RAPID 3D TRACING OF THE MOUSE BRAIN NEUROVASCULATURE WITH LOCAL MAXIMUM INTENSITY PROJECTION AND MOVING WINDOWS

A Dissertation

by

DONG HYEOP HAN

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

August 2009

Major Subject: Computer Science

RAPID 3D TRACING OF THE MOUSE BRAIN NEUROVASCULATURE WITH LOCAL MAXIMUM INTENSITY PROJECTION AND MOVING WINDOWS

A Dissertation

by

DONG HYEOP HAN

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Approved by:

Chair of Committee,	Yoonsuck Choe
Committee Members,	John Keyser
	Scott Schaefer
	Louise C. Abbott
Head of Department,	Valerie E. Taylor

August 2009

Major Subject: Computer Science

ABSTRACT

Rapid 3D Tracing of the Mouse Brain Neurovasculature
with Local Maximum Intensity Projection and Moving Windows.
(August 2009)
Dong Hyeop Han
B.S., Yonsei University, Seoul, Korea;
M.S., Yonsei University, Seoul, Korea
Chair of Advisory Committee: Dr. Yoonsuck Choe

Neurovascular models have played an important role in understanding neuronal function or medical conditions. In the past few decades, only small volumes of neurovascular data have been available. However, huge data sets are becoming available with high throughput instruments like the Knife-Edge Scanning Microscope (KESM). Therefore, fast and robust tracing methods become necessary for tracing such large data sets. However, most tracing methods are not effective in handling complex structures such as branches. Some methods can solve this issue, but they are not computationally efficient (i.e., slow). Motivated by the issue of speed and robustness, I introduce an effective and efficient fiber tracing algorithm for 2D and 3D data.

In 2D tracing, I have implemented a Moving Window (MW) method which leads to a mathematical simplification and noise robustness in determining the trace direction. Moreover, it provides enhanced handling of branch points. During tracing, a Cubic Tangential Trace Spline (CTTS) is used as an accurate and fast nonlinear interpolation approach.

For 3D tracing, I have designed a method based on local maximum intensity projection (MIP). MIP can utilize any existing 2D tracing algorithms for use in 3D tracing. It can also significantly reduce the search space. However, most neurovascular

data are too complex to directly use MIP on a large scale. Therefore, we use MIP within a limited cube to get unambiguous projections, and we repeat the MIP-based approach over the entire data set.

For processing large amounts of data, we have to automate the tracing algorithms. Since the automated algorithms may not be 100 percent correct, validation is needed. I validated my approach by comparing the traced results to human labeled ground truth showing that the result of my approach is very similar to the ground truth. However, this validation is limited to small-scale real-world data due to the limitation of the manual labeling. Therefore, for large-scale data, I validated my approach using a model-based generator. The result suggests that my approach can also be used for large-scale real-world data.

The main contributions of this research are as follows. My 2D tracing algorithm is fast enough to analyze, with linear processing time based on fiber length, large volumes of biological data and is good at handling branches. The new local MIP approach for 3D tracing provides significant performance improvement and it allows the reuse of any existing 2D tracing methods. The model-based generator enables tracing algorithms to be validated for large-scale real-world data. My approach is widely applicable for rapid and accurate tracing of large amounts of biomedical data.

To my wife, Soyoung Yoon and my parents, Hojung Han and Soonae Juan

ACKNOWLEDGMENTS

My research has been one of the hardest challenges I have ever had to face. Without the helpful and patient support of the following people, this research would not have been completed. I am deeply grateful to them for helping me find a way in my research. First of all, I would like to thank my advisor, Dr. Yoonsuck Choe, for helping my Ph.D. life. He taught and guided me how to solve problems during my research. His knowledge, wisdom, and timely advice inspired and motivated me. I also would like to thank Dr. John Keyser. He always helped me to get good ideas while discussing my research problems. His advice became an important foundation of my approach. I am grateful of Dr. Scott Schaefer for his valuable critiques and guidance during my research. His advice made me look at my research in a broader way. I am also grateful of Dr. Louise C. Abbott for her experimental data and valuable comments for my research. Without her support, test of my approach would not have been completed. I also thank Jaerock Kwon and David Mayerich for their data acquisition effort and valuable discussions. I am also grateful of Timothy Mann, Henry Choi, Jiryang Chung, Huei-Fang Yang, and Choonseog Park for their interest and discussion. I especially thank my wife Soyoung Yoon for her patience and sacrifice. Without her, this research would have been worth nothing. I cannot express enough how grateful I am to her. I am also thankful to our daughter, Jiwon, and our son, Joonwhan, for cheering me up whenever I needed it. My parents, Hojung Han and Soonae Juan, deserve heartfelt thanks for having supported, encouraged, and believed in me. My parents-in-law, Jongbo Yoon and Jungja Jin, should be sincerely thanked for all their encouragement and support. I am also thankful of my brothers, Dongki and Dongkwan, and their families for their deep love and support. I am thankful of my colleagues and bosses in Samsung Electronics Corporation, Sangyong Han, Seungsoo

