use the classification context provided by the previous network to improve detection accuracy. The following sections develop the method of serial ANNs and show that the learned context does improve detection over traditional ANNs. We also demonstrate advantages over previous membrane detection methods. The results are a significant step towards an automated system for the reconstruction of the connectome.

3.2 Neuron Segmentation Using Machine Learning

Identifying neuron membrane pixels in EM images using automated methods is a challenging task. Membranes are sparse structures that span elongated spaces and look similar to internal structures, such as vesicles, when examined in small neighborhoods. Changing contrast and different levels of noise also complicate this process. Finally, membranes from different tissues using different EM techniques result in different neuron membrane features (i.e. edge verses valleys). Membrane detection techniques for EM data must be able to handle different types of image data and the detection of extended structures.

As discussed in Chapter 2, supervised machine learning methods have proven useful for detecting membranes in EM images. To address the challenges presented above, we

developed a machine learning method that combines two bodies of related work. The first, by Jain *et al.* use a multilayer convolutional ANN to classify pixels as membrane or nonmembrane in specimens prepared with an extracellular stain [43]. The convolutional ANN has two important characteristics: it learns the filters for classification directly from data, and the multiple convolutions throughout the layers of the network account for an increasing (indirect) filter support region. This method will work well for different types of image data, since it uses, as input, raw pixel data. In addition, the multiple convolutions enable the classifier to learn elongated structures that extend across the image without using large areas of the image as input. However, this method contains more than 30,000 parameters and, therefore, is computationally intensive and requires very large training sets. Also of particular relevance is Tu's auto-context framework [105] from the computer vision literature, which uses a series of classifiers with contextual inputs to classify pixels in images. In Tu's method, the "continuous" output of a classifier, considered as a probability map, and the original set of features are used as inputs to the next classifier. The probability map values from the previous classifiers provide context for the current classifier, by using a feature set that consists of samples of the probability map at a large neighborhood around each pixel. Theoretically, the series of classifiers improves an approximation of an a posteriori distribution [105]. Hence, each subsequent classifier extends the support of the probability map, improving the decision boundary in feature space, and thus the system can learn the context, or shapes, associated with a pixel classification problem. Similar to the convolutional network, this means that a classifier can make use of information relayed by previous classifiers from pixel values beyond the scope of its neighborhood. This approach also works well for the type of extended structures being detected in this data. The particular implementation demonstrated by Tu uses 8,000 nonspecific, spatially dispersed, image features, and a sampling of probability maps in very large neighborhoods. This is appropriate for methods that use a boosting classifier strategy [33] and are being performed on smaller scale machine learning problems. However, for the data targeted in this disseration, such as the full rabbit retina dataset [2], which is total is 16TB, it is impractical to calculate thousands of image features in order to train the classifier.

One of the main contributions of this work is the formulation of a series of ANNs in an architecture similar to auto-context and convolutional networks. For our series ANN architecture, we chose to use an ANN instead of a boosting strategy and exploit the concept of increasing levels of context with each ANN to learn the membrane pixels. However, similar

to Jain *et al.* we choose to learn the image features directly from the data and use the image intensities as input to our architecture, rather than preprocessing the data and computing thousands of image features. This provides us with a much smaller set of features, compared to the convolutional network, and allows for flexibility and training of large datasets. Also, the use of the serial ANNs and increasing context allows us to focus on small sets of image features to detect membranes, while also eliminating pixels that represent vesicles or other internal structures.

3.3 Serial Neural Network Architecture

The method developed here for neuron membrane detection combines ANN classifiers and image stencil neighborhood feature vectors. First, the artificial neural network is presented, along with a discussion of the feature vectors used as input. Finally, the whole system for using ANNs in series and composing the input is discussed. The following sections provide details on each of these components.

3.3.1 Artificial Neural Network

Given the success of ANNs for membrane detection [43, 71] and because auto-context is not specifically tied to any classifier, a multilayer perceptron (MLP) ANN is implemented as the classifier for this problem. An MLP is a feed-forward neural network which approximates a classification boundary with the use of nonlinearly weighted inputs. The architecture of the network is depicted schematically in Figure 3.1. The output of each processing element (PE) (each node of the ANN) is given as [37, 81]

$$y = f(\mathbf{w}^T\mathbf{x} + b), \tag{3.1}$$

where f is, in this case, the *tanh* nonlinearity, $\mathbf{w}$ is the weight vector, and b is the bias. The input vector $\mathbf{x}$ to PEs in the hidden layer is the input feature vector discussed in more detail in Section 3.3.2. For the output PEs, $\mathbf{x}$ contains the outputs of the PEs in the hidden layer.

ANNs are a method for learning general functions from examples. They are well suited for problems without prior knowledge of the function to be approximated (a.k.a., "black box model"). They have been successfully applied to robotics [80, 114] and face and speech recognition [19, 82], and are robust to noise. Training uses gradient descent to solve for a solution which is guaranteed to find a local minimum. However, several trade offs occur in training ANNs regarding the size of the network and the number of inputs. An ANN

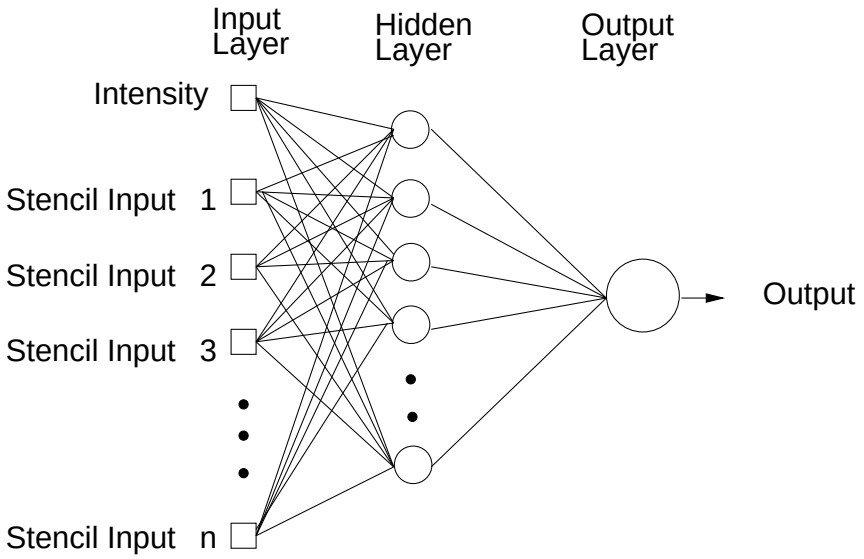


Figure 3.1. Artificial neural network diagram with one hidden layer. Inputs to the network, in this framework, include the image intensity and the values of the image at stencil locations.

with too many hidden nodes can lead to overfitting of the network [37], resulting in a set of weights that fits well to the training data, but may not generalize well to test data. At the other extreme, if the number of hidden nodes is insufficient the ANN does not have enough degrees of freedom to accurately approximate the decision boundary. The number of features should also be kept small to mitigate the problem of high dimensional spaces. Generally speaking, as the dimensionality of the input space increases, the number of observations becomes increasingly sparse which makes it difficult to accurately learn a decision boundary. Additionally, the training time tends to scale with the amount of training data and size of the network, and therefore training smaller networks with fewer features is generally preferable. Hence, the number of inputs to each ANN should be large enough to describe the data, while keeping this number to a minimum.

3.3.2 Image Stencil Neighborhood

Choosing the best set of features to use in training an ANN is crucial for obtaining good segmentations. The field of computer vision has made available several possible strategies. A possible approach uses large sets of statistical features as the input to a learning algorithm. These features can include simple local and nonlocal properties, including the pixel values, mean, gradient magnitude, standard deviation, and Hessian eigenvalues [4, 108, 105]. These

attempt to present the learning algorithm with a large variety of mathematical descriptors to train on, and are designed to work on a variety of data types. To achieve this generality, however, large numbers of these features are required to train a classifier. Another approach is to design a set of match filters and apply them to an image to approximate a pixel's similarity to a membrane. This works well if the membranes in the image are uniform and respond well using cross correlation [62, 87]. Moreover, the design of the filter bank requires significant a priori knowledge of the problem. Yet, the fixed design may not be optimal for the dataset. Most importantly, the match filters have to be redesigned for datasets with different characteristics. On the other hand, learning these filters from training data, as in the case of convolutional networks [43], has the advantage that no a priori knowledge is required. A similar idea was been used in texture classification where is was shown that direct sampling of the image with a patch is actually a simpler and better approach for training a classifier compared to the use of filter banks [109]. Image patches have also been used successfully for texture segmentation [6] and image filtering [7, 15, 97]. Similarly, using image neighborhoods in this case allows the ANNs to learn directly on the input intensity data, giving the classifier more flexibility in finding the correct decision boundary. A square image neighborhood is defined as an image patch, shown in Figure 3.2(a), centered at pixel k, l,

$$P_{k,l} = \{I_{k+i,l+j} : i, j = -\frac{R-1}{2}, \ldots, \frac{R-1}{2}\}. \tag{3.2}$$

R is the width of the square image patch. Unfortunately, the size of the image patches required to capture sufficient context can be quite large. For this reason, we propose using as input to the ANNs the values from the image and probability map of the previous classifier sampled through a stencil neighborhood, shown in Figure 3.2(b). A stencil is also centered at pixel k, l and defined as,

$$S_{k,l} = \cup_{a=1}^{n} B_{k,l,a} \tag{3.3}$$

where

$$B_{k,l,a} = \{I_{k+ai,l+aj} : i, j = -1, 0, 1\}, \tag{3.4}$$

and n is the number of rows the stencil spans in the image. The stencil in Figures 3.2 and 3.3 can cover large areas representing the desired feature space, but samples it with a spatially adaptive resolution strategy. In this way, an ANN can be trained using a low dimensional feature vector from image data, without having to use the whole image patch. Since the number of weights to be computed in an ANN are dominated by the connection

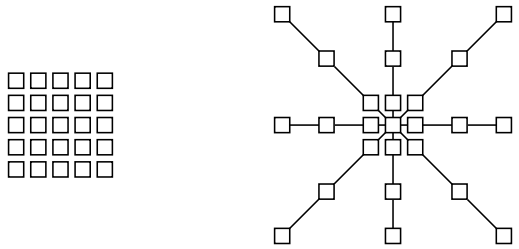


Figure 3.2. Two image neighborhood sampling techniques: image pixels sampled using (a) a patch and (b) a stencil. For this example, the stencil contains the same number of samples, yet covers a larger area of the data. This is a more efficient representation for sampling the image space.

between the input and the hidden layers, reducing the number of inputs reduces the number of weights and helps regularize the learned network. Moreover, using fewer inputs generally allows for faster training. With this, one aims to provide the classifier with sparse, but sufficient context information and achieve faster training, while obtaining a larger context which can lead to improvements in membrane detection. This strategy, combined with the serial use of ANNs (described in Section 3.3.3), grows the region of interest for classification within a smaller number of stages and without long training times. For large image features, stencils such as the one in Figure 3.3 are required.

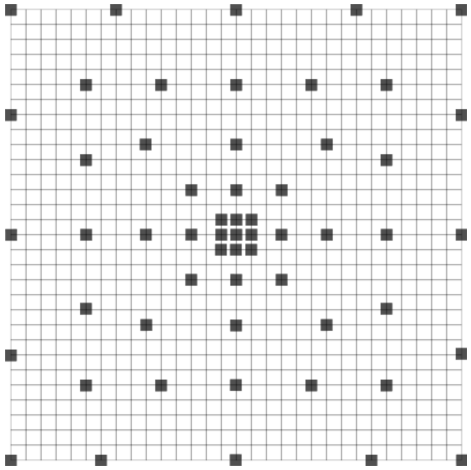


Figure 3.3. Larger image neighborhood sampling technique, covering a 31×31 patch of image pixels.

3.3.3 Serial Artificial Neural Networks

Using principles from auto-context, a series of classifiers that leverage the output of the previous network to gain knowledge of a large neighborhood is implemented. For the first classifier, the input is the image intensities around a pixel sampled using a stencil as described above. For the ANNs in the remaining series, the input vector contains the samples from the original image, used as input to the first ANN, appended with the values from the output of the previous classifier sampled through the stencil neighborhood, yielding a larger feature vector. While the desired output labels remain the same, each ANN is dependent on the information from the previous network and therefore must be trained sequentially, rather than in parallel. Figure 3.4 demonstrates this flow of data between classifiers. I denotes the image, S the image values sampled from the image using the stencil, C the output from the ANN, and T the threshold applied to C at zero, yielding the final membrane detection.

The serial structure allows the classifiers to gather, with each step, context information from a progressively larger image neighborhood to the pixel being classified, as occurs with a convolutional ANN. The pixel values are sampled with a stencil neighborhood over each pixel, containing the pixels within the stencil (Figure 3.2). The classification feature vector is also obtained with a stencil neighborhood placed over each pixel containing information about the classes, as determined by the previous classifier. Indirectly, the classification from the previous ANN contains information about features in surrounding pixels, that is not represented in the original feature set. This allows the subsequent networks in the series (Figure 3.4) to make decisions about the membrane classification utilizing nonlocal information. Put differently, each stage in the series accounts for larger structures in the

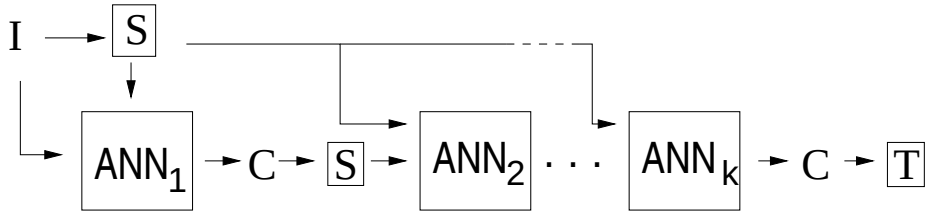


Figure 3.4. Serial neural network diagram demonstrating the flow of information between ANNs. I is the original image, C is the output (probability map) from the classifier before thresholding, S is the stencil that samples the image data, and T is the final output from the classifier thresholded to produce a binary segmentation.

data, taking advantage of results from all the previous networks. This results in membrane detection that improves after each network in the series. Figure 3.5 visually demonstrates the classification improving between ANNs in the series as gaps in weak membranes are closed and intracellular structures are removed with each iteration in the series. The receiver operating characteristic (ROC) curves in Figure 3.6 also demonstrate the increase in detection accuracy after each ANN in the series.

Combining the original image features with features sampled from the output of the previous classifier is important because, in this way, the membrane structure relevant for detection is enforced locally and then again at a higher level with each step in the series of classifiers. One of the advantages of this approach is that it provides better control of the training, allowing the network to learn in steps, refining the classification at each step as the context information it needs to correctly segment the image increases. Again, note that the membrane structure is learned directly from the data. Compared to a single large network with many hidden layers and nodes, such as the convolutional ANN of Jain *et al.* [43] which requires 34,000 parameters, the proposed classifier is easier to train. This is mainly because each of the ANNs have a relatively small number of parameters. For example, given a single ANN used to compute the results in Section 3.5.2.1, the number of parameters needed is approximately 500 for the first ANN and 1100 for the remaining ANNs in the series. The number of weights in an ANN with a single-hidden layer is given by $(n+1)h + (h+1)$, where n is the number of inputs and h is the number of nodes in the hidden layer. For the first ANN in the series, $n = s$, where s is the number of points in the stencil. For the remaining ANNs in the series, $n = 2s$, since the original image and the output from the previous classifier are each sampled once. The total number of parameters across the whole series totals to approximately 5000. In contrast, a convolutional ANN needs $(n+1)h$ for the first layer, and $(nh+1)h$ for the remaining layers, an h^2 dependence [43]. Hence, much less training data is needed in this approach, which is hard to obtain, since the ground truth must be hand labeled[1]. Furthermore, the training is simpler since backpropagation is less likely to get stuck on local minima of the performance surface [37, 81], and the network will train much faster. Moreover, this accounts for a smaller and simpler network which can be trained from smaller numbers of features in the input vector. The series of ANNs is much

[1]According to the "rule-of-thumb" in [81], one needs at least 10× training samples of the total number of parameters. Thus, compared to Jain *et al.* [43] convolutional ANN, the approach presented here needs about 27× less training samples, for the values given.

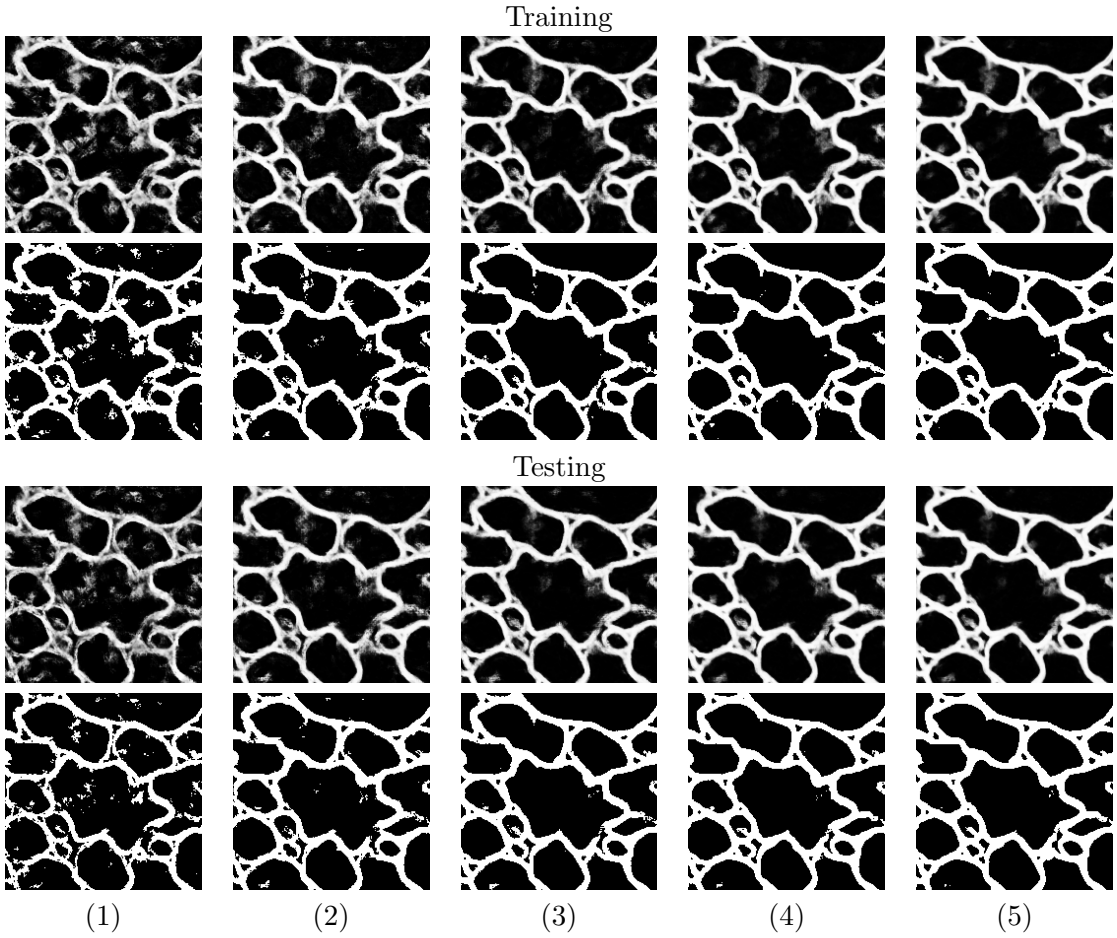
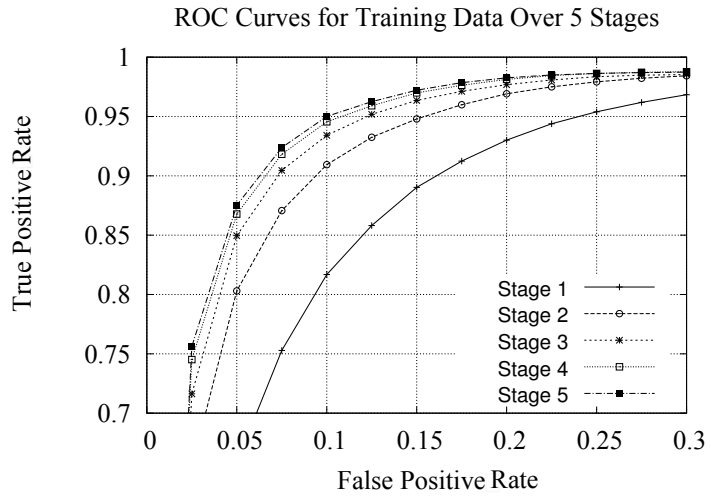
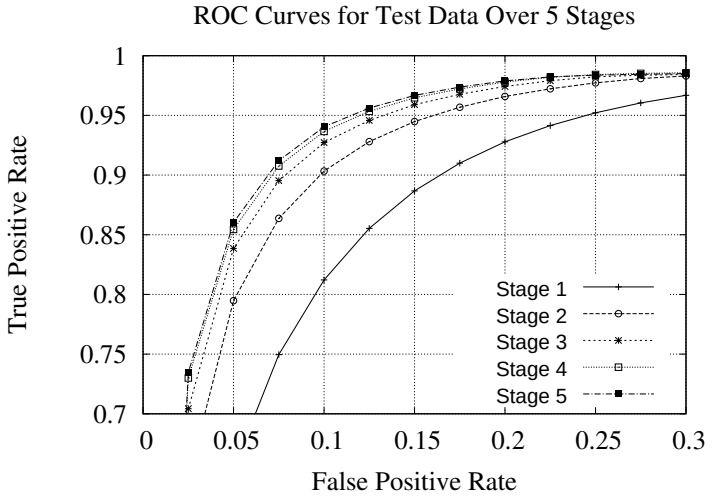


Figure 3.5. Example output using the same image, first as part of a training set (top two rows), and then separately, as part of a testing set (bottom two rows), at each stage (1-5) of the network series. The output from each network is shown in rows 1 and 3. Rows 2 and 4 demonstrate the actual membrane detection when that output is thresholded. The network quickly learns which pixels belong to the membranes within the first 2-3 stages, and then closes gaps in the last couple of stages.



(a)



(b)

Figure 3.6. ROC curves for the (a) training data and (b) testing data for each stage of the network on the *C. elegans* dataset.

more attractive to train, as opposed to using a single large network with many hidden layers and nodes. A single large network would be time consuming and difficult to train due to the many local minima in the performance surface.

3.4 Region Segmentation

Given detected membranes in each 2D section, neurons can be segmented in each section using either a watershed segmentation [34, 40] or a simple flood fill algorithm on the thresholded probability map. The flood fill algorithm operates on thresholded data and works best when a user has corrected segmentations with hand editing and wants a precise neuron membrane representation at every section. For larger problems and cases where the user might not want to correct membrane detection, the watershed algorithm has the advantage in that it can close some small gaps automatically. In this instance, the watershed segmentation would be applied to the output from the series of ANNs and the user picks the depth of the watershed that best segments the data. In using the watershed, the image becomes as a real valued height field where the depth corresponds to the level of the water in the landscape. An example output from the watershed is shown in the results of Section refsec:experiment:celegans. These results also depict how close the final segmentation is to the true segmentation.

3.5 Results

This method is demonstrated on two different types of datasets. The first, the Weizmann horse database [11], demonstrates the usefulness of this method in a general setting by comparing the series of ANNs to Tu's previous work. The second set of data contains two different types of EM images, and shows the practicality of this method on application data.

3.5.1 Weizmann Horse Database

The Weizmann horse dataset consists of 328 gray scale horse images with different sizes. The images were randomly divided into training and testing sets of equal size. The series of ANNs was used to learn to identify the horse in each image. Each ANN had 30 hidden nodes and a step size 0.0001 was used. Five Monte Carlo instances were used per network to minimize problems with local optima. Stopping of training was decided by measuring the performance on a cross validation set of 20% of the training images. The remaining images were used for testing. The input vectors were formed by sampling the input image

and output of the previous stage with the stencil neighborhood, given in Figure 3.3, without the center pixel in the latter case.

Figure 3.7 demonstrates these results on selected horse images. Visually, the results are comparable to Tus auto-context [105]. Note that in the series of ANNs, the complexity increases with the context because the feature filters are learned from data. In auto-context, however, the filters are given and can have very large support, but one needs very large sets of filters to ensure a reasonable basis. These results were generated using only 49 features, 25 from the input image and 24 from the configuration neighbors, whereas Tu [105], using a probabilistic boosting tree algorithm, selects from a set of around 12000, 8000 from the input image and 4000 from the previous classification output. This indicates that feature filters should be learned from data to ensure that all available information is used and they accurately fit the application. Again, the ROC curves, shown in Figure 3.8, verify the observations quantitatively. The f-value, shown in Figure 3.9 computed on this dataset is 0.834 (0.823 at zero threshold) compared to below 0.84 using auto-context [105]. Finally, these results highlight once more the importance of sequentially applying the different ANNs. As shown in the images, the first stages detect primarily the contours of the horses which are easier to infer locally but fails to discern the body of the horse because the interior of the horse can be dark or light while the edges are easier to detect.

However, using the context information from the output of the previous models in addition to the input image, the later stages are able to fill in the segmentation. Note that the context information increases implicitly with each stage because of the context on the output of the previous model. Training all the models of the series ANN took almost 2 weeks for the horse segmentation experiment. (Note that, in the latter case, the training data comprises more than 10 million training feature vectors.) On the other hand, once learned, the classification is very fast, taking less than 0.25 seconds per image, compared to the 40 seconds reported by Tu [105].

3.5.2 Serial Section TEM

Two TEM datasets are used as application specific test cases for the proposed method. The first dataset is a stack of 50 sections from the ventral nerve cord of the *C. elegans*. The second dataset is a single section from the 16TB rabbit retina dataset. These datasets contain very different types of neural cells. The *C. elegans* data has a resolution of 6nm×6nm×33nm and each 2D section is 662×697 pixels. Neuron membranes in the

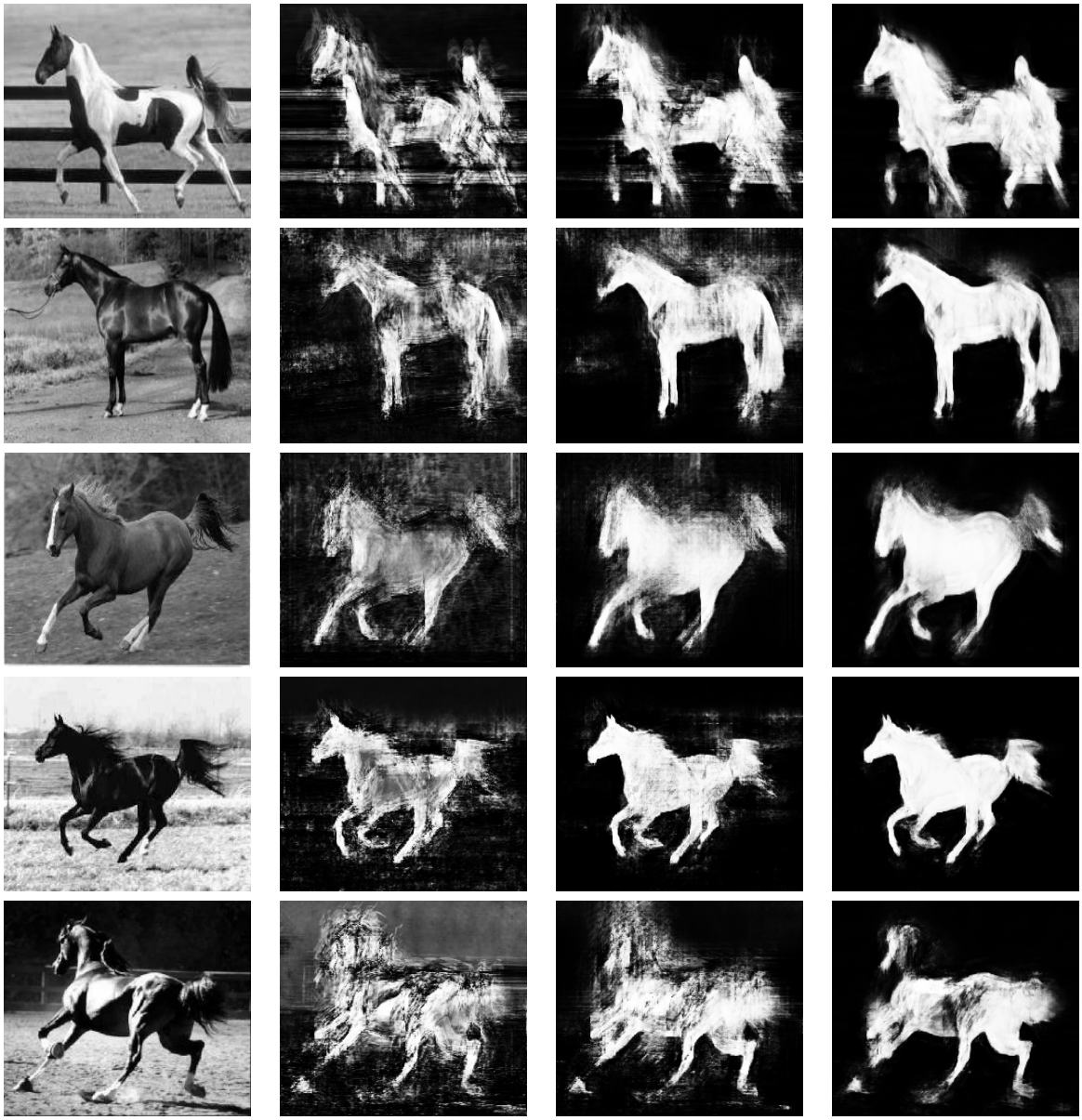
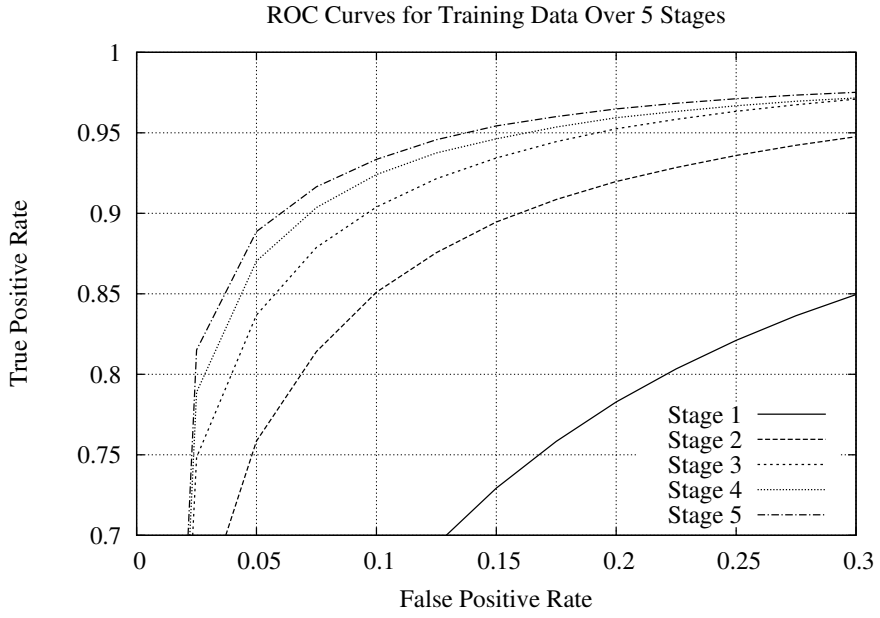
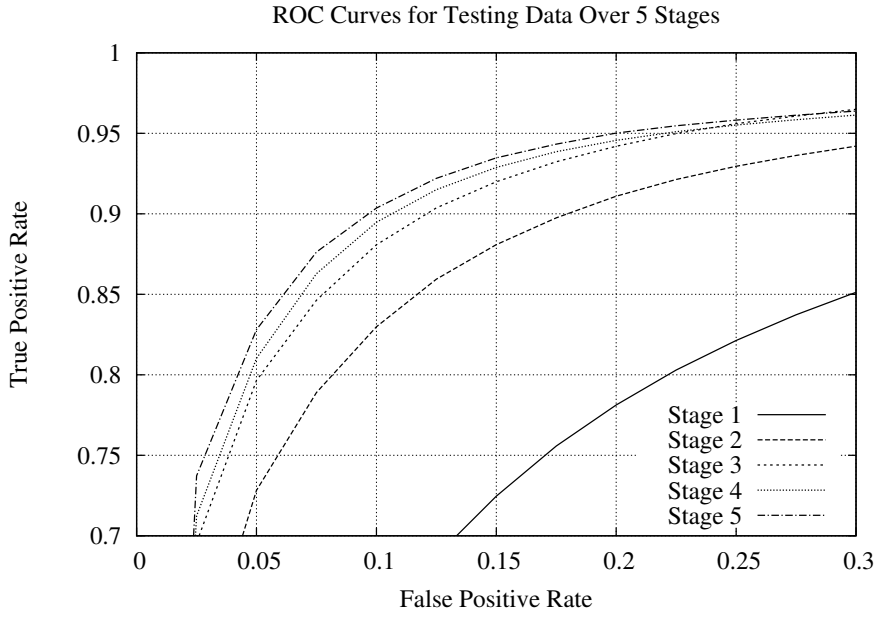


Figure 3.7. Selected images from the Weizmann horse database which were used as part of the testing set. The raw image is show in column (a). Output from the series of ANNs is shown in columns (b), (c), and (d), corresponding to stages 1, 2 and 5.

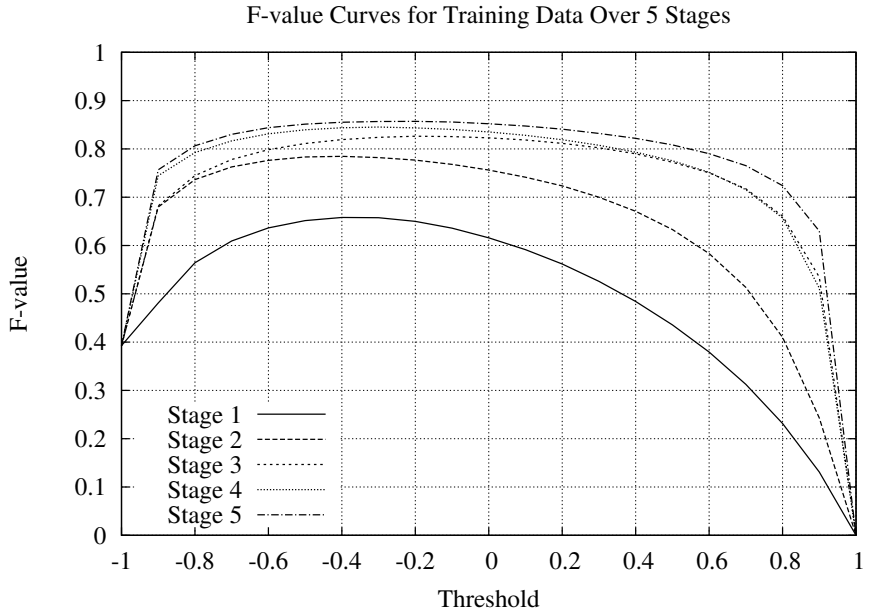


(a)

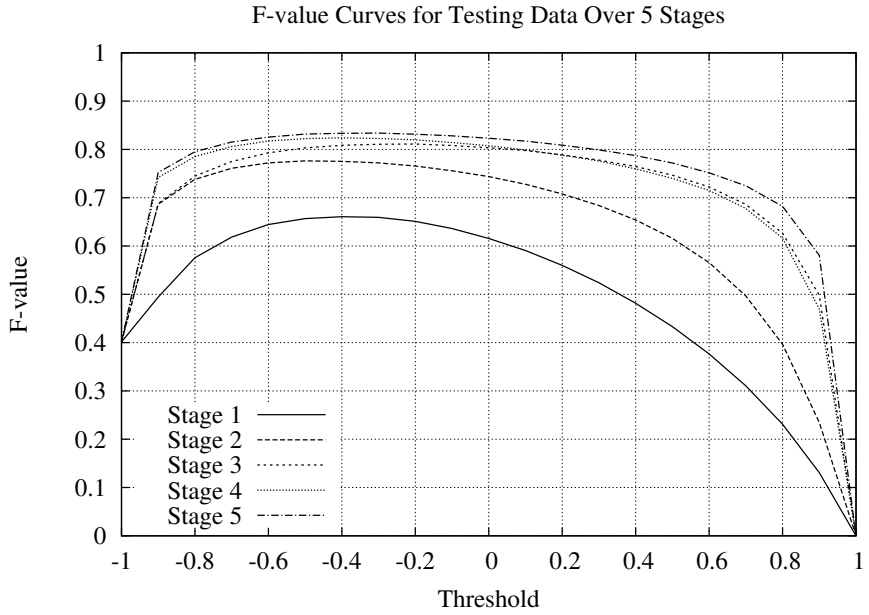


(b)

Figure 3.8. ROC curves for the (a) training data and (b) testing data for each stage of the network on the Weizmann horse dataset.



(a)



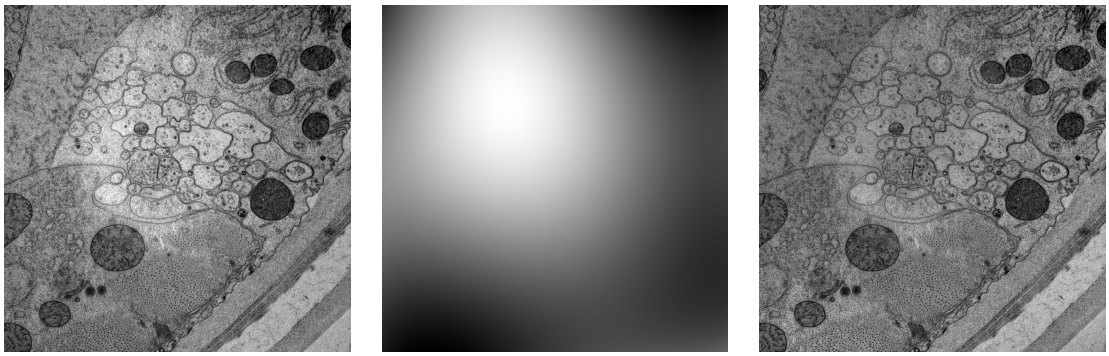
(b)

Figure 3.9. f-value curves for the (a) training data and (b) testing data for each stage of the network on the Weizmann horse dataset.

C. elegans data appear as intensity valleys; however, not all valleys in the data are neuron membranes, i.e., membranes of intracellular structures may also appear locally as valleys. The proposed method successfully learns the appropriate subset of ridges that need to be identified as neuron membranes as will be described in Section 3.5.2.1. The rabbit retina data has a resolution of 2nm×2nm×80nm and each 2D section is 7629×7351 pixels. Unlike the *C. elegans* dataset, neuron membranes in the retina data generally appear as intensity edges. Owing to the flexibility offered by the use of stencils rather than a predefined filter bank, the proposed method is also successful in learning to detect neuron membranes in this dataset as will be discussed in Section 3.5.2.2.

Before discussing detailed results of experiments on the two datasets, we will outline the common experimental details. First, the setup for these data sets used 5 ANNs in series. Additional networks could be included; however, for these datasets, the performance converges to a limit (Figure 3.6) and improvement in membrane detection is minimal. Each ANN used in the experiments contained one hidden layer with 20 nodes. We experimented with more layers and different numbers of nodes but did not find significant advantages. It is important that the number of nodes be large enough to approximate a non linear boundary and small enough that the ANN does not overfit to the training data [20, 39], as discussed in Section 3.3.1. Results using 10, 20, and 30 nodes turned out to be somewhat similar. Given the time versus performance trade off, 20 nodes was most appropriate. The networks were trained using backpropagation with a step size of 0.0001 and momentum term of 0.5. Early stopping was used as the criterion to determine when to terminate training [37, 81]. This means that a small portion of the training data (20% in this case), called the validation set, is used only to test the classifier generalization performance. The training terminates when the lowest error on the validation set is attained. To mitigate problems with local minima, each network is trained for 5 Monte Carlo simulations using randomly initialized weights.

Preprocessing can be performed for images to improve the contrast both globally and locally across the images. First nonuniform illumination correction can be applied to each image to reduce the variations in brightness across the image [98]. An example of this process, which is optional and only needed on the *C. elegans* data, is shown in Figure 3.10. Then a contrast limited adaptive histogram equalization (CLAHE) [78] filter is applied to locally adjust the contrast. This enhances the contrast of the membranes. A slope of 3 and window sizes of 64×64 and 256×256 were used for the *C. elegans* and retina datasets, respectively.



(a) (b) (c)

Figure 3.10. Example illumination correction for the *C. elegans* data. (a) is the original image, (b) is the estimated illumination, and (c) is the corrected image.

Each image used in the experiments was annotated by an expert who carefully marked neuron membranes with a one pixel wide contour. This contour was dilated using a disk shaped kernel with a radius of 2 pixels, ensuring that the positive training examples cover all of the actual membrane pixels. The negative training examples were selected as the remaining pixels in the image, after erosion to remove training pixels that are very close to the membranes. This strategy leaves a thin layer of pixels between the positive and negative training example pixels that are not used for training purposes. This ensures that the network learns on pixels that are either membrane or nonmembrane, excluding those that are more prone to labeling errors.

Finally, to optimize network performance, the total number of training examples from each image includes all of the positive examples and a random selection of negative examples such that there are twice the number of negative examples, than positive. Choosing the optimal number of training examples was difficult given there were many more negative than positive examples in this dataset. If all the negative training examples are used then the ANNs are biased towards classifying pixels as nonmembrane. After conducting a series of experiments for considering the results from different ratios of positive and negative examples and the training times, the 2:1 ratio resulted in the best segmentation while achieving a reduced training time. Using all the training data (and other increased ratios, such as 4:1) produced a similar ROC curve but results were biased towards false negatives. Clearly, the threshold could have been adjusted in the final stage to obtain the best results. Alternatively, one possible solution for this problem would be the use of a weighting factor

chosen to obtain unbiased training. However, either approach would have much slower training than the previous described strategy without improving the overall segmentation. For each pixel in the training data, a stencil with a radius of 5 (or $n = 11$ in Equation 3.3) is used to sample the image data and form the feature vector.

3.5.2.1 *C. elegans* Ventral Nerve Cord

The nematode *C. elegans* is an important organism for neural circuit reconstruction because it is the only organism for which the connectivity has been determined [110, 115]. Nevertheless, there are still numerous questions that require the determination of the connectivity, such as how genes regulate wiring [47] or how connectivity is altered to mediate different behaviors, for example, between males and females [116]. In addition, reconstructions of the full nervous system reveal topological characteristics of the neurons that are important for studies of neuronal functions. The particular dataset used in this paper is from the ventral nerve cord of the *C. elegans* and is important for studying the interwoven topology of neurons making connections to local targets.

To validate the robustness of the method, five-fold cross validation was used on a set of 50 annotated images, separated into 5 groups of 10 images in each. The network was trained on each fold according to the procedure described in Section 3.5.2, and tested on the remaining four. The improvement in the classification after each ANN in the series is visible in the classification of the training data after each stage, shown in Figure 3.5, and in the receiver operating characteristic (ROC) curves in Figure 3.6. The output from the network improves quantitatively and qualitatively with each network in the series. Directly sampling the image using a stencil and repeated uses of the network enables the method to accurately estimate the appearance of membrane pixels and pixels in surrounding neighborhoods.