Oak, Choongmo Ahn, Jaekyung Cho, Younghoon Kim, Jinhyoung Kim, Jintaek Lee, Jongseung Lee, Soohee Park, Byoungyoo Kim, Hansung Kim, Sunghoon Bae, and Hyunyoung Kim for support of my work. This research was supported in part by NIH/National Institute for Neurological Disorders and Stroke grant #R01-NS54252 and by Samsung Electronics Corporation.

TABLE OF CONTENTS

CHAPTER Page

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

FIGURE Page

FIGURE Page

FIGURE Page

CHAPTER I

INTRODUCTION

A. Research objective: Whole-brain neurovascular network reconstruction

Neurovascular models have played an important role in understanding neuronal function and medical conditions [1–5]. The neurovascular network is a good source for analyzing vascular and neuronal disease such as aneurysms, arteriovenous malformations, cavernous malformations, and moyamoya disease [6, 7]. However, most current techniques do not provide the full geometry of the neurovascular network because the data are limited to small volumes (i.e., not from the whole brain), and existing algorithms are not effective enough to automatically extract the structural information from a large volume of data [8].

New data acquisition techniques such as the Knife-Edge Scanning Microscope (KESM) enables acquisition of the complete structures of micro-vascular networks in the brain [9]. The KESM has been invented and developed by the Brain Networks Laboratory (BNL) at Texas A&M University [10, 11]. This high-throughput imaging instrument can generate high-resolution 3D volume data. Due to the KESM, it is possible to obtain unprecedented amounts of sub-cellular level tissue data [12, 13]. The KESM can section whole small animal organs (e.g., a mouse brain) at submicron resolution, generating data at a rate of 180 megabytes/s. The data from a whole mouse brain can be over 2 TB on average. Even though several mouse brain atlases are available [14–16], their z-axis resolutions are not high, which is 50 to 60 times less than that of the KESM. Therefore, it is not easy to precisely trace the micro-vascular network using available mouse brain atlases.

The journal model is *IEEE Transactions on Automatic Control.*

The goals of this research project are to develop a framework for automatic tracing in a rapid and robust way to process 2D/3D data and to validate the tracing algorithms with a model-based validation framework. The main objectives of this research were as follows:

Develop a 2D tracing algorithm. I implemented a rapid and robust 2D tracing algorithm using a moving window. It handles problems in existing 2D tracing algorithms such as improper handling of branches, lack of smoothness of trace, and backtracking.

Develop a 3D tracing algorithm. Based on the 2D tracing algorithm, I developed a fast and robust 3D tracing framework. It is significantly faster than existing 3D tracing algorithms, since it reduces the dimension of the search space from 3D to 2D.

Develop a large-scale validation framework. I designed and implemented a large-scale validation framework. It is based on human-labeled ground truth and on model-based virtual neurovascular models.

B. Background: The knife-edge scanning microscope

In the Brain Networks Laboratory a new data acquisition method has been developed called the Knife-Edge Scanning Microscope (KESM) [9]. Fig. 1(a) is a full view of the Knife-Edge Scanning Microscope. There are three main parts for the KESM.