Figure 3.11(a) shows four sections from the *C. elegans* dataset. The final membrane detection with the proposed method is shown in Figure 3.11(e). Note that these are testing results; that is, these four sections were not used as training data. To demonstrate the advantages of the proposed method, two other methods are presented. The first method, shown in Figure 3.11(b), performs thresholding after enhancing the membranes with anisotropic directional smoothing [55]. Figure 3.11(c) shows results from an approach similar to the approach in Mishchenko [71], which learns boundary confidences using Hessian eigenvalues as input to a single layer neural network. It can be seen that the proposed method removes a substantially larger percentage of the intracellular structures from the

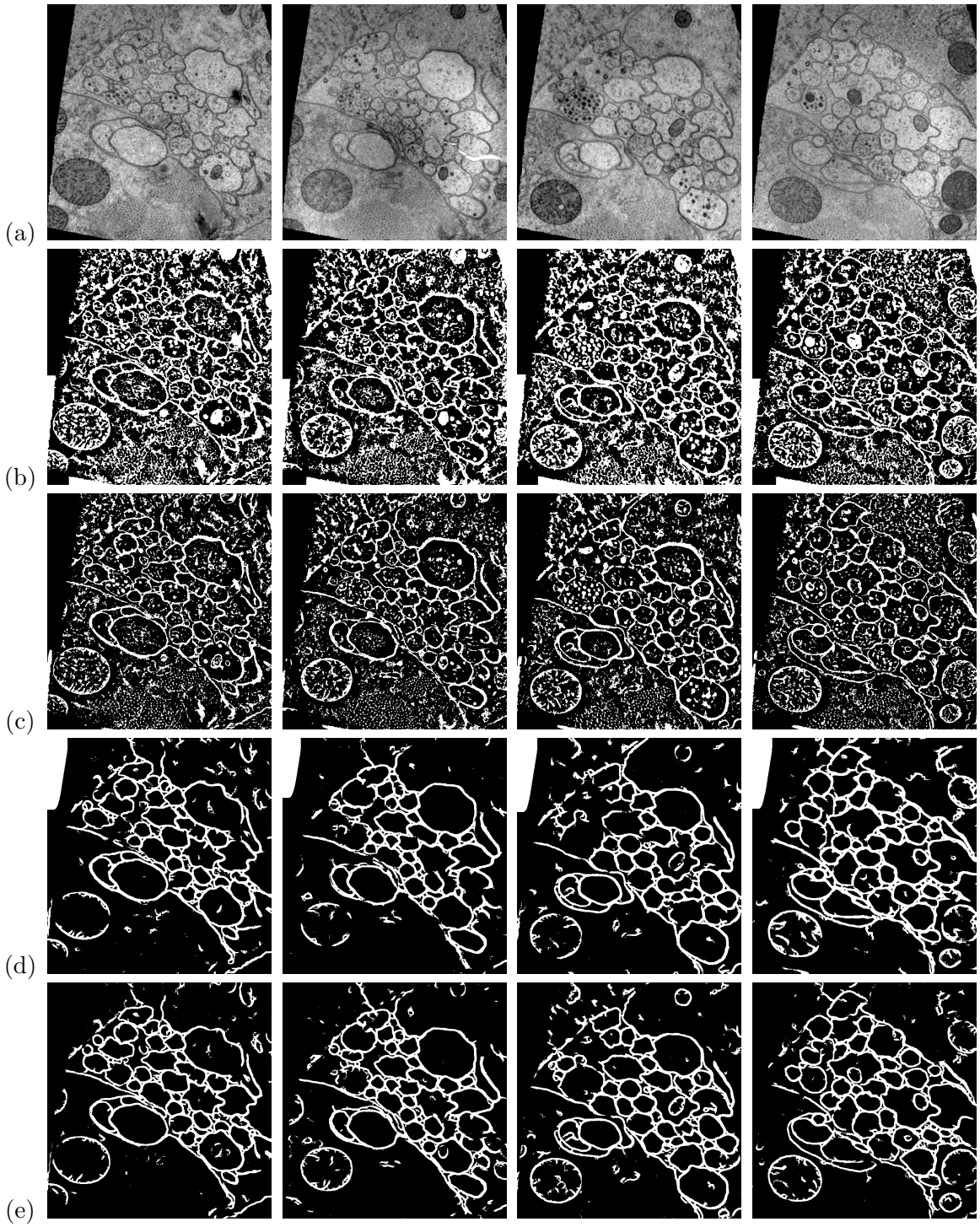


Figure 3.11. Demonstration of three membrane detection techniques.(a) Cross sections of the nematode *C. elegans* acquired using EM. (b) intensity thresholding after directional anisotropic smoothing [55], (c) thresholded boundary confidences from a single ANN trained using Hessian eigenvalues [71], (d) membrane detection from serial ANNs, trained using membrane filter banks, and (e) membrane detection from serial ANNs, trained using image data sampled from stencils.

detection results as well as providing better membrane continuity. It is important to note that in Mishchenko [71] further postprocessing is performed to interpolate between broken boundaries and complete contours, resulting in an improved result compared to the one shown here. However, we compare against only the single layer network part of that method since the goal is to demonstrate the improvement achieved by the use of ANNs and auto-context. Of course, the same post processing methods could be applied to the results of the proposed method as well. Figure 3.12 shows enlarged regions demonstrating the removal of large intracellular structures and closing of weak membranes.

To demonstrate the advantages of directly sampling the image with a stencil, We also tested the proposed auto-context ANN strategy but with inputs to the ANNs that are derived from a line detection filter bank rather than sampling the image. Three feature detection filters were constructed to generate responses for the different types of membranes. The first and second filters are both bars, one to detect membranes and the other to detect membrane junctions. The width of the bar approximates the width of the membrane, which in this case is about 5-7 pixels wide. To detect membranes at different angles, each filter is rotated between 0 and 180 by 20°. The third filter is a simple vesicle detection filter which helps the network learn pixels it should not classify as membranes, i.e., for rejecting vesicles. It is constructed as a circle with an off center surround ranging in radius depending on the size of the vesicles in the images. For this data, the radius varies between 3 and 5 pixels. The circle detection filters were included to help the auto-context ANN learn to remove vesicles from the membrane detection results. Each filter is convolved with each pixel in the image: $I_i = I * F_i$, where I is the input image and F_i is the ith filter convolved with I. The complete filter bank, F, contains the membrane and junction filters, at rotational increments of 20°, and several scales of vesicle filters, for a total of 32 filters. In total, we used a filter bank that consists of a set of 32 line detection filters oriented at different angles and 5 circle detection filters with different radii.

Figure 3.11(d) is the output obtained with the filter bank/series ANN approach. While these results are better than the results in Figure 3.11(b) and (c), they contain more false positives than the results of the full stencil/auto-context ANN approach shown in Figure 3.11(e). The advantages of using the stencil becomes clearer in a quantitative comparison as discussed in the next paragraph. Furthermore, an important practical advantage of using the stencil is that it does not require any a priori knowledge. Therefore, it can be trained to detect different structures as will be shown for the retina dataset in

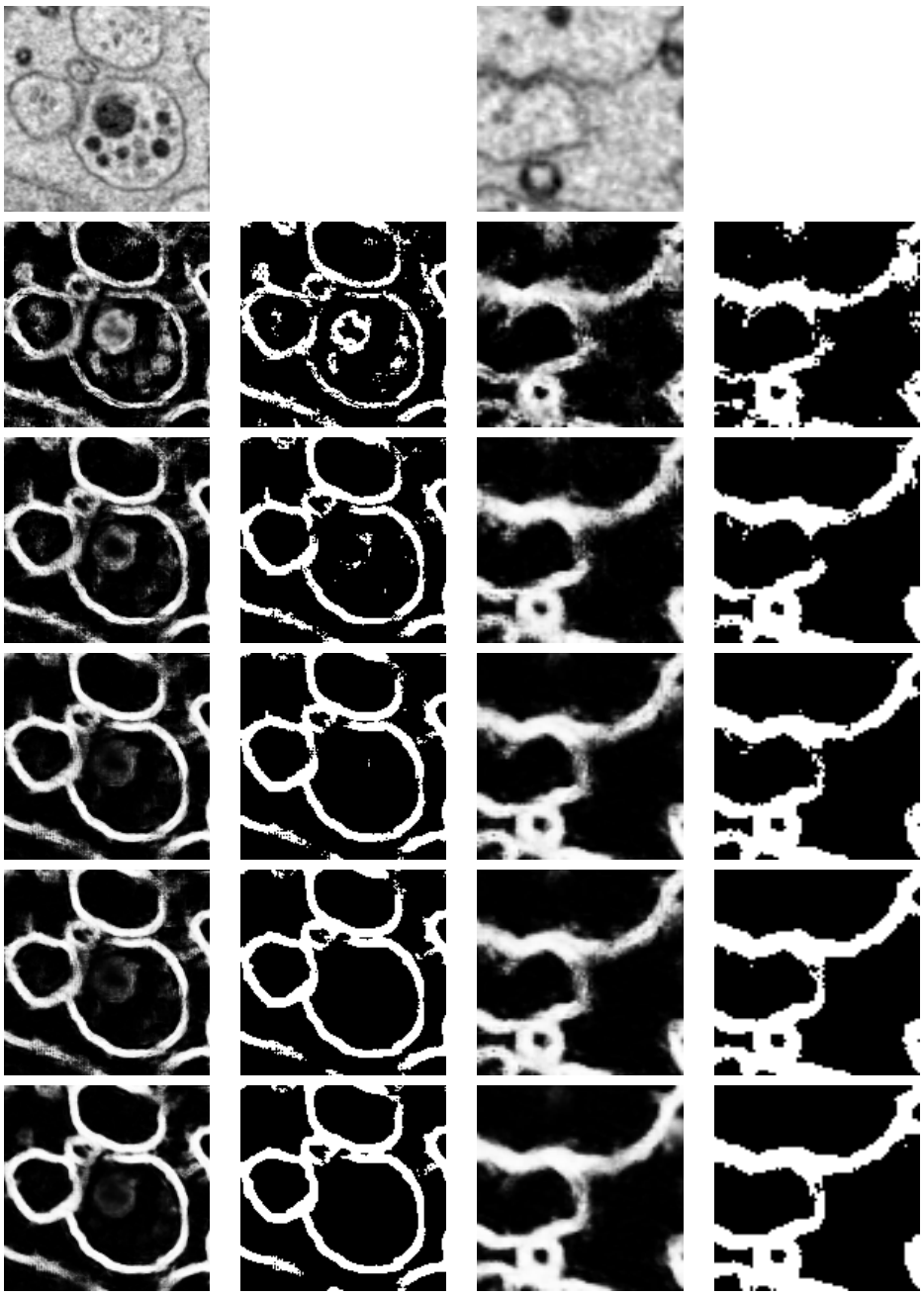


Figure 3.12. Examples demonstrating how the proposed method removes intracellular structures (left two columns) and closes gaps in a weak membrane (right two columns). The top row is the original image, columns (a) and (c) show the classifier output, and columns (b) and (d) show the final thresholded segmentation.

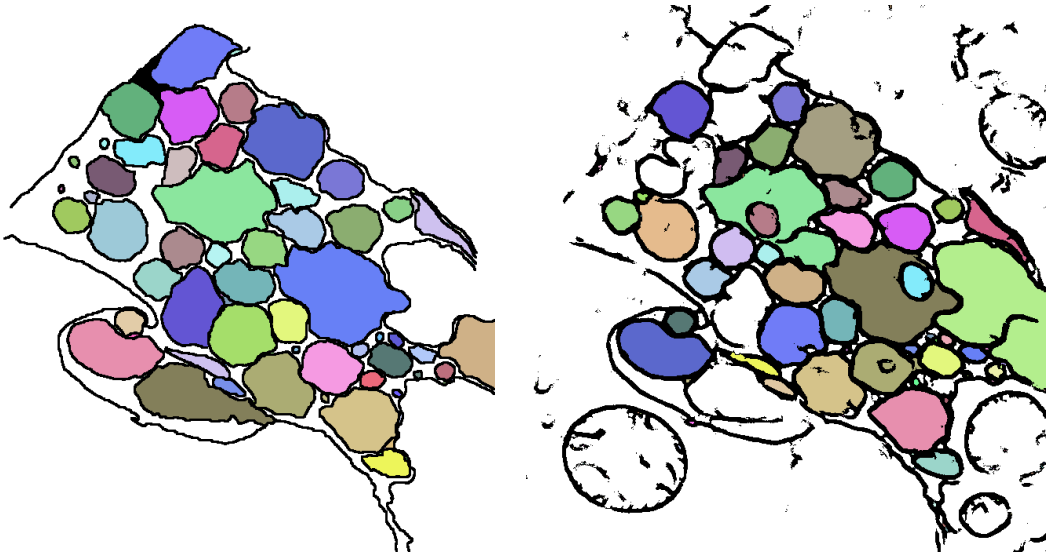
Section 3.5.2.2. In comparison, a filter bank designed to capture the relevant structures for the *C. elegans* dataset is not expected to capture the relevant structures in a different dataset which necessitates the design of a new filter bank.

Figure 3.13 shows the actual neruon region segmentation with a flood fill on the output from the membrane detection. Figures 3.14 and 3.15 compare the final ROC curves for each method from Figure 3.11. For this particular data, a single layer ANN using Hessian eigenvalues as inputs (labeled "Hessian") demonstrates no quantitative differences from thresholding after directional anisotropic diffusion (labeled "Jurrus et al."). These ROC curves correspond to the qualitative results in Figure 3.11(b) and (c), respectively. The other three curves all show a large improvement in performance. The use of membrane detection filters (labeled "Filters") demonstrates how a carefully chosen set of features can be used for learning to detect membranes. Image patches (labeled "Patches") are just as successful in training to detect membranes as filters. However, in testing, the patches outperform the filters. We argue that this is due to the fact that raw image data, in the patches, generalize better to new data. sample the image directly and give more flexibility to the classifier than a filter bank. Using a stencil (labeled "Stencil") results in the best performance. The stencil provides the classifier with two important features. First, similar to patches, it trains the classifier on image sample directly, as opposed to a fixed representation as obtained from the filters. Second, it samples a larger area than the patches, while maintaining the same number of features (as seen in Figure 3.2). The latter feature is very helpful in practice since it ensures improved performance without sacrificing the network training time (actually, in these experiments, using the stencil improved the training reliability and time).

3.5.2.2 Rabbit Retina

The retina is a complex structure containing several layers of neurons. Processing light sets off a series of chemical events and connections among these neurons that scientists would like to model. Most importantly, scientists would like to characterize neural circuitry that is damaged and in a diseased state. However, unraveling the connective patterns in this complex tissue is an enormous task.

To demonstrate the robustness of this method on a very different dataset, an expert hand segmented all of the bipolar, amacrine, and horizontal cells in a single 2D section through the retina. This section is 7629×7351 pixels and contains approximately 500 neurons. The image was divided into four equal sections and a four fold cross validation technique is used



(a) (b)

Figure 3.13. Segmentation of neurons using a flood fill on the image of detected membranes. (a) Ground truth and (b) membranes detected with proposed method.

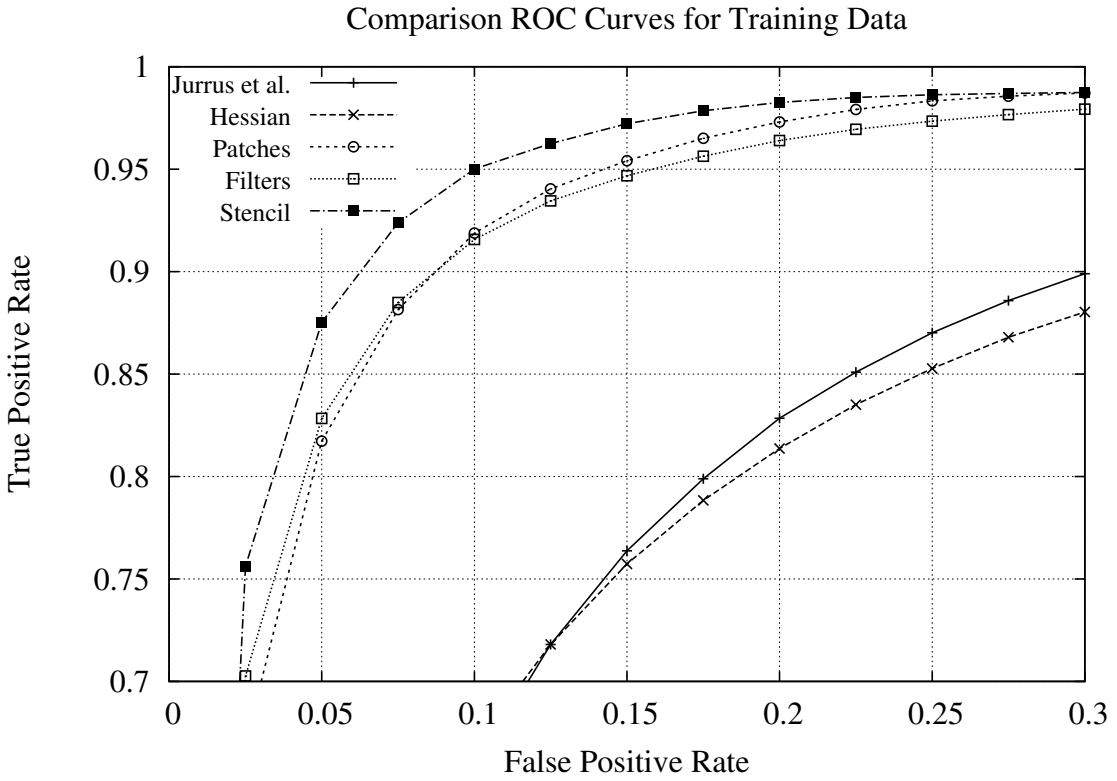


Figure 3.14. ROC curves for the *C. elegans* training data. "Jurrus et al.": thresholding after directional anisotropic smoothing [55]. "Hessian": single layer neural network operating on Hessian eigenvalues similar to Mishchenko [71]. The remaining three curves demonstrate the results from different inputs to the proposed auto-context ANN approach.

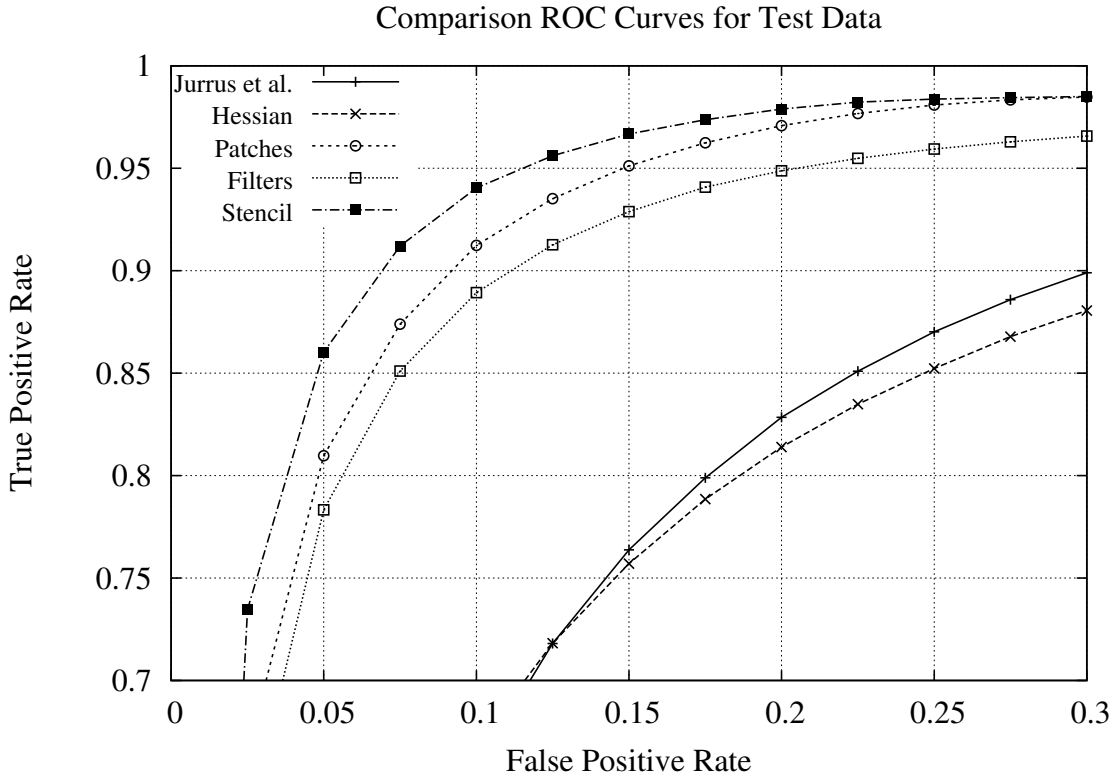


Figure 3.15. ROC curves for the *C. elegans* testing data. "Jurrus et al.": thresholding after directional anisotropic smoothing [55]. "Hessian": single layer neural network operating on Hessian eigenvalues similar to Mishchenko [71]. The remaining three curves demonstrate the results from different inputs to the proposed auto-context ANN approach.

to assess the performance of the algorithm.

Figure 3.16 shows the output on the test data. Figure 3.16(a) shows portions of the TEM image, cropped to show the cellular details. Figure 3.16(b) is a simple baseline membrane detection obtained by thresholding the intensity gradient after smoothing the input image with a Gaussian kernel (standard deviation 3 pixels). Thresholding the gradient results in some obvious problems. Differences in contrast and the presence of intracellular features make isolation of the neuron edges difficult. Figure 3.16(c) shows the results of applying the series of ANN method with a filter bank as input. For this data, 25 Leung-Malik edge filters [62] were used. The Leung-Malik filter banks consists of first derivatives of a Gaussian kernel (standard deviation 3 pixels) at various orientations. The results in Figure 3.16(d) are from the stencil/serial ANN approach identical in architecture to the one used for the *C. elegans* dataset. From a qualitative perspective, the stencil removes more intracellular structures and is more robust to changes in contrast. When the weights from the final network, which was trained with a filter bank, are applied to the testing images, the edge detection performs poorly. However, sampling the image using a stencil is a more robust way to detect membrane edges and provides more consistent results across images. Figure 3.17 gives clear examples of how this method removes nonmembrane structures and closes complex gaps resulting from inconsistent membrane data. Most importantly, the results from this dataset demonstrate the flexibility of the method on different feature types. The feature vectors for both dataset are the same, that is, they are simply raw image values sampled from the input data.

Figures 3.18 and 3.19 show the quantitative comparison for the methods demonstrated in Figure 3.16. The gradient magnitude provides a baseline for how well a simple edge detection method can be expected to perform. While it detects many of the neuron boundaries, it also has a lot of false positive responses for internal structures and fails to close gaps in weak parts of the membranes. The serial networks provide a very large improvement over this simple method when a filter bank is in place. However, the proposed stencil/auto-context ANN method is demonstrated to do still a significantly better job at detecting boundaries than the filters.