Knife-edge scanning. There are two main functions of the first main part for the KESM. First, it provides image registration between each slice of the specimen block. Second, it removes the cut slice in order to avoid undesirable events (back-scattering of light and bleaching of fluorescent-stained tis-

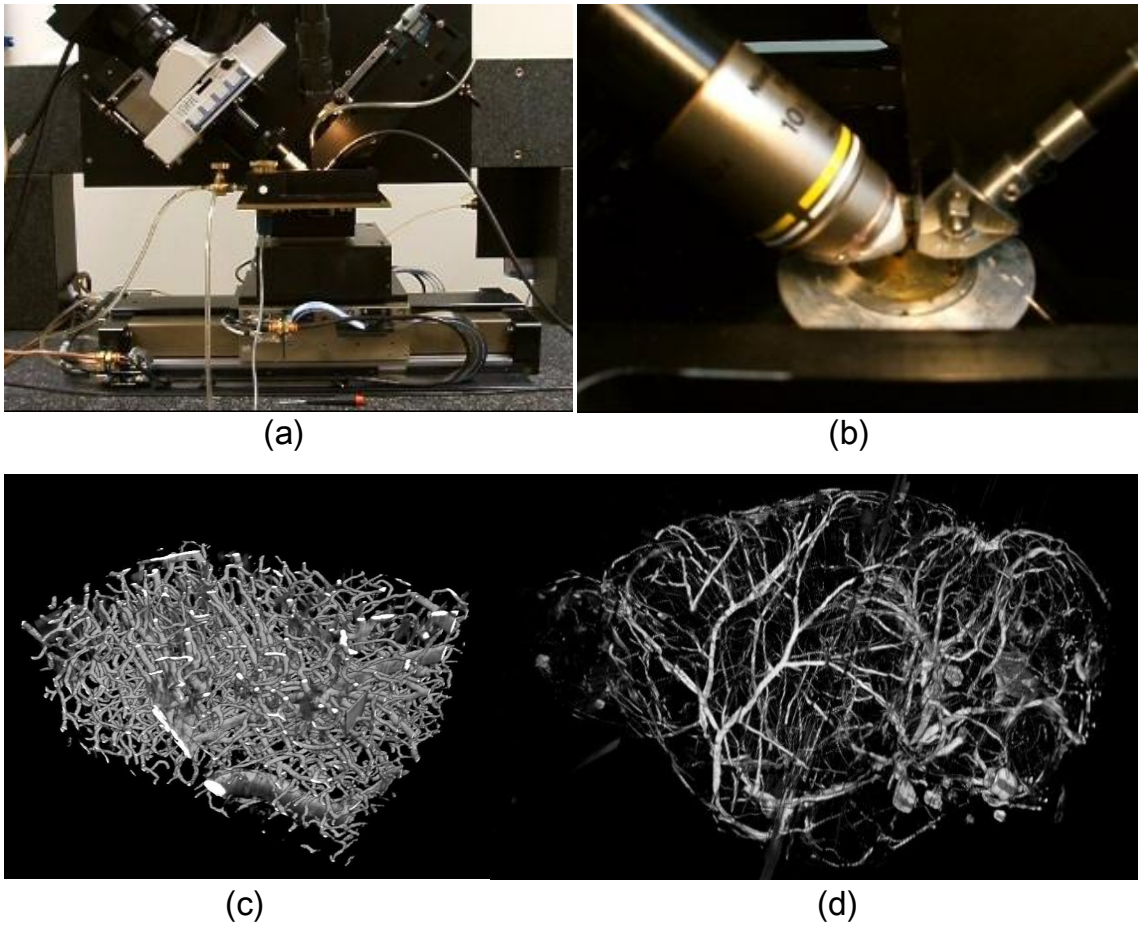


Fig. 1. KESM description. (a) A full view of the Knife-Edge Scanning Microscope. (b) Cutting session: Microscope objective (left) and the diamond knife (right) in place for cutting. The specimen (block in the center) and the knife are fully submerged. (c) and (d) show microvascular data in the whole mouse brain data obtained by the KESM. Approximate volume of (c) is about $500\mu\text{m}\times500\mu\text{m}\times500\mu\text{m}$. Approximate volume of (d) is about $1\text{cm}\times1\text{cm}\times1\text{cm}$.

sue below the knife). The tissue is line-sampled at 300nm resolution, corresponding to the Nyquist sampling interval for an ideal optical resolution of (0.77)(532nm)/(0.80NA)=512nm [13]. A custom diamond knife is used for cutting the block. Each slice section is typically 0.5μm thick. Fig. 1(b) shows a cutting session.

Precision positioning system and ultramicrotome. The Aerotech precision positioning system has been installed on the KESM. This precision positioning system is controllable along three axes and has a specimen block on top.

Image capture system. The microscope objective images the tip of the diamond knife. The objective is aligned perpendicular to the top surface of the knife. Using line-scan cameras, the KESM system captures images from the newly-cut thin section and delivers the captured data to an embedded computer system.

Fig. 1(c) and (d) show microvascular data in the whole mouse brain data obtained by the KESM.

C. Background: Tracing algorithms

In order to extract geometric/topological information from the raw data acquired by techniques like the KESM, we need automated volume data analysis techniques. Before deciding which data processing techniques to use, we should consider the characteristics of the data generated by techniques like the KESM. KESM volume data is densely packed. For example, if all of the slices were connected end-to-end, the resulting ribbon of tissue would stretch 2.9km from a single mouse brain [13]. These densely

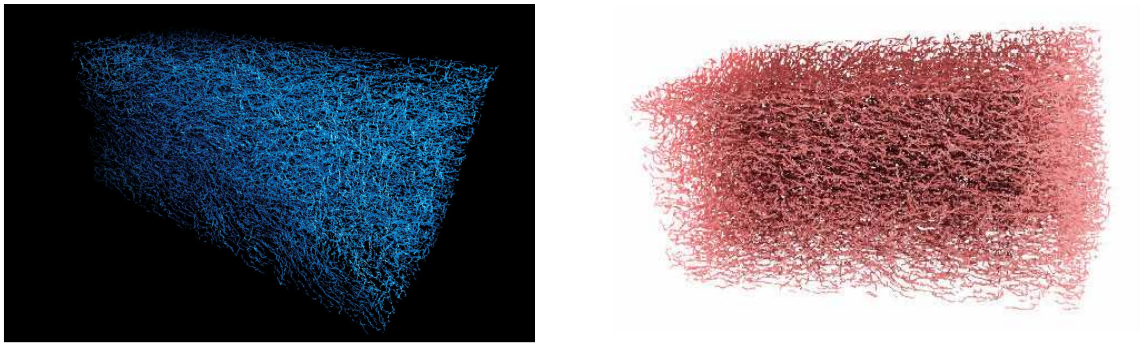


Fig. 2. Two visualizations of the neuronal threads. Note that they are densely packed [17].

packed data can be simply visualized by using an isosurface method as shown in Fig. 2. However, isosurface methods do not provide full, complete morphological models including geometric/topological information such as medial axis, triangles, and vertices. Therefore, we need other rapid and accurate methods to get the morphological model from the KESM volume data.

In my dissertation, I selected tracing method to get the morphological model from the KESM volume data. Tracing algorithm can generate geometric/topological information and can be used instead of segmentation and skeletonization because tracing algorithm can be a way of segmenting or a way of getting a skeleton. This wide applicability made me implement a tracing algorithm to get the morphological model for biomedical data.

D. 2D and 3D tracing algorithm

This dissertation introduces a new algorithm using Moving Windows (MW) and Cubic Tangential Trace Splines (CTTS) to solve simultaneously the issues of speed, accuracy, and branch handling. The MW approach is robust to noise by use of simple intensity

checking of pixels that are located on the rectangular border (edge) of the window. MW is also able to handle branched fibers within the window using the parts of the fiber that intersect the edge of the window. The CTTS approach uses interpolation to help speed up processing. The interpolation skips a majority of pixels within the window and operates on a small number of sampled pixels-of-interest on the spline deduced from the fiber cross section on the window edge so that the computation time is significantly reduced. Furthermore, CTTS can trace the medial axis points with C^1 or G^1 continuity except for branch points. It uses control points for the interpolation and the derivatives at the source points giving accurate results.

The MW-CTTS is suitable for tracing both 2D and 3D data. Even though it works with 3D data, local maximum intensity projection (MIP) can make it faster. The basic idea is to project fibers in the 3D data set onto the 2D planes within a MIP cube and use 2D tracing algorithm on the projections. I tested my algorithm with synthetic data covering curvatures ranging from 0.001 to 0.1 using Frangi et al.'s 2D tracing algorithm to prove that my algorithm works well with well-known tracing algorithms. The result showed that this method is fast and feasible for tracing 3D data. I used Moving Window for local MIP tracing and also applied my algorithm to synthetic data. The result was comparable to the Frangi et al.'s tracing method, but faster.

E. Model-based validation

For validating my tracing algorithm, I designed and implemented two methods (synthetic data or human-labeled validation) and obtained desirable results from the two methods. However, the synthetic data do not guarantee to cover all possible real data types and human-labeled validation is limited to realistic yet small-scaled data.