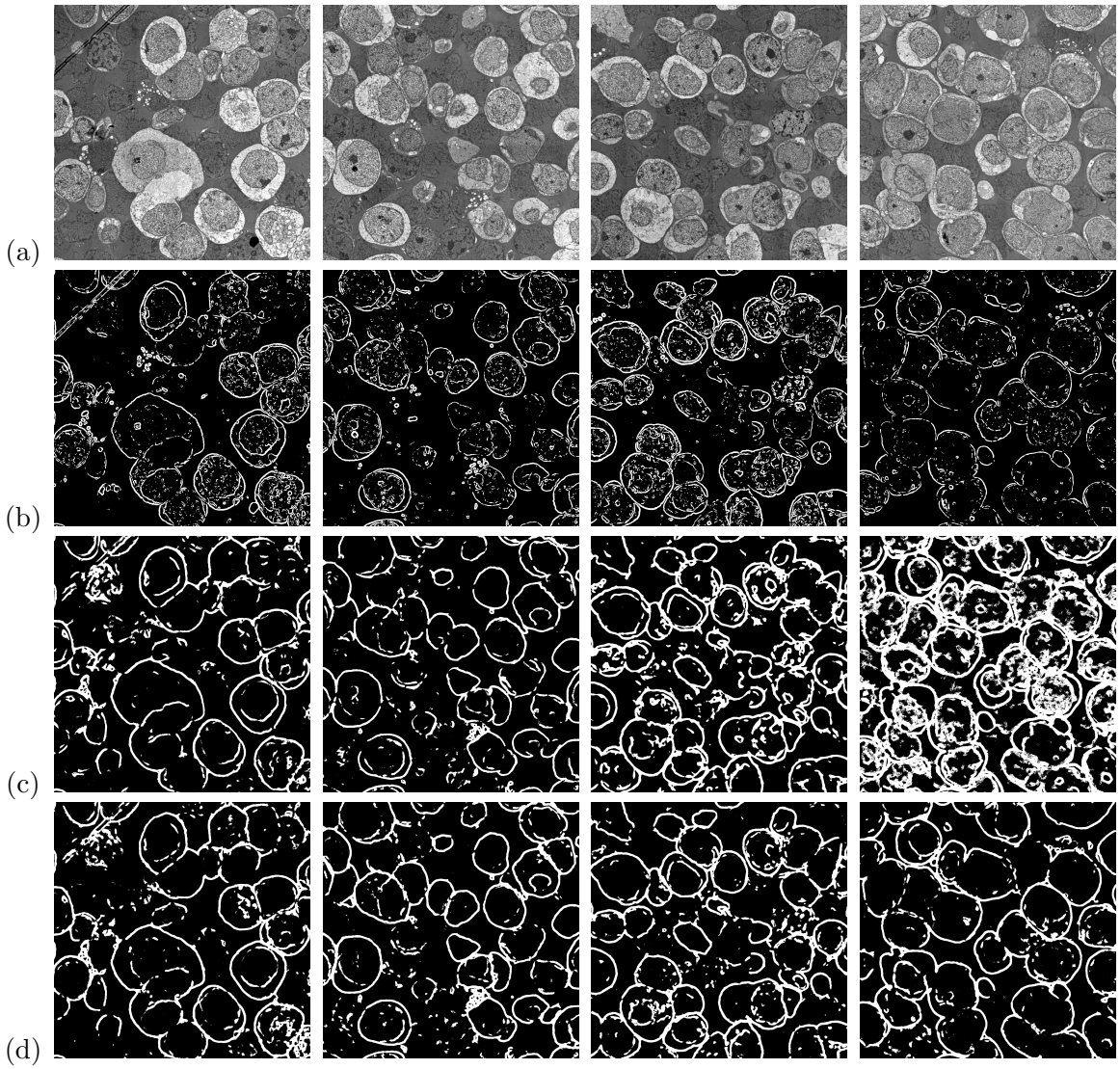


Figure 3.16. Membrane detection of the rabbit retina. (a) Original TEM images from a rabbit retina. Membrane detection with: (b) thresholding on the gradient magnitude, (c) serial ANNs using the output of an edge detection filter bank, and (d) serial ANNs using image intensities sampled from a stencil.

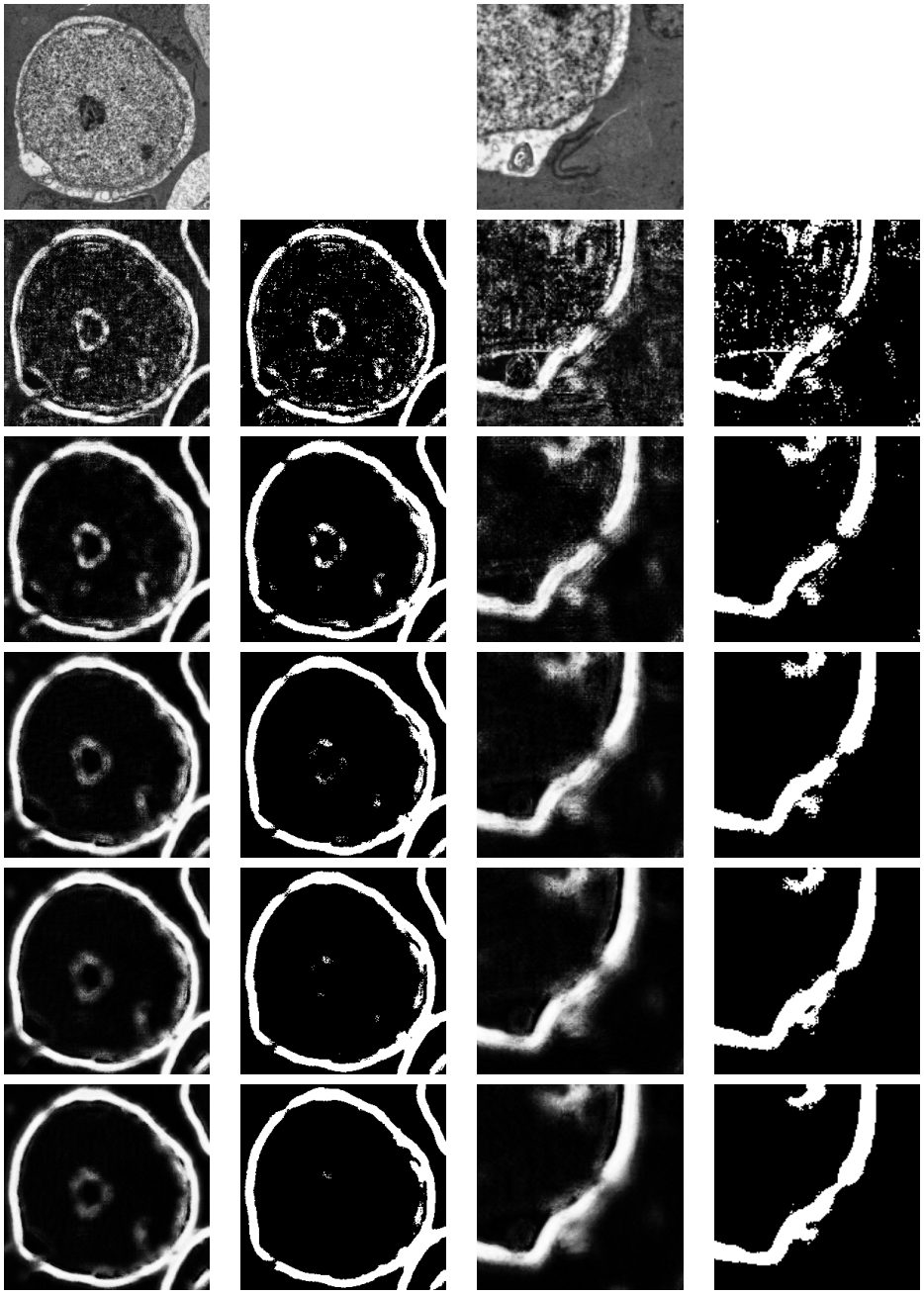


Figure 3.17. Examples of locations in the data where intracellular structures are removed (left two columns) and gaps in membranes are closed (right two columns). The top row shows the raw images, columns (a) and (c) show the classifier output, and columns (b) and (d) are the final thresholded segmentations.

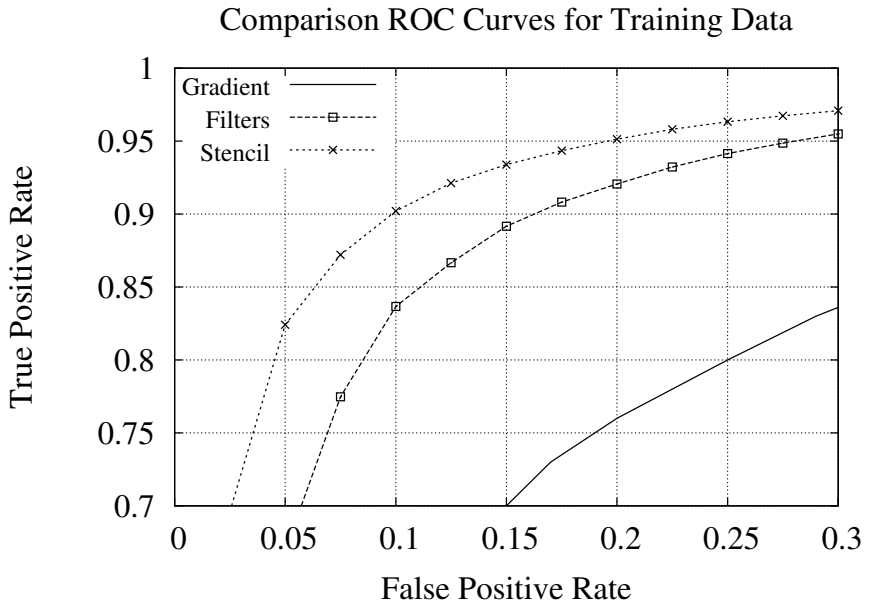


Figure 3.18. ROC curves computed on the retina training data. For comparison, ROC curves are included for other methods. "Gradient" shows the best membrane detection when the gradient magnitude is thresholded (shown in Figure 3.16b). "Filters" is the use of the Leung-Malik filter bank in training the series of ANNs (Shown in Figure 3.16c).

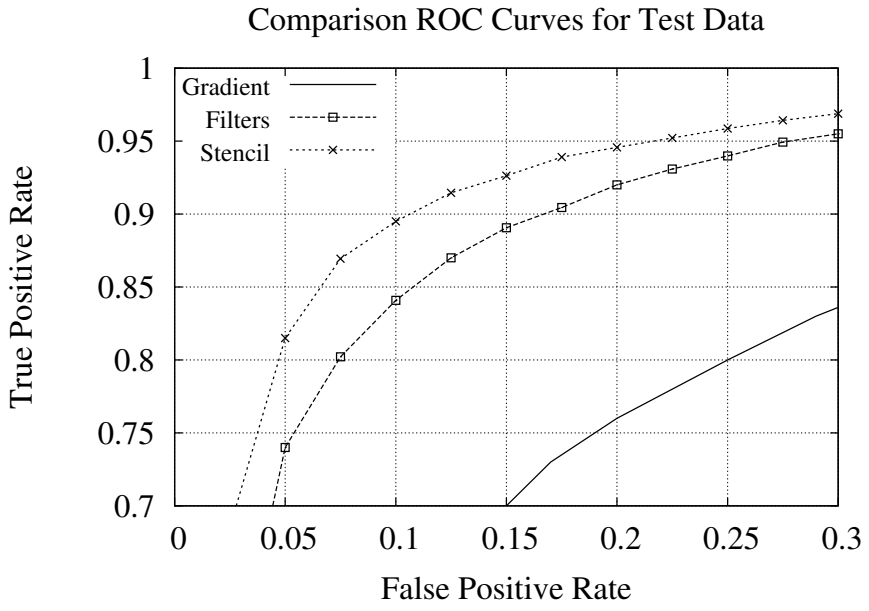


Figure 3.19. ROC curves computed on the retina testing data. For comparison, ROC curves are included for other methods. "Gradient" shows the best membrane detection when the gradient magnitude is thresholded (shown in Figure 3.16b). "Filters" is the use of the Leung-Malik filter bank in training the series of ANNs (Shown in Figure 3.16c).

CHAPTER 4

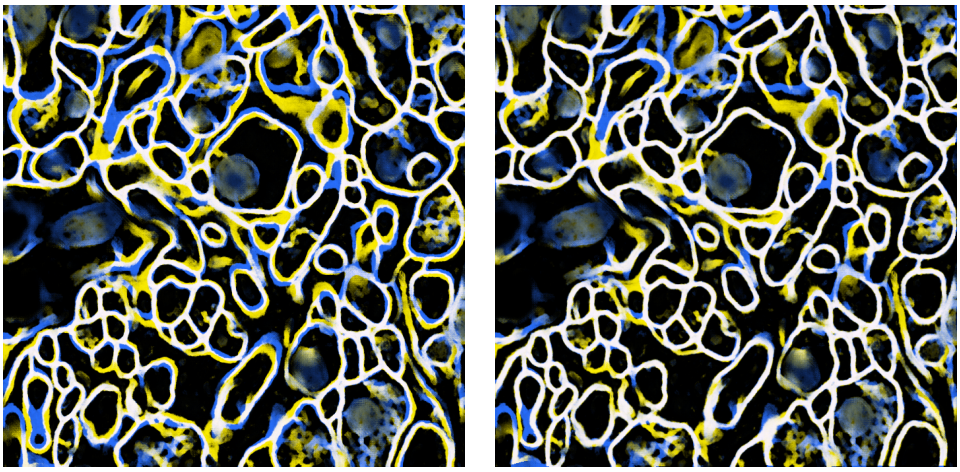
3D MEMBRANE DETECTION AND POSTPROCESSING

4.1 Introduction

Chapter 3 presented an advanced method for segmenting individual neurons in 2D using a series of artificial neural networks. In each EM image, membranes are detected and a segmentation algorithm is applied to fully segment each neuron region. To improve upon this algorithm, an additional set of methods are applied to the output of the serial ANN to improve the detection of membranes, thus, resulting in better segmentations of each 2D neuron.

4.2 Sequential Section Artificial Neural Network

Sequential sections from EM data often contain similar structures that have the potential to improve the quality of the 2D segmentation. One way to do this is with a stencil that spans multiple sections. However, the membrane locations between sections have poor correspondence. This is partly because of the anisotropic nature of the data, which often results in large movement of membranes between sections, and membranes sometimes do not run perpendicular to the cutting plane causing membranes to have low contrast and appear fuzzy. The differences between two sections is seen in Figure 4.1(a) which shows two sequential images with detected membranes overlaid with each other. Membranes in sequential sections are near each other, but they do not correspond well enough to use them directly in a 3D stencil that would span multiple sections. To correct for this, we propose a novel approach which aligns sequential membrane probability map images using a correlation based nonlinear registration. We prefer to register only the membrane probability images because the classification process has removed many of the internal structures that would make an extremely fine scale nonlinear registration on raw image data difficult. Once registered, a 3D stencil that spans 3 adjacent sections samples the



(a) (b)

Figure 4.1. Two sequential sections from the mouse neuropil with membranes detected after the serial ANN overlaid with each other with (a) no registration and (b) after the intensity based nonlinear registration. Blue and yellow colors indicate membrane overlay mismatches and white indicates shared membranes.

classification results from the previous stage and provides information to be used in the final classification step.

More specifically, after the membrane detection is complete for each section using the serial ANN architecture, images are registered to each other and used as input for the ANN. The registration method proposed is a B-spline deformable registration [40]. Given an image to be registered and a static template image, a nonlinear deformation can be generated which minimizes the mean squared difference energy, given by,

$$\int_{\Omega} \left(C^M \circ t(x) - C^S(x)\right)^2 dx. \tag{4.1}$$

where Ω is the image domain and $t(x)$ is the deformation $\mathbb{R}^2 \rightarrow \mathbb{R}^2$, in this case given by a 2D tensor product B-spline transform of order 2 [85]. C^M is the moving classification image, and C^s is the static classification image. For the case presented here, C_i is the static image and C_{i-1} and C_{i+1} are the moving images. The serial ANN with the proposed registration step is depicted in Figure 4.2.

Each section has its own set of neighboring registered sections. The change in the membrane locations after two images are registered is shown in Figure 4.1(b). Now that membranes are more carefully aligned across neighboring sections, a new stencil can be used to sample the 3D space. The 3D, three section, stencil is similar to the one shown

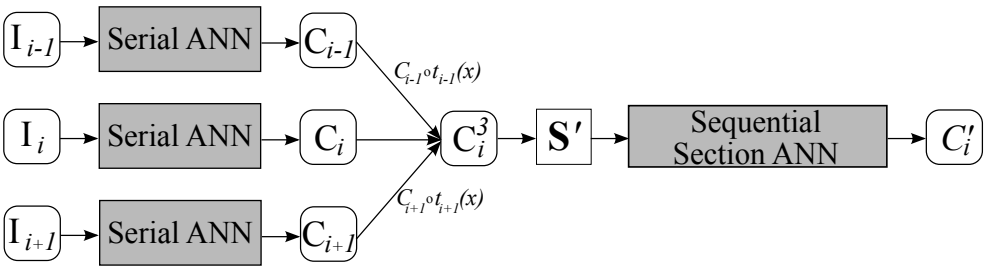


Figure 4.2. Diagram demonstrating the flow of data for the sequential section ANN architecture. I_i are the input images, $SerialANN$ is the diagram in Figure 3.4 collapsed, and C_i is the output of the classifier on image I_i. $C_{i-1} \circ t_{i-1}(x)$ is the registration of C_{i-1} to C_i and $C_{i+1} \circ t_{i+1}(x)$ is the registration of C_{i+1} to C_i. C_i^3 is the stack of all three registered images. S' is the 3D stencil used on the combined images as input to the classifier. C_i' is the final classification.

previously in Figure 3.3. This stencil is used on the middle slice, while the stencil on the top and bottom slice have a shorter radius. The output from the ANN using this stencil is show in Figure 3.5(e). Using information from the sequential sections, the ANN learns to identify membranes in C_i that were not previously detected, because the membranes were detected in C_{i-1} and C_{i+1} improving the overall segmentation. This helps specifically in cases where C_i contains grazed membranes, but C_{i-1} and C_{i+1} do not. A good example of this is shown in Figure 4.4, second column. Membranes in the raw image appear fuzzy and are not well detected after the serial ANN. Using information from the registered sequential sections strengthens these boundaries. In this way, the ANN also learns the possible shapes of membranes across several sections.

4.3 Tensor Voting

Tensor voting [69] is a method for detecting salient curves and surfaces in low level vision. In 2D, a 2×2 tensor can be used to represent local curviness and direction information. For instance, a pixel on a neuron membrane is part of a curve; therefore, ideally it is represented with a tensor having one non zero and one zero eigenvalue. In practice, such a tensor, also known as a stick tensor, will have one large and one small eigenvalue. In tensor voting, each tensor casts a vote in a regional area around itself where the voting field is determined by the orientation and the stick-ness of the tensor as determined by the ratio of the eigenvalues. Tensor voting strengthens salient curvilinear structures, while removing noise and blobby artifacts. Since this is a computationally intensive method, it is typically applied in a sparse

manner such that only a subset of the pixels in the image where curve features are detected are allowed to cast votes. However, in problems where detection is hard, such as with neuron membranes, it can be advantageous to allow every pixel to cast votes proportional to its strength as determined by the classifier and to postpone detection until after this step. To achieve this in a computationally efficient manner a rapid tensor voting algorithm was used which uses steerable filters as a basis for the voting field [32, 61]. In this algorithm, a set of basis voting fields are convolved with the image and then linearly combined to form the desired voting field at each pixel.

4.4 Results

Two EM datasets are segmented using the proposed improved 2D membrane detection methods. The nematode *C. elegans* as discussed in Section 3.5.2.1, is used again as a test dataset for membrane detection. However, this dataset is a more recent acquisition, attempting to section a much longer portion of the VNC. As a result, the size of the data is much larger with a resolution of 6nm×6nm×33nm and each 2D section is 4008×2672 pixels. The second dataset is a stack of 400 sections from the mouse neuropil, with a pixel resolution of 10nm×10nm×50nm and each 2D section is 4096×4096 pixels. Note that the membranes in each image are very different, shown in Figure 1.2. The section from the *C. elegans* nerve cord, Figure 1.2(a), has a low signal to noise ratio and the neuron membranes have varying thickness and contrast. In comparison, the membranes from the mouse neuropil, in Figure 1.2(b), are strong in contrast and have a high signal to noise ratio, but contain more variable internal structures. Both datasets contain grazed membranes, corresponding to neurons cut at non perpendicular angles. This makes it difficult for even the human eye to identify all the membrane structures. More traditional statistical based machine learning methods would require a specific filter design for each dataset. However, the use of stencils rather than a predefined filter bank means the proposed method can adapt to the idiosyncrasies of different samples, and is successful in learning to detect neuron membranes in both datasets.

4.4.1 The *C. elegans* Ventral Nerve Cord

To segment the membranes in this dataset and create a 3D reconstruction, first all the ssTEM images had to be aligned to form a volume. We performed a ridged alignment using a brute force search for the unknown rotation and translation between adjacent pairs of sections [99]. This was a challenging task because there are significant changes between

the sections resulting from slicing artifacts and missing sections. Approximately 10% of the images required hand alignments. We did not perform a nonlinear alignment on these sections because we wanted to maintain the shape of the neurons and prevent distortion.

For validation, experts segmented 40 selected images from the first 400 sections. Each expert placed a one pixel wide line along the membranes of the neurons, which was dilated using a 5 pixel wide structuring element, to cover most of the membrane pixels. Before training, Gaussian blurring was performed on the EM images with a small $\sigma = 2$ to remove noise, and down sampled by 2 to reduce the computational complexity. Then a contrast limited adaptive histogram equalization (CLAHE) [78] filter was used to enhance the contrast in the neuron membranes. For the training data, 30 images were randomly selected for training data and the remaining 10 were used for validation. From those images, 1 million samples were randomly selected from the manually marked images. Because of the relatively small percentage of positive examples (representing membrane pixels), the training data were balanced to contain $\frac{1}{3}$ positive and $\frac{2}{3}$ negative examples. The stencil used to sample the image values had a radius of 10 and was similar to the one in Figure 3.3. The ANN used had one hidden layer of 20 nodes. To mitigate problems with local minima in training, each network was trained for 5 Monte Carlo simulations using randomly initialized weights. Each stage of the serial ANN took between 9 and 12 hours to complete. The ANN for the sequential sections took about 22 hours to train. Finally the tensor voting, which was implemented in Matlab, was completed in 6 minutes.

To segment the neuron regions in these images, a user corrected the segmentation and a flood fill was applied. The watershed applied to the blurred output of the tensor voting failed to provide reasonable segmentations because the blurring caused small, but necessary, features to be lost. This is because there is a trade off between the ability to close large gaps and the ability to segment smaller features. This trade off is controlled by the parameter sigma. Large sigma enables the watershed to close large gaps but also loses the ability to segment smaller features. The corrected data from this segmentation is used later, in Section 5.4.