These limitations gave me the motivation of implementing a model-based generator. The model-based generator yields large-scale neurovascular trees using the parameters extracted from the ground truth of human brain microvascular network [18].

For the validation of my tracing algorithm with the model-based generator, I need to check the two following steps. First, the validation of the model-based generator should be conducted. The model-based generator created the synthetic vascular trees and compared the trees' statistical information to the ground truth for validating itself. Second, the tracing results of the synthetic vascular trees, using my tracing method, should be compared to the centerline information yielded by the model-based generator. If these two steps are successfully satisfied, we can tell that my tracing algorithm can be useful for large-scale realistic data.

F. Significance

Tracing the vascular network is an important scientific task in the following ways.

1. **Biomedicine**: As stated above, the tracing results of vascular networks can provide geometric/topological information. After obtaining the information from many specimens, we can get statistical geometric information. For example, we can see where large primary vessels are located, where they flow, and where critical branches are. This tracing information gives us which part of brain is served by which vessel. By using this geometric/topological information, researchers in biomedicine can easily analyze and diagnose many diseases with neurovascular relations such as Alzheimer's disease [19], strokes [20], and cerebral hemorrhages [21].

2. **Bioengineering**: Using the full geometric/topological information from tracing of vascular network, researchers in bioengineering can develop applications

such as the following. First, a nanoscale medical robot for cleaning blood vessels can be made. This robot can make its way to a target position based on the information such as vessel segment length, the location of branch points, vessel direction, and vessel curvature [22]. Second, reliable artificial body parts can be made. Correct and complete geometric information makes it possible for bio-engineers and tissue-engineers to implement a total artificial heart (TAH) and ventricular assist device (VAD) [23]. Third, a virtual 3D visualization system for vascular network can be made. All statistical geometric/topological information of a vascular network can give a generic vascular network model. This model can be widely utilized in the medical industry [24].

3. **Education**: In a broader perspective, surgeons can conduct virtual operations using the synthetic neurovascular system. It will help the doctors be experts without risking lives in the real world. This synthetic neurovascular system should be made based on the correct geometric/topological information from the tracing result. In addition to this, the statistic geometric information of vascular networks can be a good resource for teaching students in biology.

CHAPTER II

BACKGROUND AND RELATED WORK

In this chapter, I briefly review related works on tracing methods, and compare them with my method. Prior work in tracing can be categorized into two classes: pixel or segmentation-based tracing and vector-based tracing. After reviewing these two classes, I describe the complexity of 3D tracing methods by comparing 3D tracing methods and 2D tracing methods. I conclude the chapter with a discussion of previous work on implementing the model-based generators for validation.

A. Pixel or segmentation-based tracing

The first class of prior work uses pixel or segmentation-based processing [25–40]. This tracing algorithm is applied to every pixel or segment unit on structures that have complex fibers and abundant branches. Chadhuri et al. used a two-dimensional template matching process on two-dimensional images as shown in Fig. 3(a) [25]. Descoteaux et al. presented a segmentation method using Hessian matrix based on scale-space theory [28]. They applied the Hessian matrix on all points of the surface of an ellipsoid to get the accuracy of vessel measurement as shown in Fig. 3(b). Friman et al. used a 2D/3D template for tracing blood vessels [30]. They generated a cylinder template based on the Full Width Half Maximum (FWHM) as shown in Fig. 3(c). Because several operations are required such as template processing and thresholding [41, 42] on every pixel along the edges as shown in Fig. 4(a), it takes quite a long time to trace the structures. Some methods use a priori knowledge about the structures of interest, which makes the process less generalizable to different conditions [43,44]. Fig. 4(b) shows the three primary modes of variance of the corpus callosum training dataset in [43]. In some cases, interesting structures are considered