Figure 4.3(a) shows three randomly selected sections from the *C. elegans* dataset. The final membrane detection with the proposed method is shown in Figure 4.3(e). The sequential section ANN is using information about membranes also detected in neighboring sections to improve the current segmentation. The tensor voting uses, as input, the final classification, closing remaining gaps. This is demonstrated in closer detail in Figure 4.4.

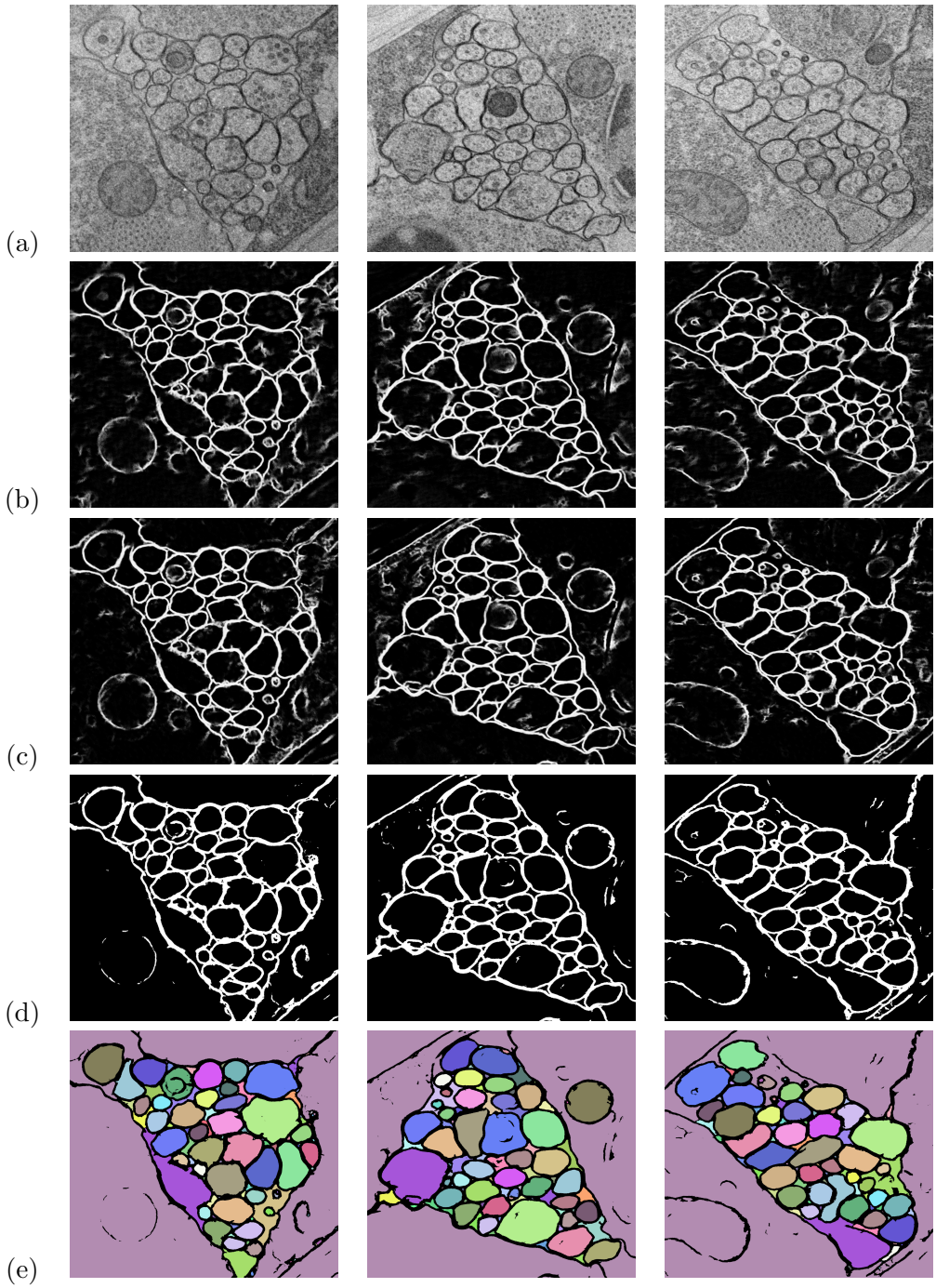


Figure 4.3. Output of the method on test images. (a) is the raw image, (b) is the output from the final stage of the series ANN (Section 3.3.3), (c) is the output from the sequential section ANN (Section 4.2), (d) is the output after tensor voting (Section 4.3), and (e) is the segmentations of the neuron regions from a flood fill. To evaluate the quality of the output in this example of the membrane detection, the user did not correct any of the segmentations.

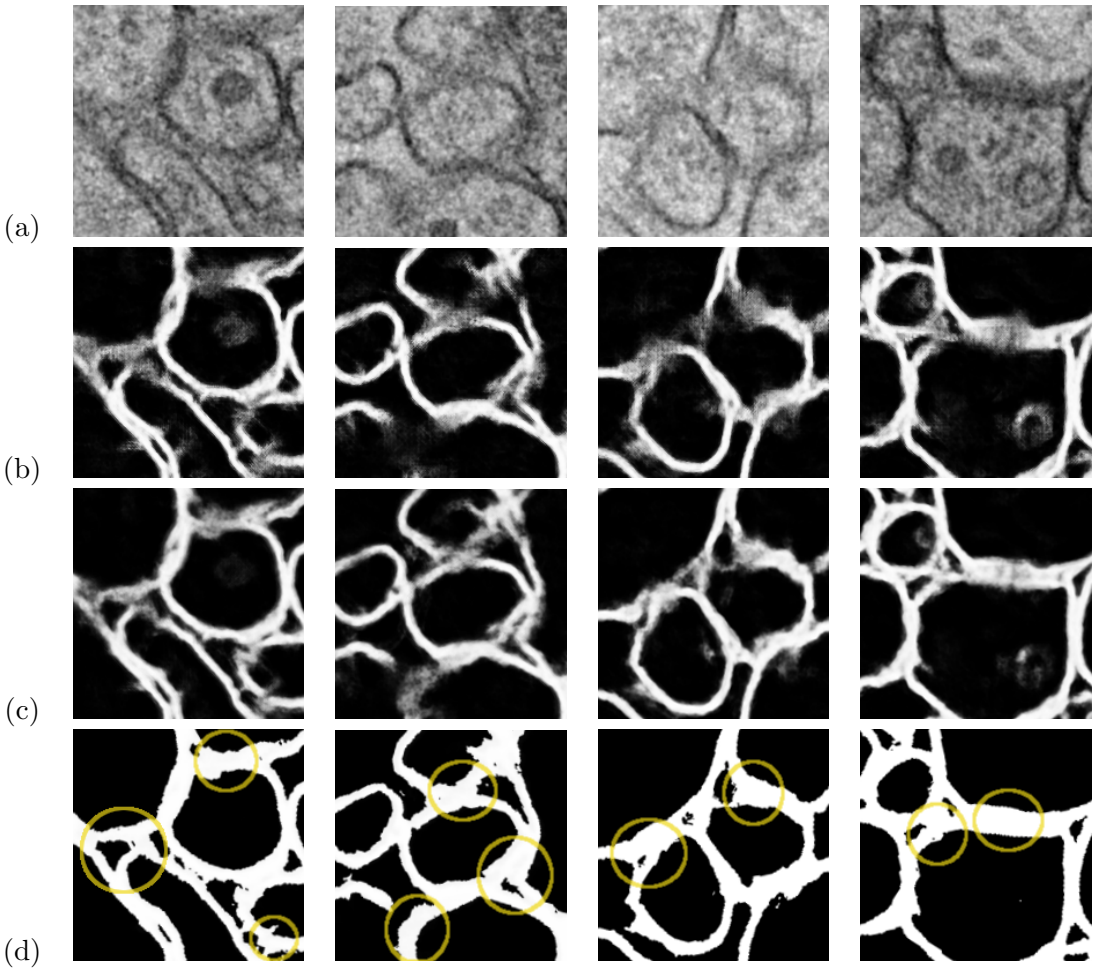


Figure 4.4. Example images demonstrating how this method strengthens undetected or grazed membranes and closes gaps on the *C. elegans* ventral nerve cord data. (a) is the raw image, (b) is the output from the final stage of the series ANN (Section 3.3.3), (c) is the output from the sequential section ANN (Section 4.2), and (d) is the final output after tensor voting (Section 4.3). Yellow circles highlight improved membrane detection and gap closing that results from these methods.

Improved membrane detection is annotated with yellow circles. Most often a strong membrane in a neighboring section provides confidence for enhancing membranes with poor contrast in the current section. Figure 4.5 gives a numerical evaluation of the improvement between the serial ANN and the sequential section ANN on the validated test images using ROC curves. While the impact from the gap closing using tensor voting is demonstrated qualitatively, its true positive and false positive values do not change as much. This is partly due to the dilation that results from the tensor voting and that the addition of pixels for closing is small compared to the overall segmentation.

4.4.2 The Mouse Neuropil

Understanding the connectivity, types of connections, and roles of different cells in the mouse neuropil is an increasingly more common area of study. The glial cell's roles in the neuopil are mostly unknown. Initially thought to be the support cells of the nervous system, evidence exists that they are active signaling participants [104]. Questions with regard to how the cells interface with the axon terminal, synaptic cleft, and dendrite still exist. The 3D organization of these structures provides insight into how the nervous system functions at very basic levels [113]. Most scientists focus on just 2 or 3 synapses at a time [21, 73], leading to different conclusions. The need to study large volumes and examine hundreds of synapses at a time is required to statistically validate these hypotheses. In an effort to help answer some of these questions, a segmentation of a large volume of the mouse neuropil is necessary.

The entire neuropil dataset is $4096 \times 4096 \times 400$. To train and validate the neural networks, a subset of this data ($700 \times 700 \times 70$) was manually segmented using Amira [41] by an expert. From that set, 14 images were randomly selected and used for training in the classifier. The training set contained 4.5 million examples. To decrease training time, the ANN was trained first on 1 million examples for 50 iterations. The weights from this network were used to initialize the ANN for the 4.5 million training examples. The ANN contained one hidden layer of 10 nodes. The images required no preprocessing to remove noise or enhance contrast and were sampled with a stencil of radius 10.

Figure 4.6 shows the segmentation results on three images from different sections. Figure 4.7 demonstrates in more detail the gap closing that occurs when the sequential section ANN and tensor voting are used. Figure 4.8 gives a quantitative comparison on the improvement of different methods.

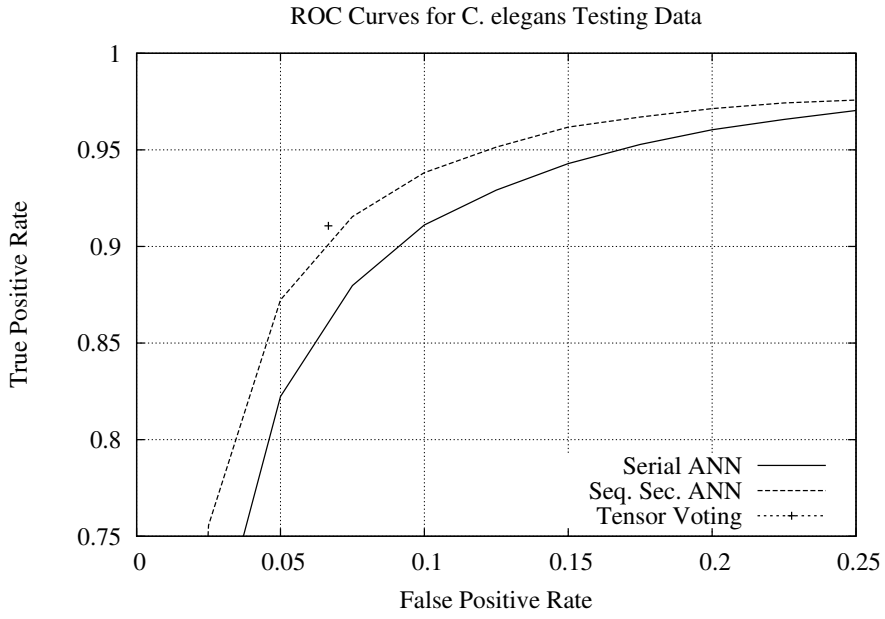


Figure 4.5. This ROC curve shows the improvement in the true positive and false negative rate with the use of the sequential section ANN (Seq. Sec. ANN) and the tensor voting, compared to using the serial ANN alone.

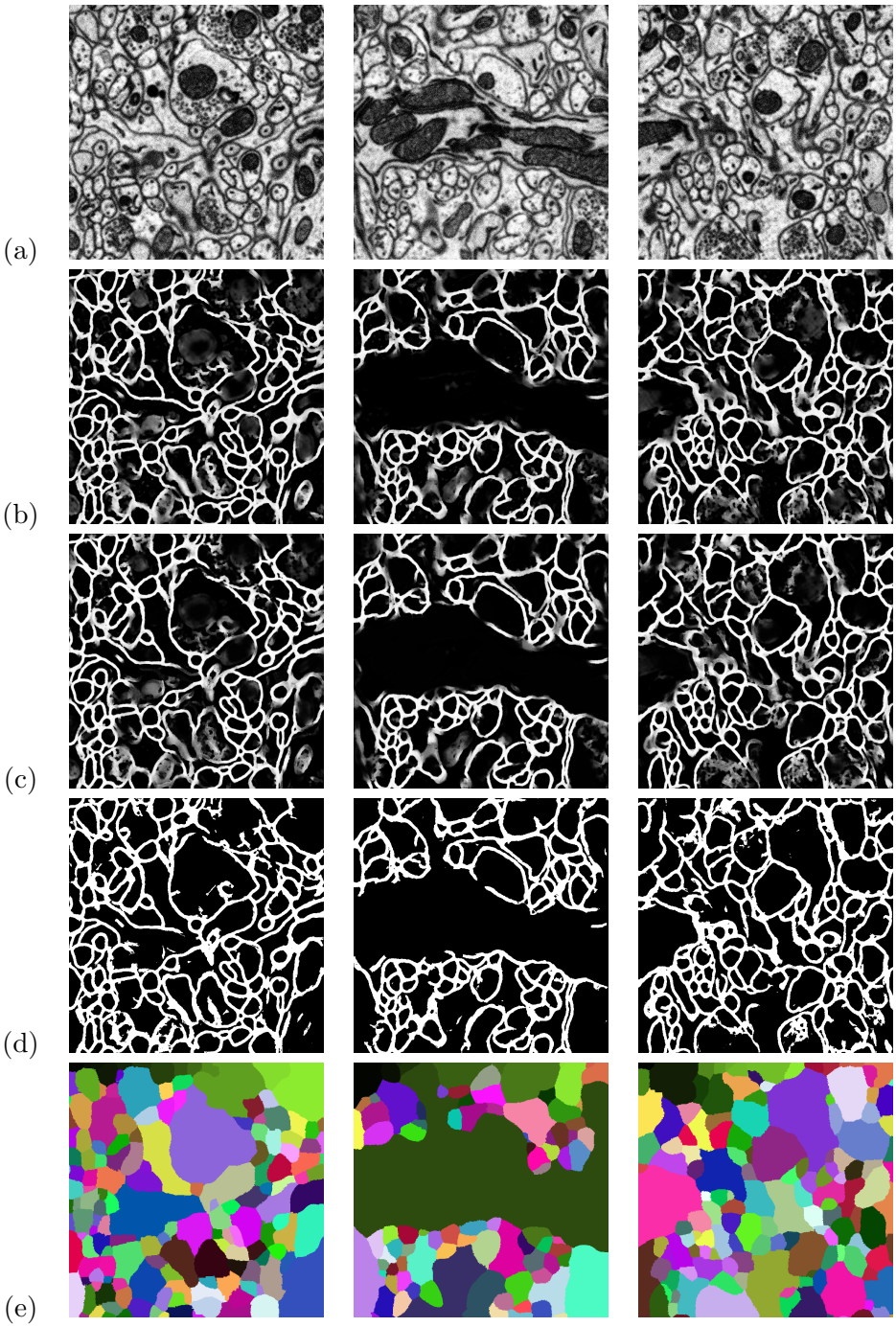


Figure 4.6. Output of the method on a three different test images from the mouse neuropil. (a) is the raw image, (b) is the output from the final stage of the series ANN (Section 3.3.3), (c) is the output from the sequential section ANN (Section 4.2), (d) is the output after tensor voting (Section 4.3) and (e) is the segmentations of the neuron regions from a watershed filter.

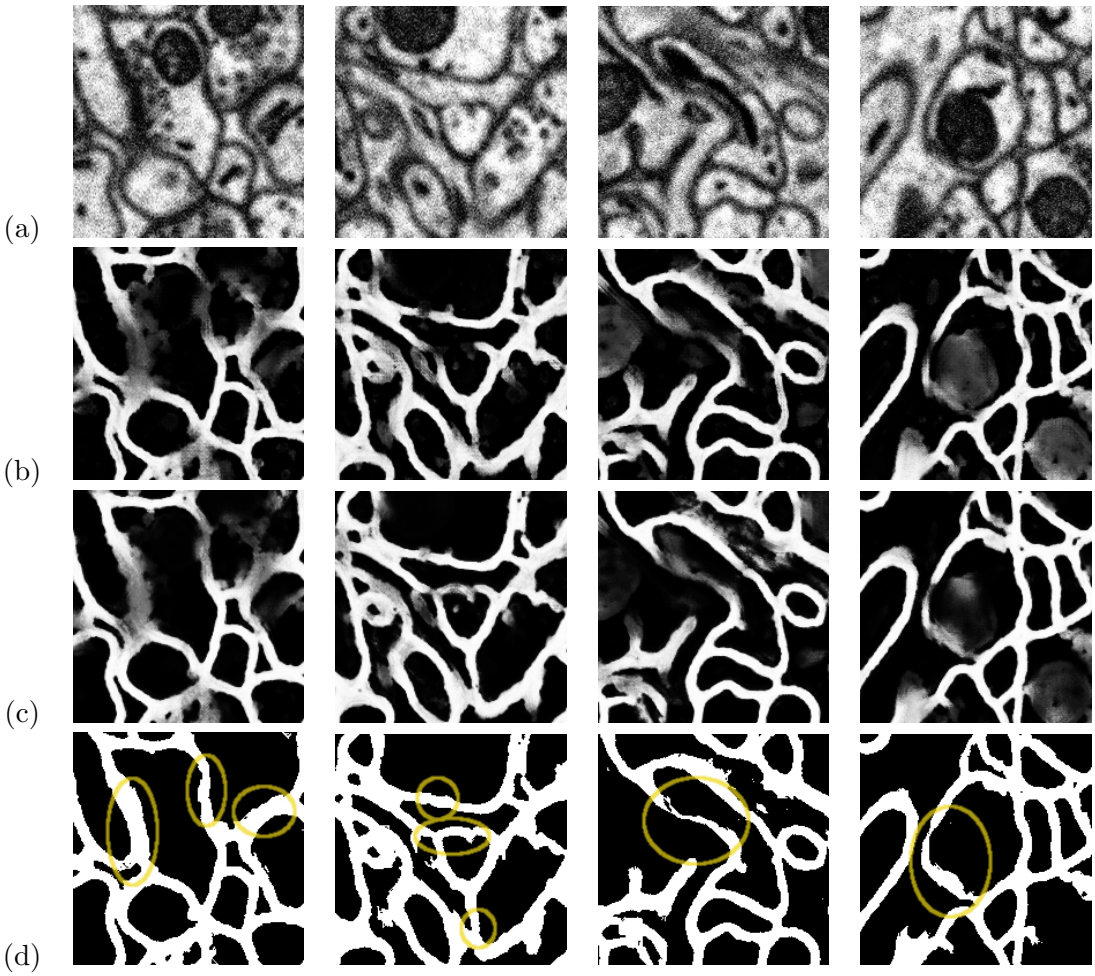


Figure 4.7. Example images demonstrating this method closing gaps on neuropil data. (a) is the raw image, (b) is the output from the final stage of the series ANN (Section 3.3.3), (c) is the output from the sequential section ANN (Section 4.2), and (d) is the final output after tensor voting (Section 4.3). Yellow circles highlight membranes that were enhanced and detected using these methods.

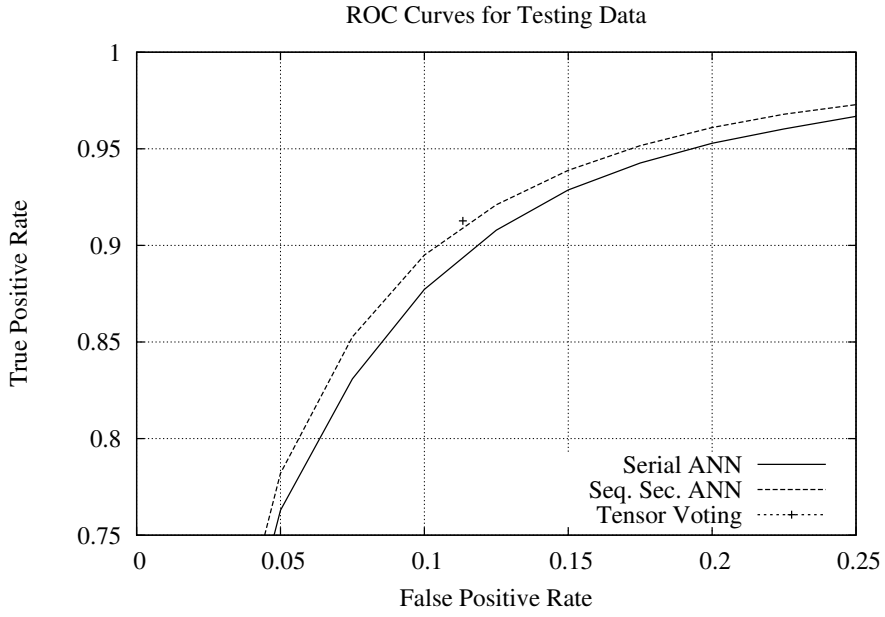


Figure 4.8. ROC curves showing the improvement of the two methods highlighted in this paper to improve the segmentation.

CHAPTER 5

3D NEURON RECONSTRUCTION

5.1 Introduction

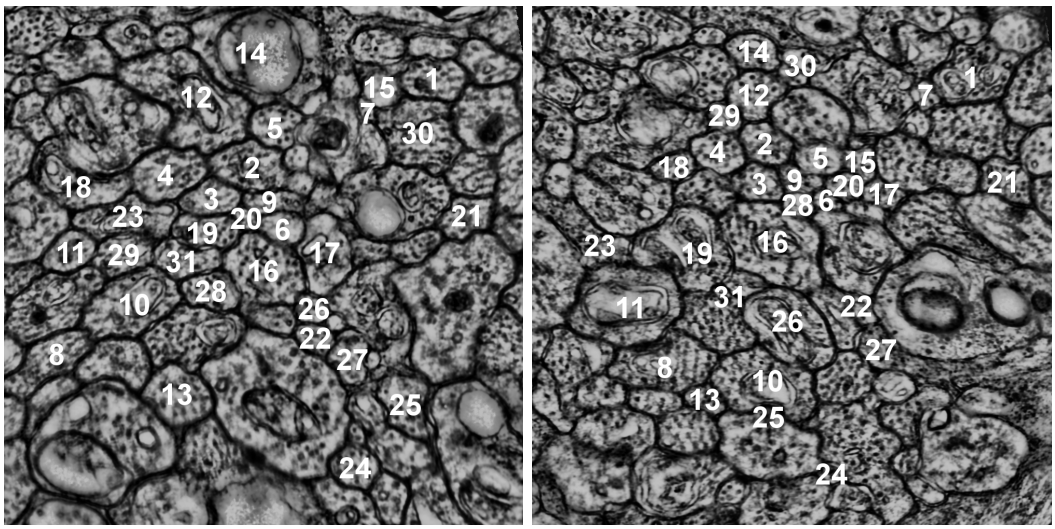
For this thesis, neuron identification across a stack of EM images is formulated as an optimal path problem with a graph data structure [55]. The vertices of the graph are defined as the regions obtained by 2D segmentation of the individual sections as described in Section 3.4. Edges in the graph represent possible linkages between regions in neighboring sections. Linking together the neuron regions in the graph is performed using Dijkstra's shortest path algorithm. The goal of this work is to map neuron regions in the first section to neurons in the last section, tracking processes that do not branch or terminate in between the first and last sections. This mapping looks similar to Figure 5.1. This is described in detail in Section 5.2. Once all of these paths through the volume are identified, the neurons can be rendered in 3D (see Section 5.5.2). This process is performed through a semi automatic reconstruction tool, called the Neuron Reconstruction Viewer (NeRV). Combining the automated tools with user interaction enables complete rendering of neurons in a 3D volume. This process is expanded on in Section 5.4.

5.2 Region Linking Method

Linking segmented regions across sections is formulated as a single source shortest path problem in a graph structure. Let $R_{s,i}$ be the i^{th} region from the 2D segmentation in section s. A directed graph containing a set of nodes that corresponds to the set of segmented regions in section s is constructed. The set of directed edges on the graph is between all nodes in adjacent sections. That is,

$$E = \left\{ \bigcup_{s,i,j=1}^{N,Q_s,Q_{s+1}} E_{s,i,j} \right\} \text{ where } E_{s,i,j} = [R_{s,i}, R_{s+1,j}], \tag{5.1}$$

N is the total number of sections, and Q_s denotes the number of segmented regions in section s.



(a) (b)

Figure 5.1. Example of images from a serial section TEM. (a) Labeled neurons in section 1 and (b) labeled neurons on section 28, identified through the optimal path finding algorithm.

A path through the graph is defined as a sequence of nodes connected by edges. Paths must span all sections $P = (R_{1,i_1}, R_{2,i_2}, \ldots, R_{N,i_N})$, and the cost of the path is defined as the sum of the costs of the edges

$$K(P) = \sum_{s=0}^{N-1} W(E_{s,i_s,i_{s+1}}), \tag{5.2}$$

where $i_1, \ldots i_N$ is the set of indices that the path follows on each section; because of the directed nature of the graph, paths cannot cross back to previous sections.

For biologists, the identification of neurons between sections relies on texture, shape, and proximity. These properties motivate the construction of the edge cost as the negative of the log product of the correlation between regions and a Gaussian penalty on in section displacement. That is:

$$W(E_{s,i,j}) = -\log\left[C(R_{s,i}, R_{s+1,j}) \exp\left(\frac{-D(R_{s,i}, R_{s+1,j})^2}{\sigma^2}\right)\right], \tag{5.3}$$

where $D(R_{s,i}, R_{s+1,j})$ is the Euclidean distance between region center of mass in the x, y coordinates of the section. σ is the maximum distance the neurons are expected to move between sections. C is the maximum value of the normalized cross correlation of the two segmented regions. Correlation is used most commonly in image processing and computer vision for locating or matching specific features across scenes. In this case, it is used to measure how well a region in section s matches with another region in section $s + 1$. The

two section images are multiplied with the characteristic function of the regions (0 outside, 1 inside) corresponding to $R_{s,i}$ and $R_{s+1,i}$ to obtain the masked images $I'_{s,i}$ and $I'_{s+1,j}$, respectively. Then, the normalized cross correlation between two vertices of the graph is computed as

$$C(R_{s,i}, R_{s+1,j}) = \frac{\max\limits_{t_x,t_y} \sum\limits_{x,y} I'_{s,i}(x - t_x, y - t_y) I'_{s+1,j}(x, y)}{\sqrt{\left(\sum\limits_{x,y} I'_{s,i}(x, y)^2\right)\left(\sum\limits_{x,y} I'_{s+1,j}(x, y)^2\right)}}. \tag{5.4}$$

For computational efficiency, the cross correlation is computed in the Fourier domain. The log is used so that the formulation is equivalent to a product through the sections, and the system avoids seeking out very good connections at the expense of very bad ones. Cell identity is lost if a connection between sections is not sufficiently strong. Finally, the log product, which can be seen as an edge connection weight, is negated to create a cost function.

5.3 Region Linking Extensions

The algorithm described in Section 5.2 is moderately effective, but fails in cases where the 2D segmentation fails or the quality of a section is particularly bad. The method can be made more robust with two additional features. The first is to account for over segmentation by inserting extra nodes in the graph that correspond to *merged* regions. Let

$$M_{s,i} = \{j : R_{s,j} \text{ is adjacent to } R_{s,i}\}, \tag{5.5}$$

represent the indices of all regions that are neighbors of $R_{s,i}$ (i.e. they contain adjacent pixels). Next, a subset of $M_{s,i}$ is defined corresponding to those neighbors of $R_{s,i}$ whose union with $R_{s,i}$ will be considered as additional nodes in the graph:

$$\tilde{M}_{s,i} = \{j \in M_{s.i} \text{ and } g(R_{s,i}, R_{s,j}) < T\}. \tag{5.6}$$

The function $g(R_{s,i}, R_{s,j})$ measures the boundary strength between any two adjacent regions in the same section. Because boundaries are dark, the negative of the maximum value of the intensities along the boundary that separates the two regions is used to evaluate boundary strength. If the edge strength is less than a threshold T, these regions become part of $\tilde{M}_{s,i}$ in (5.6). A new set of regions is defined in a section as

$$\{R_{s,i} \cup R_{s,j}\}_{j \in \tilde{M}_{s,i}}, \tag{5.7}$$

and augment the set of nodes, edges, and edge costs accordingly. The inclusion of these merged regions as well as their individual constituent regions provides this approach with

the flexibility to correct 2D over segmentation problems, but does not address 2D under segmentation problems.

Another important extension to this basic framework allows paths to skip sections, in order to avoid poor quality sections (which can happen regularly). To accomplish this, edges are added to the graph that allow connections up to M sections away:

$$E = \left\{ \bigcup_{k,s,i,j=1}^{M,N,Q_s,Q_{s+k}} E_{s,i,j,k} \right\} \text{ where } E_{s,i,j,k} = [R_{s,i}, R_{s+k,j}] \tag{5.8}$$

where k is the number of skipped sections. For the datasets presented in this dissertation, $M = 2$, thereby allowing connections between sections separated by at most a single intermediate section. This gives Dijkstra's algorithm a choice in calculating the best path in the case where an immediately adjacent section does not have the best match. This changes the construction of costs for these edges, because cost functions that favor skipping sections when there is sufficient data to support a path through a section need to be avoided. The function in Equation 5.3 is adjusted to penalize the correlation and distance terms for the skipped sections. Generally,

$$W(E_{s,i,j}) = -\log\left[\alpha^{k-1} C(R_{s,i}, R_{s+1,j}) \exp\left(\frac{-D(R_{s,i}.R_{s+1,j})^2}{k\sigma^2}\right)\right], \tag{5.9}$$

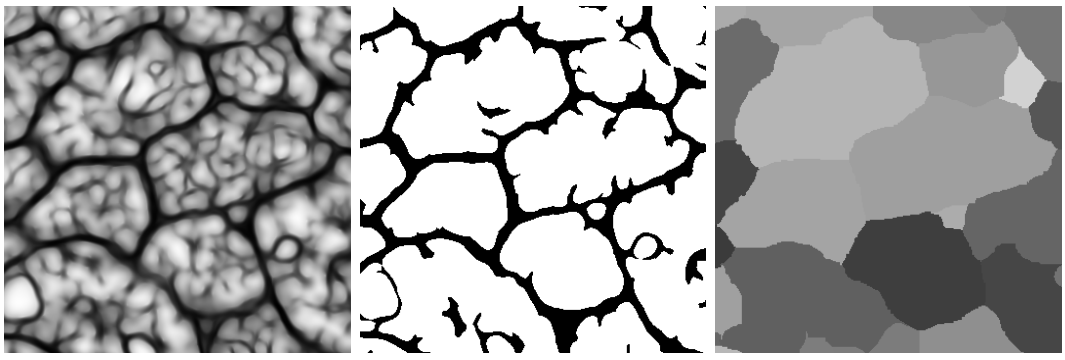
α is the typical normalized correlation penalty between a cell in two adjacent sections, which was found empirically to be about 0.6. The displacement Gaussian's variance is multiplied by k allowing more spatial movement when a section is skipped. The effect of these changes is to normalize the correlation, but allow for more displacement between the skipped regions. The displacement variance for the Gaussian is multiplied by k allowing for more spatial movement when a section is skipped. Overall, this increases the edge cost for $k > 1$.

Dijkstra's algorithm, which finds a minimum distance path in a directed graph is used to find the optimal connectivity for each neuron (region) in the first section. Dijkstra is run with a zero cost for all the regions in the first section. The region with the best cost is found on the last section, and tracing the solution backwards results in the optimal path (best neuron) for the whole data set. Of course in this solution, neurons can share paths, which is not desirable this particular application. To account for this, uniqueness is enfored iteratively, in a greedy optimization strategy. That is, to solve for the best path, remove those nodes from the graph, and repeat, producing a sequence of cells associated with a decreasing degree of evidence for connectivity.

5.3.1 Region Linking Example

The first step of the algorithm is to segment the individual neurons in the 2D images. As a proof of concept, a segmentation was constructed first with a contrast limited adaptive histogram equalization (CLAHE) applied to the raw data, improving the contrast of the neuron membranes and fixing the contrast variation. Furthermore, many of the cell membranes appear to contain gaps. To address this problem, a directional diffusion filter is applied [102] to enhance the neuron boundary. To isolate the membranes, the image is thresholded and a connected component filter is applied to remove components that are small in size and do not correspond to neuron membranes. This results in a binary image of the neuron membranes, which are then blurred with a Gaussian filter to obtain a fuzzy edge map. The final step applies a watershed segmentation using the implementation from ITK [16]. A perfect 2D segmentation is not possible because of the problems with grazed membranes, as described in Chapter 1, and the presence of intracellular structures. Figure 5.2 is an example of this process.

This method for tracking neurons through sections was applied to a mouse retinal dataset with a resolution of 2.5×2.5×90nm. As a preprocessing step, the dataset is registered and assembled using displacement histograms [100], which accurately aligns all the individual sections from the microscope and creates a volume. Each section is segmented in 2D, using the method described above, to create a series of regions through the volume. Figure 5.3 shows regions identified as the best path for 4 neurons across 28 sections using the proposed method. The initial segmentation is the top right image were performed by hand. The



(a) (b) (c)

Figure 5.2. Preliminary segmentation method applied to the image in Figure 5.1.(a) output from the diffusion filter, (b) thresholding and connected component removal, and (c) the watershed segmentation.

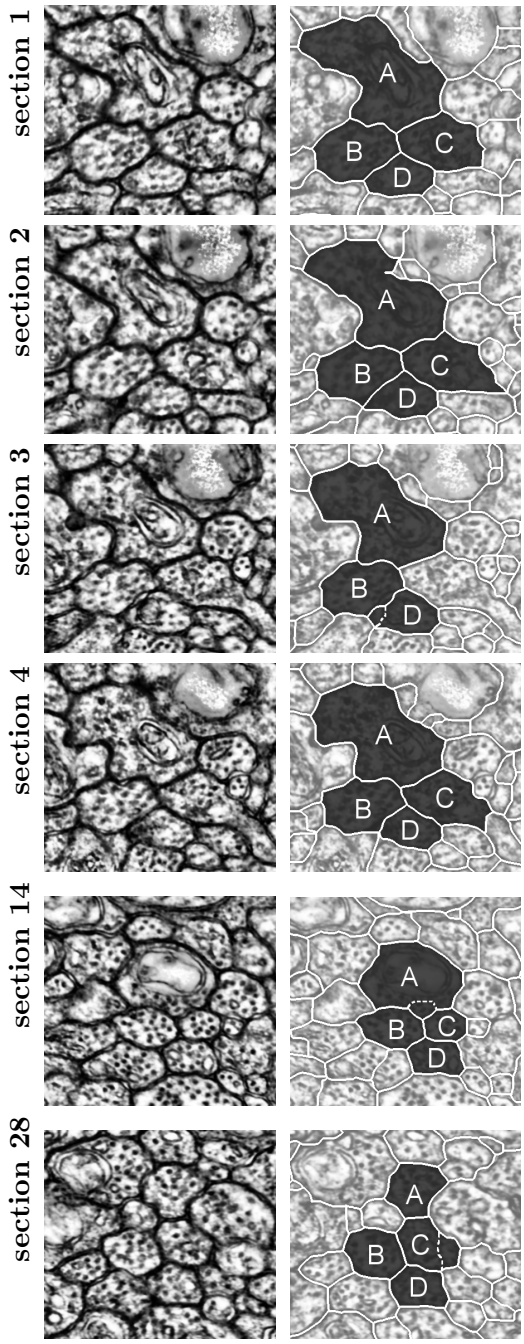


Figure 5.3. Sequence of images showing the result of tracking 4 neurons over 28 sections. The left column is the CLAHE enhanced raw data and the right column is the corresponding segmentation for the tracked neurons. Sections 1, 2, 3, 4, 14, and 28 are shown from top to bottom. Letters correspond to matching regions between sections.

following sections are segmented automatically and identified using the optimal path finding algorithm. The labels for these regions are found to be correct when compared to data labeled by an expert after the system parameters were set. The quality and the consistency of the neuron boundaries can change dramatically between sections. For example, in Figure 5.3, Section 3, the lower right portion of the boundary for region C is not captured and therefore under segmented. As a result, the edge cost from region C in section 2 to region C in section 4 is much smaller when compared to the edge costs from region C in section 2 to any regions in section 3. This causes section 3 to be skipped for region C. It is possible this neuron is splitting, as indicated by the abrupt change in region shape. This is an important area of future research, since splitting indicates the beginning of other neurons. Also in Section 3, region D is over segmented. As a consequence, the algorithm prefers a merged alternative region over all of the single regions directly obtained from the 2D segmentation. The boundaries between the two regions that were merged are shown as dashed lines in Figure 5.3. Region A in section 14 and region C in section 28 are also merged in Figure 5.3. Over the 28 sections, regions A, B, C and D skipped a total of 9 sections and performed 6 merges.

This algorithm demonstrates an effective method for tracking neurons through serial section TEM images. This is a significant step in bridging the gap between image acquisition and the reconstruction of neural circuitry. Individual sections are segmented into sets of regions, a connected graph is built in 3D, and a version of Dijkstra's algorithm to used to calculate a unique path through the graph. To account for over and under segmented regions, this method can merge regions within a section or skip regions between sections. Failure to correctly identify a sequence of neurons can still occur when an accurate segmentation of a neuron is missing for more than one section or more than two regions need to be merged for the best possible correlation. Both of these cases will most likely result in diverting the optimal path to a different path, with a higher correlation.

5.4 Neuron Reconstruction Viewer

The automatic methods described up until this point all work fairly well on their own, but in the end, require the ability for viewing and editing of the segmentation results. The Neuron Reconstruction Viewer (NeRV) (shown in Figure 5.4) attempts to bridge these two requirements by providing an interface to large volumes of EM images and viewing of neuron segmentations, with the option to make corrections, which will, in the long term improve

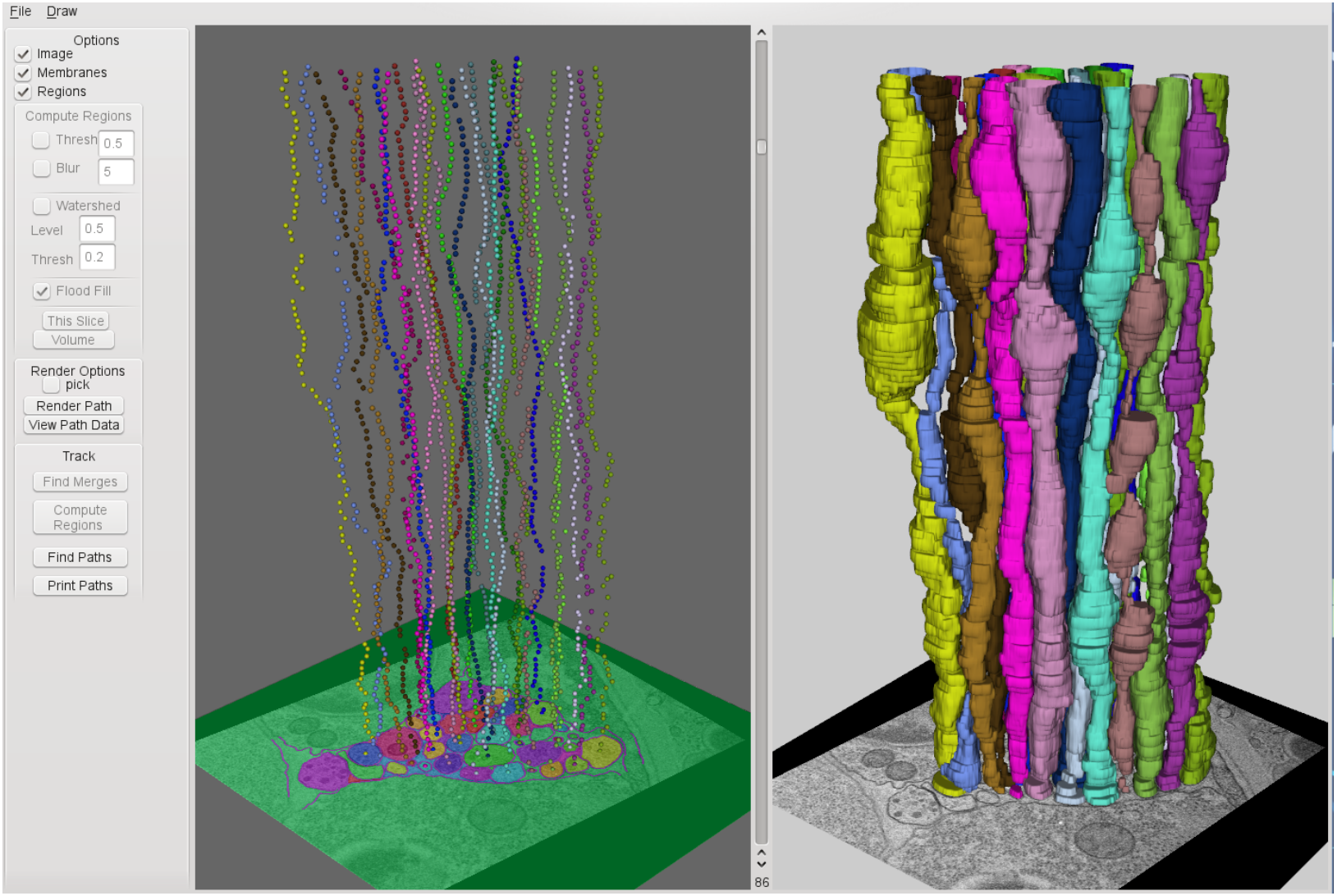


Figure 5.4. Screen capture of NeRV displaying the automatic segmentation results on the *C. elegans* ventral nerve cord for a portion of the data.

the segmentation.

Primarily, NeRV is an interface for the user to view the raw image data and the 3D reconstruction. Interacting with the image data and the rendered neuron provides insight for the scientist on the arrangement of the neurons within the data. The pane on the left, in Figure 5.4, is mainly a slice viewer. The user can view the membrane detection, the region segmentation, and the raw data all in one viewer. Spheres highlight the paths neurons take through the volume. The keyboard arrow keys or the slider in the middle lets the user scroll through the sections. The pane on the right, is a 3D viewer of the reconstructed neuron. Raw image data can be turned off and on in this view, and users can select other sections simply by clicking on the area of the neuron. Figure 5.5 gives an overview of how NeRV and all the command line tools written to process the EM image data interact.

Command line tools: There is a set of command line tools that generate all the results and data described in this thesis. These tools are required for NeRV to have access to all the processed data.

1. *Membrane detection:* Once the series ANN and sequential section ANN membranes are learned, all the weights from this step are used to compute final membrane

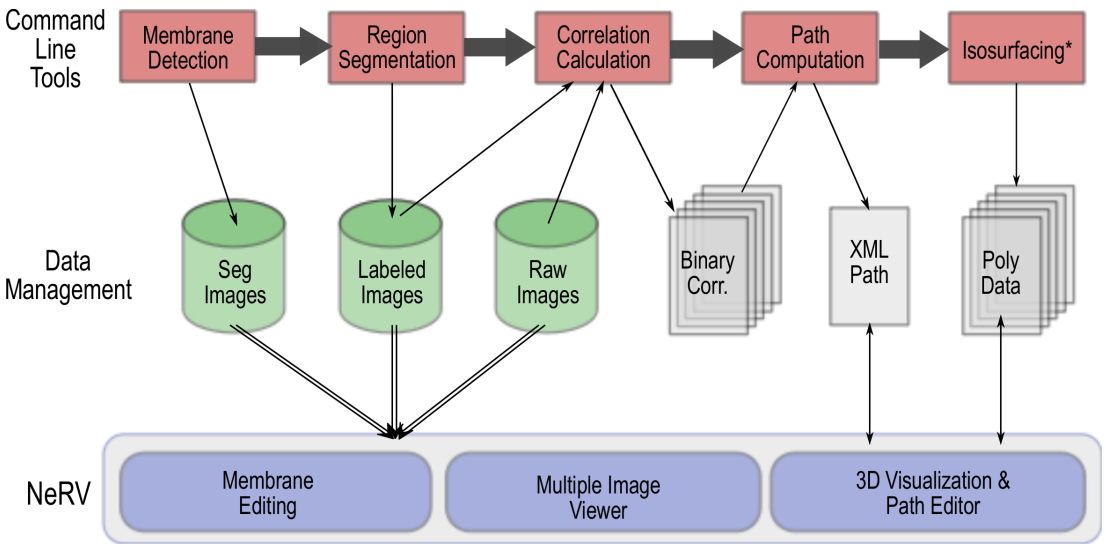


Figure 5.5. Overview of NeRV and the command line tools architecture. The top set of boxes describe the data flow through all the command line tools. Arrows indicate read/input and write/output operations. Arrows with double lines indicate data that is being streamed from disc to the application. * The isosurfacing command line tool also takes as input the labeled images.

segmentation on the whole data set, not just the test and training images. The tensor voting is also included in this step.

2. *Region segmentation:* Given a set of membranes, computing a flood fill or watershed segmentation, as discussed in Section 3.4, is computed.
3. *Correlation calculation:* This tool takes two main input, the labeled images from the previous step and the raw image files. The labels are used to isolate neurons and a normalized correlation is computed, along with a distance between centers of regions. This information is put into a correlation file containing all the related correlation information between every two sections. Given large data files, this can be a very time consuming step. As a result, all the data is stored in a binary file, and only correlations between a distance are computed.
4. *Path computation:* Reading in the correlations, a graph is quickly built only for regions within a σ distance, as discussed in Section 5.2. This greatly reduces the size of the graph.
5. *Isosurface renderings:* The last step takes, as input, the paths, stored as an XML data file, and the labeled images and computes an isosurface representing the neuron segmentation.

Data management: Next, we outline the image data and associated files required for these tools to be processed.

1. *Raw images:* This is the image data from the microscope, assembled and aligned.
2. *Membrane detection images:* Binary images containing the detected membranes in each section.
3. *Labeled images:* Labels for each neuron are represented as unsigned int images. Each label is unique, increasing in order from largest region to smallest.
4. *Correlation files:* Binary matrix of correlations between all regions between two sections.
5. *XML path file:* All the paths are stored in one XML file. Each path is identified by its section number and x, y center.
6. *VTK poly data:* VTK file for storing a polygonal mesh of neurons.

NeRV user capabilities Users can interact with this using the graphical interface on the far left. First, users can correct segmentations to close gaps with a simple drawing tool. Then recompute the regions and correlations to improve the optimal path calculation (as discussed in Section 5.2). Users can manually select regions in slices and create their

own 3D renderings with the automatic path calculation. For precomputed and segmented neurons, a separate window allows users to select different neurons for viewing, deleting or joining.

NeRV is built primarily using VTK [88] and Qt [74]. To handle large datasets, the VTK image data streamer is used to load only the images required for viewing and requested by the user. Since slices are loaded as needed, the memory of this system is limited only by the size of a single section. Other optimizations, such as down sampling and efficient memory deletion, enable building the isosurfaces for the 3D reconstruction in the right pane. Specifically, NeRV's capabilities include the following:

1. *Membrane editing:* Users can edit membranes with a simple paint-like interface.
2. *Image viewing:* NeRV acts as a simple 2D image viewer, displaying and loading into memory only the sections the user requests. All three image data types (raw, membranes, labels) can be optionally shown overlaid with each other.
3. *Manual path selection:* A window can be requested that shows a list of all the possible rendered neurons. Selecting neurons from this list turns them on and off in the viewer. Additional tools exist to manually select a new path, join already computed paths, and deleting paths.
4. *3D viewer:* Along with the image data, isosurfaces representing the neurons are shown in one view. Because of the large set of sections, users can click anywhere along the neuron to view raw data at that section.
5. *Isosurface renderings:* As users manual select paths, renderings of neurons can be computed in this application, not just in a command line tool.

5.5 Results

Two EM datasets, presented in Section 4.4, are reconstructed into 3D visualizations using the proposed methods. The first dataset is a stack of 300 sections from the ventral nerve cord of the *C. elegans* and 400 sections of the mouse neuropil.

5.5.1 The *C. elegans* Ventral Nerve Cord

Figure 5.6 shows the 3D reconstruction of 10 neurons through the first 300 sections of the *C. elegans* ventral nerve cord. Each image was edited by a user to close remaining gaps and a flood fill was performed to segment each 2D region. Building the volume was a two part processes. First, we identified 6 significant breaks in the image volume where there was missing data, due to lost or badly imaged sections. These were places where the data

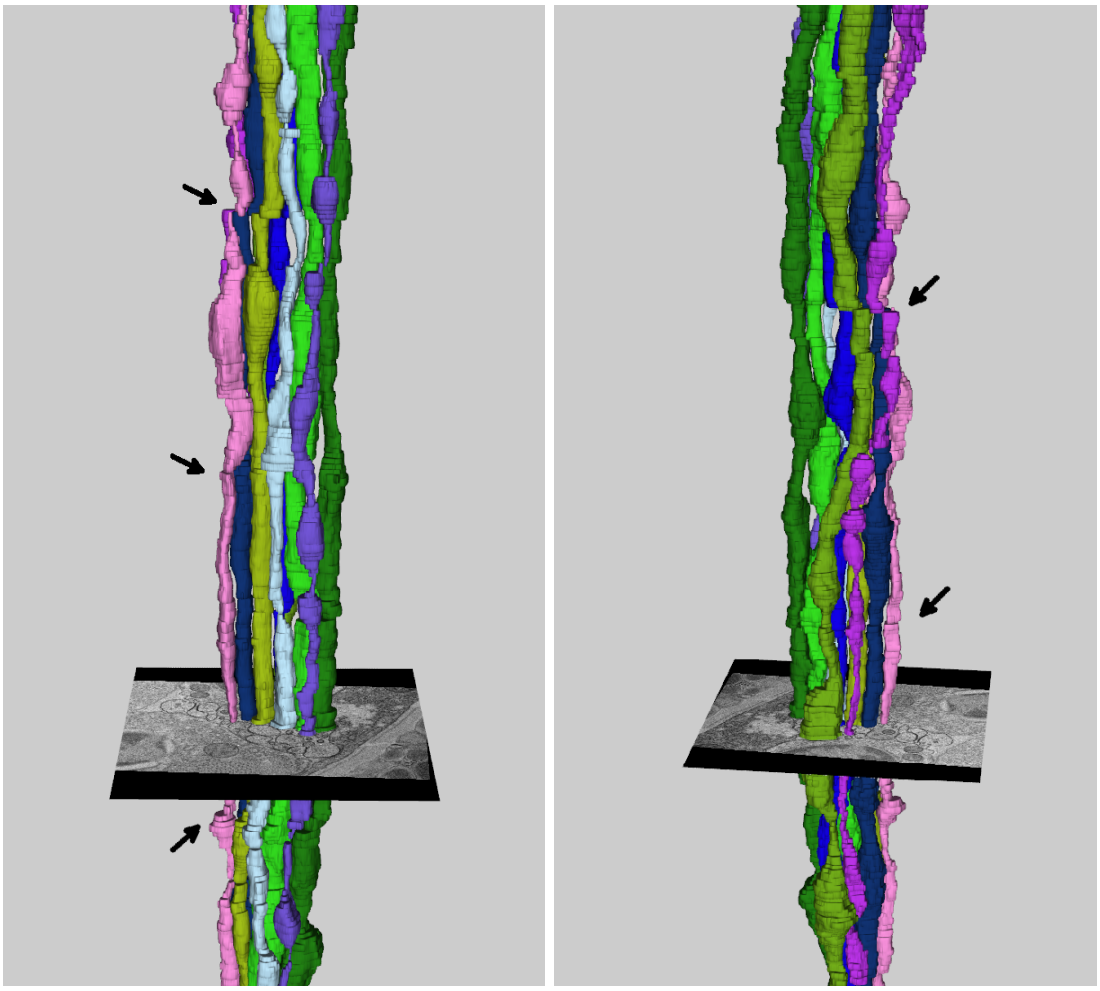


Figure 5.6. Two views of 10 neurons spanning 300 sections of the ventral nerve cord of the *C. elegans*. Neuron paths were generated automatically between six pairs of sections where known breaks in the image data existed. NeRV was used to connect paths between the breaks. Arrows (above) identify discontinuities in neurons where some of these breaks occurred.

had significant changes and would cause the region linking algorithm to fail. As a result, neuron regions were linked only between the sections without breaks, producing six sets of paths that spanned the whole volume. To completely reconstruct these paths through the whole 300 sections, NeRV was used to manually merge neurons in sequential sections, forming complete reconstructions through the whole volume. Figure 5.6 is the final output from this process.

5.5.2 The Mouse Neuropil

Final reconstruction of the volume on the entire dataset turned this task into a large data challenge, since the actual size of the full volume is much larger than the training data. First, neuron membranes needed to be detected in each 4Kx4K section, which took about 40 minutes for each section, including applying all the weights in series from the ANNs and using tensor voting. The watershed segmentation algorithm worked well on this dataset because of its large size, making hand editing too time consuming, and the small gaps in remaining membranes. This was applied to the blurred output from the tensor voting. This was much faster, taking only about 1 minute per section. Calculating the correlations between every two sections took between 40 minutes and 1.5 hours, depending on the number of regions in each section. To reduce computational time and the required memory, correlations were only calculated for regions within β distance away. For this dataset $\beta = 150$. Once all the correlations are calculated, a graph can be constructed and the Dijkstra algorithm can be used to find paths through the volume. The generic implementation of Dijkstra's algorithm has a complexity of $O(r^2)$, where r is the number of nodes in the graph. However, for this case, since the edges in the graph are limited to connections between sections, the complexity is $O(\frac{r^2}{N})$. The graph can be made even more sparse by limiting the number of edges to regions by $D(R_{s,i}, R_{s+1,j}) < d$, where d is the maximum distance a region is allowed to connect between region centers. For this dataset $d = 100$. This means the algorithm can scale more easily to larger graphs. To scale the path calculation even further, the volume was divided into smaller slabs and paths were found through every 25 sections. Each path then took approximately 4 minutes to compute and paths were easily joined by matching paths with overlapping sections. To view in 3D, each path that spanned more than 300 sections was rendered in 45 minutes. Multiple processor machines were used to compute these results as efficiently as possible, in parallel. NeRV easily handles the size of this data because it only loaded into memory what was requested

by the user. Finally, users can easily select the neurons they want to view in the volume. The final 3D visualization of this dataset can be seen in Figure 5.7.

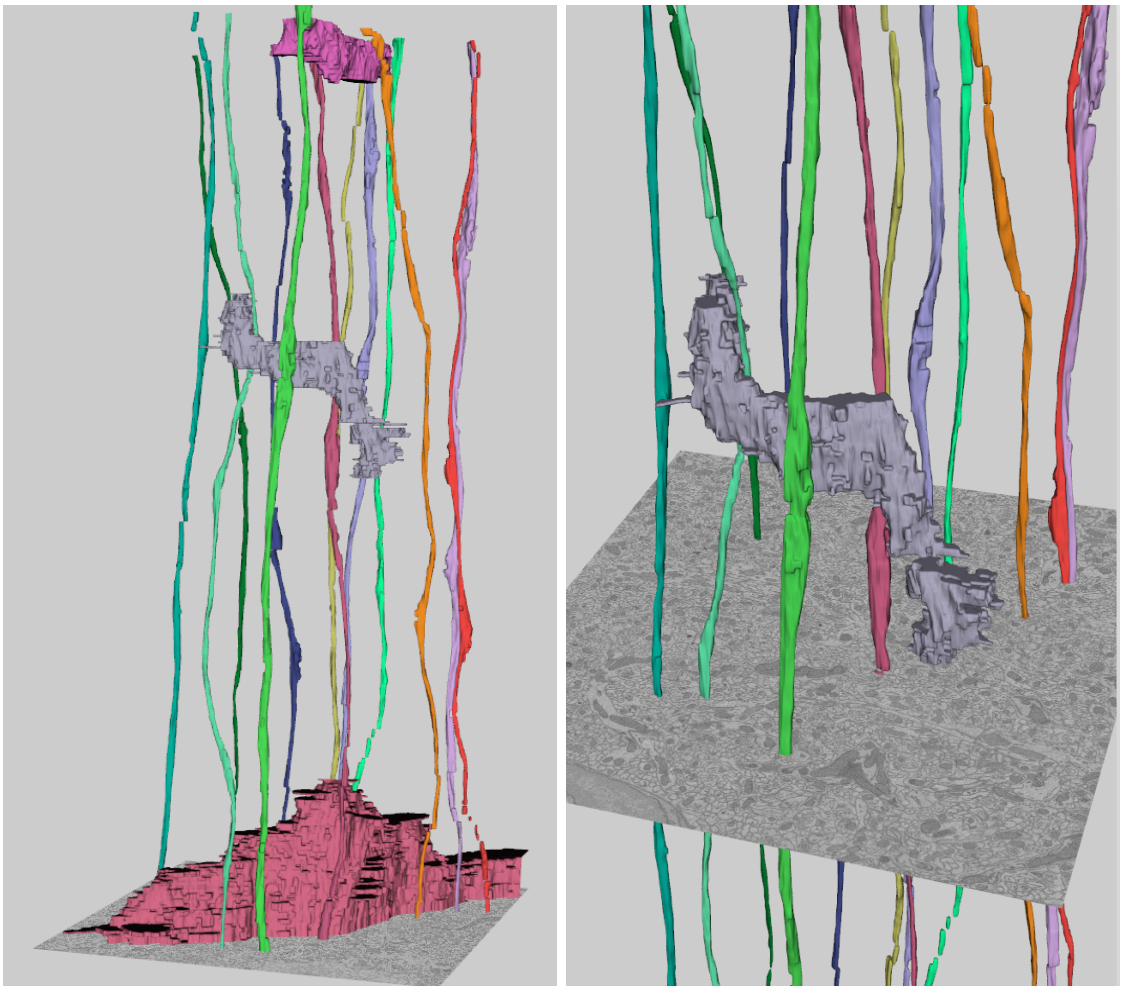


Figure 5.7. Two views of 15 fully automatically segmented parallel fibers spanning 400 sections of the mouse neuropil. The larger 3 structures were segmented manually using the NeRV interface. Discontinuities in the neuron renderings indicate sections the automatic algorithm skipped because of changing neuron regions.

CHAPTER 6

CONCLUSION

Neural circuit reconstruction is an important method for studying neural circuit connectivity and its behavioral implications. The differences between neuronal classes, patterns, and connections are central to the study of the nervous system and critical for scientific discoveries in understanding how species learn and how scientists can fight diseases. Electron microscopy is a useful modality for scientists attempting to map the anatomy of individual neurons and their connectivity because it has a resolution that is high enough to identify features, such as synaptic contacts and gap junctions. These features are important indicators for types of neuron topology and connectivity, and therefore are required for neural circuit reconstruction. Towards this goal, neurobiologists are acquiring large electron microscopy datasets. However, the sheer volume of these datasets renders manual analysis infeasible. Hence, automated image analysis methods are required for reconstructing neuronal processes from these very large image collections.

Automated image processing techniques are challenging due to the diverse nature of neuronal tissue in EM data. Cell membrane structures change depending on the source of the tissue. The presence of intracellular structures and varying nature of cell membranes, due to the cutting plane, also make image segmentation a difficult problem. An algorithm that works well on one dataset may fail on a different dataset. Another challenge is the anisotropic nature of the data making direct 3D approaches impractical.

Addressing these challenges, this dissertation presented a method for the segmentation and visualization of neurons from EM images. To be able to detect membranes in different types of EM images, membranes in each section are segmented using a series of ANNs that uses only image intensities as input, making this method flexible for different data sets. Also the series of ANNs learns context associated with membrane structures, removing intracellular objects and closing gaps that result from low contrast membranes. Making use of overlapping information in sequential sections, the learned membranes from this algorithm are registered with neighboring learned membranes and used in a final ANN to

improve the classification. Examining the detected membranes in sequential sections, above and below the current section, also helps the classifier learn to detect membranes that are grazed or complicated by internal cellular structures. In the last step, tensor voting closes remaining gaps in the image. Incorporating all these tools together, a software application, NeRV, provides an interface for users to correct 2D segmentations and manually assign neuron paths through sections. Users can also view automatically generated paths and join sections. This aids the user in visualizing the reconstruction data. These algorithms and tools provide a process for segmenting neurons at all levels, from the raw data to the 3D visualization.

6.1 Future Work

There are two main areas for improvement in the method proposed in this dissertation. First, the path computation and linking of neuron regions needs to handle neurons that branch and terminate. A more flexible algorithm, such as bipartite graph matching, should be able to better handle these special cases. The second approach that can be improved is the 2D region segmentation, currently performed using a flood fill or watershed segmentation. The flood fill requires the user to edit membrane segmentations and the watershed tends towards over segmented regions. A more advantageous technique may include incorporating the membrane confidences into a graph cut or level set energy term that would produce more accurate 2D segmentations.

The machine learning framework can be extended in a several ways. First, it would be relatively simple to train the series of ANNs to detect synapses and vesicles, which are necessary for complete neuron connectivity. In addition, incorporation of a multiscale context sampling method to train the series of ANNs would improve the initial segmentation. To make this method more available to multiple image types, we envision implementing a database of pretrained classifiers that will segment neuron membranes for multiple image types. Users can then select the most appropriate classifier to segment their images, without having to wait for annotation and training to take place.

NeRV should be extended with more user interfaces for editing neuron segmentations, such as making corrections and handling branching. Current interfaces to correct 2D segmentation are limited to simple drawing techniques. NeRV would be more powerful if it implemented live wire for aiding in membrane segmentation corrections. One of the bottlenecks associate with NeRV is reading in all the relevant data, such as the correlation

files and the polygonal data. This would be more efficient if stored in a database for fast access when requested by the user. NeRV should also expose the confidence associated with correlations so that bad paths can be quickly identified and corrected by the user.

We plan on extending this method on the full *C. elegans* ventral nerve cord dataset described in this dissertation. This would reveal in 3D the physical layout of neurons and can then easily compared to ground truth data from White *et al.* Further work to segment the muscles that surround the nerve cord would provide insight into communication and wiring in the *C. elegans*. Likewise, a more thorough analysis of the 3D reconstructions in the mouse neuropil needs to be completed so we can develop a better understanding of the types of connections present in this dataset.

REFERENCES

[1] Allen, B. A., and Levinthal, C. Cartos ii semi-automated nerve tracing: Three-dimensional reconstruction from serial section micrographs. *Computerized Medical Imaging and Graphics 14*, 5 (1990), 319 – 329.

[2] Anderson, J., Jones, B., Yang, J.-H., Shaw, M., Watt, C., Koshevoy, P., Spaltenstein, J., Jurrus, E., U.V., K., Whitaker, R., Mastronarde, D., Tasdizen, T., and Marc, R. A computational framework for ultrastructural mapping of neural circuitry. *PLoS Biology 7*, 3 (2009), e74.

[3] Anderson, J. R., Jones, B. W., Watt, C. B., Shaw, M. V., Yang, J. H., Demill, D., Lauritzen, J. S., Lin, Y., Rapp, K. D., Mastronarde, D., Koshevoy, P., Grimm, B., Tasdizen, T., Whitaker, R., and Marc, R. E. Exploring the retinal connectome. *Mol. Vis. 17* (2011), 355–379.

[4] Andres, B., Köthe, U., Helmstaedter, M., Denk, W., and Hamprecht, F. A. Segmentation of SBFSEM volume data of neural tissue by hierarchical classification. In *Pattern Recognition* (2008), G. Rigoll, Ed., vol. 5096 of *LNCS*, Springer, pp. 142–152.

[5] Arbelaez, P., Maire, M., Fowlkes, C., and Malik, J. From contours to regions: An empirical evaluation. *IEEE Conference on Computer Vision and Pattern Recognition* (to appear 2009).

[6] Awate, S. P., Tasdizen, T., and Whitaker, R. T. Unsupervised Texture Segmentation with Nonparametric Neighborhood Statistics. In *Proceedings of the European Conference on Computer Vision* (2006), pp. 494–507.

[7] Awate, S. P., and Whitaker, R. T. Unsupervised, information-theoretic, adaptive image filtering for image restoration. *IEEE Trans. on Pattern Analysis and Machine Intelligence 28*, 3 (2006), 364–376.

[8] Berger, T., Ahuja, A., Courellis, S., Deadwyler, S., Erinjippurath, G., Gerhardt, G., Gholmieh, G., Granacki, J., Hampson, R., Hsaio, M. C., Lacoss, J., Marmarelis, V., Nasiatka, P., Srinivasan, V., Song, D., Tanguay, A., and Wills, J. Restoring lost cognitive function. *Engineering in Medicine and Biology Magazine, IEEE 24*, 5 (Sept.-Oct. 2005), 30–44.

[9] Bertalmío, M., Sapiro, G., and Randall, G. Morphing active contours. *IEEE Trans. Pattern Anal. Mach. Intell. 22* (July 2000), 733–737.

[10] Betzig, E., Patterson, G., Sougrat, R., Lindwasser, O., Olenych, S., Bonifacino, J., Davidson, M., Lippincott-Schwartz, J., and Hess, H. Imaging intracellular fluorescent proteins at nanometer resolution. *Science 313*, 5793 (2006 Sep 15), 1642–5.

[11] Borenstein, E., Sharon, E., and Ullman, S. Combining top-down and bottom-up segmentation. In *CVPRW '04: Proceedings of the 2004 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'04) Volume 4* (Washington, DC, USA, 2004), IEEE Computer Society, p. 46.

[12] Braitenberg, V., and Schuz, A. *Cortex: Statistics and Geometry of Neuronal Connectivity*. Springer, Berlin, 1998.

[13] Briggman, K. L., and Denk, W. Towards neural circuit reconstruction with volume electron microscopy techniques. *Current Opinion in Neurobiology 16*, 5 (October 2006), 562–570.

[14] Briggman, K. L., and Denk, W. Towards neural circuit reconstruction with volume electron microscopy techniques. *Current Opinion in Neurobiology 16*, 5 (October 2006), 562–570.

[15] Buades, A., Coll, B., and Morel, J.-M. A non-local algorithm for image denoising. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition* (2005), pp. 60–65.

[16] Cates, J., Whitaker, R., and Jones, G. Case study: an evaluation of user-assisted hierarchical watershed segmentation. *Medical Image Analysis 9* (2005), 566–578.

[17] Chen, X., Winters, C. A., and Reese, T. S. Life inside a thin section: Tomography. *The Journal of Neuroscience 28*, 38 (2008), 9321–9327.

[18] Chklovskii, D. B., Vitaladevuni, S., and Scheffer, L. K. Semi-automated reconstruction of neural circuits using electron microscopy. *Current Opinion in Neurobiology 20*, 5 (2010), 667 – 675.

[19] Cottrell, G. Extracting features from faces using compression networks: face, identity, emotion and gender recognition using holons. Morgan Kaufmann, pp. 328–337.

[20] Cybenko, G. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems 2*, 4 (Dec. 1989), 303–314.

[21] Davalos, D., Lee, J. K., Smith, W. B., Brinkman, B., Ellisman, M. H., Zheng, B., and Akassoglou, K. Stable in vivo imaging of densely populated glia, axons and blood vessels in the mouse spinal cord using two-photon microscopy. *Journal of Neuroscience Methods 169*, 1 (2008), 1 – 7.

[22] DeBello, W. M., Feldman, D. E., and Knudsen, E. I. Adaptive Axonal Remodeling in the Midbrain Auditory Space Map. *J. Neurosci. 21*, 9 (2001), 3161–3174.

[23] Deerinck, T. J., Bushong, E. A., Thor, A., and Ellisman, M. H. Ncmir methods for 3d em: A new protocol for preparation of biological specimens for serial block face scanning electron microscopy. In *Microscopy and Microanalysis Meeting* (Aug 2010).

[24] Denk, W., and Horstmann, H. Serial block-face scanning electron microscopy to reconstruct three-dimensional tissue nanostructure. *PLoS Biol 2*, 11 (2004).

[25] Denk, W., Strickler, J. H., and Webb, W. W. Two-photon laser scanning microscopy. *Science 248* (1990), 73–76.

[26] Dollar, P., Tu, Z., and Belongie, S. Supervised learning of edges and object boundaries. In *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on* (Los Alamitos, CA, USA, 2006), vol. 2, IEEE Computer Society, pp. 1964–1971.

[27] Egner, A., and Hell, S. W. Fluorescence microscopy with super-resolved optical sections. *Trends Cell Biol 15*, 4 (April 2005), 207–215.

[28] Fiala, J. C., , and Harris, K. M. Computer-based alignment and reconstruction of serial sections. *Microscopy and Analysis 87* (2002), 5–8.

[29] Fiala, J. C., and Harris, K. M. Extending unbiased stereology of brain ultrastructure to three-dimensional volumes. *J Am Med Inform Assoc. 8*, 1 (2001), 1–16.

[30] Fiala, J. C., and Harris, K. M. Synapseweb. http://synapses.clm.utexas.edu/tools/reconstruct/reconstruct.stm, jun 2010.

[31] Fowlkes, C., Belongie, S., Chung, F., and Malik, J. Spectral grouping using the nyström method. *IEEE Transactions on Pattern Analysis and Machine Intelligence 26* (2004), 214–225.

[32] Franken, E., van Almsick, M., Rongen, P., Florack, L., and ter Haar Romeny, B. An efficient method for tensor voting using steerable filters. In *ECCV06* (2006), pp. IV: 228–240.

[33] Freund, Y., and Schapire, R. E. A decision-theoretic generalization of on-line learning and an application to boosting. In *EuroCOLT '95: Proceedings of the Second European Conference on Computational Learning Theory* (London, UK, 1995), Springer-Verlag, pp. 23–37.

[34] G., R. C., and Woods, R. E. *Digital Image Processing*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1992.

[35] Griffith, A. Chipping in. *Scientific American* (Feb 2007).

[36] Hama, K., Arii, T., Katayama, E., Marton, M., and Ellisman, M. H. Tridimensional morphometric analysis of astrocytic processes with high voltage electron microscopy of thick golgi preparations. *Journal of Neurocytology 33* (2004), 277–285. 10.1023/B:NEUR.0000044189.08240.a2.

[37] Haykin, S. *Neural networks - A comprehensive foundation*, 2nd ed. Prentice-Hall, 1999.

[38] Hayworth, K., Kasthuri, N., Schalek, R., and Lichtman, J. Automating the collection of ultrathin serial sections for large volume tem reconstructions. *Microscopy and Microanalysis 12*, Supp 2 (2006), 86–87.

[39] Hornik, K. Approximation capabilities of multilayer feedforward networks. *Neural Netw. 4*, 2 (1991), 251–257.

[40] Ibanez, L., Schroeder, W., Ng, L., and Cates, J. *The ITK Software Guide*, second ed. Kitware, Inc. ISBN 1-930934-15-7, http://www.itk.org/ItkSoftwareGuide.pdf, 2005.

[41] Imaging, V. Visage imaging, amira. http://www.amira.com.

[42] Jain, V., Bollmann, B., Richardson, M., Berger, D., Helmstaedter, M., Briggman, K., Denk, W., Bowden, J., Mendenhall, J., Abraham, W., Harris, K., Kasthuri, N., Hayworth, K., Schalek, R., Tapia, J., Lichtman, J., and Seung, H. Boundary learning by optimization with topological constraints. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on* (june 2010), pp. 2488 –2495.

[43] Jain, V., Murray, J., Roth, F., Turaga, S., Zhigulin, V., Briggman, K., Helmstaedter, M., Denk, W., and Seung, H. Supervised learning of image restoration with convolutional networks. *IEEE 11th International Conference on Computer Vision* (Oct. 2007), 1–8.

[44] Jain, V., Seung, H. S., and Turaga, S. C. Machines that learn to segment images: a crucial technology for connectomics. *Current Opinion in Neurobiology 20*, 5 (2010), 653 – 666.

[45] Jeong, W.-K., Beyer, J., Hadwiger, M., Blue, R., Law, C., Vazquez-Reina, A., Reid, R. C., Lichtman, J., and Pfister, H. Ssecrett and neurotrace: Interactive visualization and analysis tools for large-scale neuroscience data sets. *IEEE Computer Graphics and Applications 30* (2010), 58–70.

[46] Jeong, W.-K., Beyer, J., Hadwiger, M., Vazquez, A., Pfister, H., and Whitaker, R. T. Scalable and interactive segmentation and visualization of neural processes in em datasets. *IEEE Transactions on Visualization and Computer Graphics 15* (2009), 1505–1514.

[47] Jin, Y., Hoskins, R., and Horvitz, H. R. Control of type-D GABAergic neuron differentiation by C. elegans UNC-30 homeodomain protein. *Nature 372*, 6508 (Dec 1994), 780–3.

[48] Jones, B. W., and Marc, R. E. Retinal remodeling during retinal degeneration. *Exp. Eye Res. 81* (Aug 2005), 123–137.

[49] Jones, B. W., Watt, C. B., Frederick, J. M., Baehr, W., Chen, C. K., Levine, E. M., Milam, A. H., Lavail, M. M., and Marc, R. E. Retinal remodeling triggered by photoreceptor degenerations. *J. Comp. Neurol. 464* (Sep 2003), 1–16.

[50] Jones, B. W., Watt, C. B., and Marc, R. E. Retinal remodelling. *Clin Exp Optom 88* (Sep 2005), 282–291.

[51] Jurrus, E., Hardy, M., Tasdizen, T., Fletcher, P., Koshevoy, P., Chien, C.-B., Denk, W., and Whitaker, R. Axon tracking in serial block-face scanning electron microscopy. *Medical Image Analysis 13*, 1 (Feb 2009), 180–188.

[52] Jurrus, E., Paiva, A., Watanabe, S., Anderson, J., Jones, B., Whitaker, R., Jorgensen, E., Marc, R., and Tasdizen, T. Detection of neuron membranes in electron microscopy images using a serial neural network architecture. *Medical Image Analysis 14*, 6 (2010), 770–783. DOI: 10.1016/j.media.2010.06.002.

[53] Jurrus, E., Paiva, A., Watanabe, S., Whitaker, R., Jorgensen, E., and Tasdizen, T. Serial neural network classifier for membrane detection using a filter bank. In *Proc. Workshop on Microscopic Image Analysis with Applications in Biology* (2009).

[54] Jurrus, E., Tasdizen, T., Watanabe, S., Davis, M., Jorgensen, E., and Whitaker, R. Semi-automated reconstruction of the neuromuscular junctions in the c. elegans. In *Workshop on Microscopic Image Analysis with Applications in Biology, MICCAI* (September 2008).

[55] Jurrus, E., Whitaker, R., Jones, B., Marc, R., and Tasdizen, T. An optimal-path approach for neural circuit reconstruction. In *Proceedings of the 5th IEEE International Symposium on Biomedical Imaging: From Nano to Macro* (2008), pp. 1609–1612.

[56] Knott, G., Marchman, H., Wall, D., and Lich, B. Serial section scanning electron microscopy of adult brain tissue using focused ion beam milling. *The Journal of Neuroscience 28*, 12 (2008), 2959–2964.

[57] Koyama, R., Yamada, M. K., Fujisawa, S., Katoh-Semba, R., Matsuki, N., and Ikegaya, Y. Brain-derived neurotrophic factor induces hyperexcitable reentrant circuits in the dentate gyrus. *J. Neurosci. 24* (Aug 2004), 7215–7224.

[58] Kremer, J. R., Mastronarde, D. N., and McIntosh, J. R. Computer visualization of three-dimensional image data using imod. *Journal of Structural Biology 116*, 1 (1996), 71 – 76.

[59] Kumar, R., Reina, A. V., and Pfister, H. Radon-like features and their application to connectomics. In *MMBIA* (2010).

[60] Lebedev, M. A., and Nicolelis, M. A. Brain-machine interfaces: past, present and future. *Trends in Neurosciences 29*, 9 (2006), 536 – 546.

[61] Leng, Z., Korenberg, J. R., Roysam, B., and Tasdizen, T. Automatic markup of neural cell membranes using boosted decision stumps. In *Proceedings of the 6th IEEE International Symposium on Biomedical Imaging* (2011).

[62] Leung, T., and Malik, J. Representing and recognizing the visual appearance of materials using three-dimensional textons. *Int. J. Comput. Vision 43*, 1 (2001), 29–44.

[63] Macke, J., Maack, N., Gupta, R., Denk, W., Schölkopf, B., and Borst, A. Contour-propagation algorithms for semi-automated reconstruction of neural processes. *Journal of Neuroscience Methods 167* (2008), 349–357.

[64] Mahamud, S., Williams, L., Thornber, K., and Xu, K. Segmentation of multiple salient closed contours from real images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 25*, 4 (April 2003), 433–444.

[65] Marc, R. E., Jones, B. W., Anderson, J. R., Kinard, K., Marshak, D. W., Wilson, J. H., Wensel, T., and Lucas, R. J. Neural reprogramming in retinal degeneration. *Invest. Ophthalmol. Vis. Sci. 48* (Jul 2007), 3364–3371.

[66] Marc, R. E., Jones, B. W., Watt, C. B., and Strettoi, E. Neural remodeling in retinal degeneration. *Prog Retin Eye Res 22* (Sep 2003), 607–655.

[67] Marc, R. E., Jones, B. W., Watt, C. B., Vazquez-Chona, F., Vaughan, D. K., and Organisciak, D. T. Extreme retinal remodeling triggered by light damage: implications for age related macular degeneration. *Mol. Vis. 14* (2008), 782–806.

[68] Martone, M. E., Tran, J., Wong, W. W., Sargis, J., Fong, L., Larson, S., Lamont, S. P., Gupta, A., and Ellisman, M. H. The cell centered database project: An update on building community resources for managing and sharing 3d imaging data. *Journal of Structural Biology 161*, 3 (2008), 220 – 231. The 4th International Conference on Electron Tomography, The 4th International Conference on Electron Tomography.

[69] Medioni, G., Lee, M.-S., and Tang, C.-K. *Computational Framework for Segmentation and Grouping*. Elsevier Science Inc., New York, NY, USA, 2000.

[70] Minsky, M. Microscopy apparatus. U.S. Patent number 301467.

[71] Mishchenko, Y. Automation of 3d reconstruction of neural tissue from large volume of conventional serial section transmission electron micrographs. *J Neurosci Methods* (Sept 2008).

[72] Mishchenko, Y., Hu, T., Spacek, J., Mendenhall, J., Harris, K. M., and Chklovskii, D. B. Ultrastructural analysis of hippocampal neuropil from the connectomics perspective. *Neuron 67* (2010), 1009–1020.

[73] Namba, K., Sugihara, I., and Hashimoto, M. Close correlation between the birthdate of purkinje cells and the longitudinal compartmentalization of the mouse adult cerebellum. *The Journal of Comparative Neurology* (2011), n/a–n/a.

[74] Nokia. Qt: Cross-platform application and ui framework. http://qt.nokia.com.

[75] Paiva, A., Jurrus, E., and Tasdizen, T. Using sequential context for image analysis. In *Pattern Recognition (ICPR), 2010 20th International Conference on* (aug. 2010), pp. 2800 –2803.

[76] Peng, Y. W., Hao, Y., Petters, R. M., and Wong, F. Ectopic synaptogenesis in the mammalian retina caused by rod photoreceptor-specific mutations. *Nat. Neurosci. 3* (Nov 2000), 1121–1127.

[77] Peters, A., and Feldman, M. L. The projection of the lateral geniculate nucleus to area 17 of the rat cerebral cortex. i. general description. *Journal of Neurocytology 5* (1976), 63–84. 10.1007/BF01176183.

[78] Pizer, S., Johnston, R., Ericksen, J., Yankaskas, B., and Muller, K. Contrast-limited adaptive histogram equalization: speed and effectiveness. *Visualization in Biomedical Computing, 1990., Proceedings of the First Conference on* (May 1990), 337–345.

[79] Pollard, H., Khrestchatisky, M., Moreau, J., Ben-Ari, Y., and Represa, A. Correlation between reactive sprouting and microtubule protein expression in epileptic hippocampus. *Neuroscience 61* (Aug 1994), 773–787.

[80] Pomerleau, D. Knowledge-based training of artificial neural networks for autonomous robot driving. In *Robot Learning*, J. Connell and S. Mahadevan, Eds. Kluwer Academic Publishing, 1993, pp. 19–43.

[81] Principe, J. C., Euliano, N. R., and Lefebvre, W. C. *Neural and Adaptive Systems: fundamentals through simulations.* John Wiley & Sons, 2000.

[82] Rabi, G., and Lu, S. Visual speech recognition by recurrent neural networks. *JEI 7*, 1 (January 1998), 61–69.

[83] Ren, X., Fowlkes, C., and Malik, J. Scale-invariant contour completion using conditional random fields. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on* (Oct. 2005), vol. 2, pp. 1214–1221 Vol. 2.

[84] Ren, X., and Malik, J. A probabilistic multi-scale model for contour completion based on image statistics. In *ECCV '02: Proceedings of the 7th European Conference on Computer Vision-Part I* (London, UK, 2002), Springer-Verlag, pp. 312–327.

[85] Rueckert, D., Sonoda, L., Hayes, C., Hill, D., Leach, M., and Hawkes, D. Nonrigid registration using free-form deformations: application to breast mr images. *Medical Imaging, IEEE Transactions on 18*, 8 (aug. 1999), 712 –721.

[86] Rust, M. J., Bates, M., and Zhuang, X. Sub-diffraction-limit imaging by stochastic optical reconstruction microscopy (storm). *Nature Methods 3*, 10 (August 2006), 793–796.

[87] Schmid, C. Constructing models for content-based image retrieval. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on* (2001), vol. 2, pp. II–39–II–45 vol.2.

[88] Schroeder, W., Martin, K., and Lorensen, B. *The VTK User's Guide*, 11th ed. Kitware, Inc. ISBN 1-930934-19-X, http://www.vtk.org, 2010.

[89] Shashua, A., and Ullman, S. Structural saliency: The detection of globally salient structures using a locally connected network. In *Computer Vision., Second International Conference on* (Dec 1988), pp. 321–327.

[90] Shotton, J., Blake, A., and Cipolla, R. Multiscale categorical object recognition using contour fragments. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 30*, 7 (July 2008), 1270–1281.

[91] Singer, E. A wiring diagram of the brain. *Technology Review* (Nov 2007).

[92] Sommer, C., Straehle, C., Köthe, U., and Hamprecht, F. ilastik: Interactive learning and segmentation toolkit. In *Proceedings of the 6th IEEE International Symposium on Biomedical Imaging* (2011).

[93] Sorra, K. E., and Harris, K. M. Overview on the structure, composition, function, development, and plasticity of hippocampal dendritic spines. *Hippocampus 10* (2000), 501–511.

[94] Soto, G. E., Young, S. J., Martone, M. E., Deerinck, T. J., Lamont, S., Carragher, B. O., Hama, K., and Ellisman, M. H. Serial section electron tomography: A method for three-dimensional reconstruction of large structures. *NeuroImage 1*, 3 (1994), 230 – 243.

[95] Sutula, T. Seizure-Induced Axonal Sprouting: Assessing Connections Between Injury, Local Circuits, and Epileptogenesis. *Epilepsy Curr 2* (May 2002), 86–91.

[96] Tang, C.-K., and Medioni, G. Curvature-augmented tensor voting for shape inference from noisy 3d data. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 24*, 6 (Jun 2002), 858–864.

[97] Tasdizen, T. Principal components for non-local means image denoising. Proceeding of International Conference on Image Processing, 2008.

[98] Tasdizen, T., Jurrus, E., and Whitaker, R. Non-uniform illumination correction in transmission electron microscopy. In *MICCAI Workshop on Microscopic Image Analysis with Applications in Biology* (Sept 2008).

[99] Tasdizen, T., Koshevoy, P., Grimm, B. C., Anderson, J. R., Jones, B. W., Watt, C. B., Whitaker, R. T., and Marc, R. E. Automatic mosaicking and volume assembly for high-throughput serial-section transmission electron microscopy. *Journal of Neuroscience Methods 193*, 1 (2010), 132 – 144.

[100] Tasdizen, T., Koshevoy, P. A., Jones, B. W., Whitaker, R. T., and Marc, R. E. Assembly of three-dimensional volumes from serial-section transmission electron microscopy. In *MIAAB* (2006), pp. 10–17.

[101] Tasdizen, T., Whitaker, R., Marc, R., and Jones, B. Enhancement of cell boundaries in transmission electron microscopy images. In *ICIP* (2005), pp. 642–645.

[102] Tasdizen, T., Whitaker, R., Marc, R., and Jones, B. Enhancement of cell boundaries in transmission electron microscopy images. In *ICIP* (2005), pp. 642–645.

[103] terry, W., El-Baz, F., Harris, W., Juma, C., Kurzweil, R., and Langer, R. The unveiling of the grand challenges for engineering. In *AAAS Meeting* (Feb 2008).

[104] Trupp, M., Scott, R., Whittemore, S. R., and Ibez, C. F. Ret-dependent and -independent mechanisms of glial cell line-derived neurotrophic factor signaling in neuronal cells. *Journal of Biological Chemistry 274*, 30 (1999), 20885–20894.

[105] Tu, Z. Auto-context and its application to high-level vision tasks. *IEEE Conference on Computer Vision and Pattern Recognition* (June 2008), 1–8.

[106] Turaga, S. C., Briggman, K. L., Helmstaedter, M., Denk, W., and Seung, H. S. Maximin affinity learning of image segmentation. *CoRR abs/0911.5372* (2009).

[107] Turaga, S. C., Murray, J. F., Jain, V., Roth, F., Helmstaedter, M., Briggman, K., Denk, W., and Seung, H. S. Convolutional networks can learn to generate affinity graphs for image segmentation. *Neural Computation 22*, 2 (2010), 511–538.

[108] V., K. U., Paiva, A., Jurrus, E., and Tasdizen, T. Automatic markup of neural cell membranes using boosted decision stumps. In *Proceedings of the 6th IEEE International Symposium on Biomedical Imaging* (2009), pp. 1039–1042.

[109] Varma, M., and Zisserman, A. Texture classification: are filter banks necessary? In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on* (June 2003), vol. 2, pp. II–691–8 vol.2.

[110] Varshney, L. R., Chen, B. L., Paniagua, E., Hall, D. H., and Chklovskii, D. B. Structural properties of the ¡italic¿caenorhabditis elegans¡/italic¿ neuronal network. *PLoS Comput Biol 7*, 2 (02 2011), e1001066.

[111] Vazquez, L., Sapiro, G., and Randall, G. Segmenting neurons in electronic microscopy via geometric tracing. In *Proc. of ICIP* (1998), pp. 814–818.

[112] Vu, N., and Manjunath, B. Graph cut segmentation of neuronal structures from transmission electron micrographs. In *Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on* (Oct. 2008), pp. 725–728.

[113] Watanabe, K., Takeishi, H., Hayakawa, T., and Sasaki, H. Three-dimensional organization of the perivascular glial limiting membrane and its relationship with the vasculature: a scanning electron microscope study. *Okajimas folia anatomica japonica 87*, 3 (2010), 109–121.

[114] Wells, G., Venaille, C., and Torras, C. Promising research: Vision-based robot positioning using neural networks. *IVC 14*, 10 (December 1996), 715–732.

[115] White, J., Southgate, E., Thomson, J., and Brenner, F. The structure of the nervous system of the nematode caenorhabditis elegans. *Phil. Trans. Roy. Soc. London Ser. B Biol. Sci. 314* (1986), 1–340.

[116] White, J. Q., Nicholas, T., Gritton, J., Truong, L., Davidson, E. R., and Jorgensen, E. M. The sensory circuitry for sexual attraction in C. elegans males. *Curr. Biol. 17*, 21 (Nov 2007), 1847–57.

[117] Xiao, Y. P., Wang, Y., and Felleman, D. J. A spatially organized representation of colour in macaque cortical area v2. *Nature 421*, 6922 (2003), 535–539.

[118] Yang, H.-F., and Choe, Y. Cell tracking and segmentation in electron microscopy images using graph cuts. In *Biomedical Imaging: From Nano to Macro, 2009. ISBI '09. IEEE International Symposium on* (June 28-july 1 2009 2009), pp. 306 –309.

[119] Zhu, Q., Song, G., and Shi, J. Untangling cycles for contour grouping. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on* (Oct. 2007), pp. 1–8.