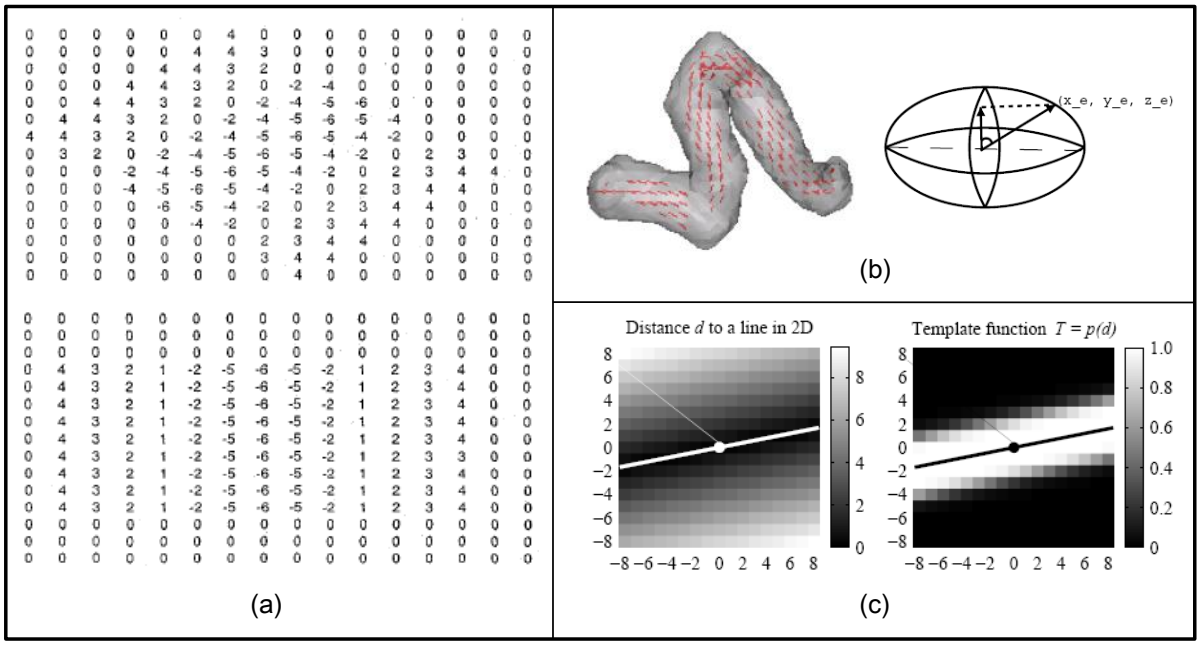


Fig. 3. Pixel based approaches (first group). (a) Chaduri et al. used two-dimensional matched kernels [25]. Two of the 12 different kernels that have been used to detect vessel segments along different directions: the top part shows segments along the 45 degree direction and the bottom part shows segments along the vertical direction. (b) Descoteaux et al. took the accuracy of vessel measurement on all voxels on the ellipsoid [28]. The red vectors on the left image indicates the accuracy of vessel measurement. (c) The vessel template was generated by first calculating the distance to the line spanned by the template (left) and then applying a vessel profile function to this distance (right) [30].

with their adjacent objects using the information such as location, size, orientation, and shape. Integration of this information makes the tracing process slower even though they enhance accuracy [45, 46]. Fig. 4(c) shows an example of Yang et al.'s method [45]. The left image is an MR image showing an axial cross-section of the brain and right image is hand segmentation of the three subcortical brain structures within the image. Using this mutual information, they improved the shape-based deformable active contour model. Moreover, most of them have difficulty in handling

branches.

There are some methods that do handle branches well. For example, Osher et al. introduced level set methods [47] to operate properly on branch points and the algorithm provided robustness. However, their method needs to compute derivatives at every pixel on the isosurface, which makes the algorithm unsuitable for real-time environments as shown in Fig. 4(d). Adalsteinsson et al. [48] presented fast level set methods to address the computational time issue. However, it still takes a lot of processing time even though they used only points close to the curve that are to be traced at every time step [49].

B. Vector-based tracing

The second class is based on vector tracing [11, 50–61]. Can et al. used sixteen directional templates for tracing blood vessels [51]. Right and left templates are used on each direction as shown in Fig. 5(a). Given a seed point, the directional templates estimate the direction of vessels based on the maximum response of the templates. The size of each template is 5×6. Due to the discretization error of the directional templates, a refinement vector is used at each tracing step. After the refinement, the directional templates are used again to find the next direction of vessels until stopping criteria are satisfied. Kofahi et al. extended Can et al.'s 2D templates to 3D templates for tracing blood vessels in 3D [50]. This method is a fully automated three-dimensional (3-D) tracing of neurons that are imaged by fluorescence confocal microscopy. This approach works by recursively following the neuronal topology, using a set of $4 \times N^2$ large directional kernels (e.g., N= 32), guided by a generalized 3-D cylinder model. Since the centerlines are of primary interest, the 3-D extension can be accomplished by four rather than six sets of kernels as shown in Fig. 5(b). Shen

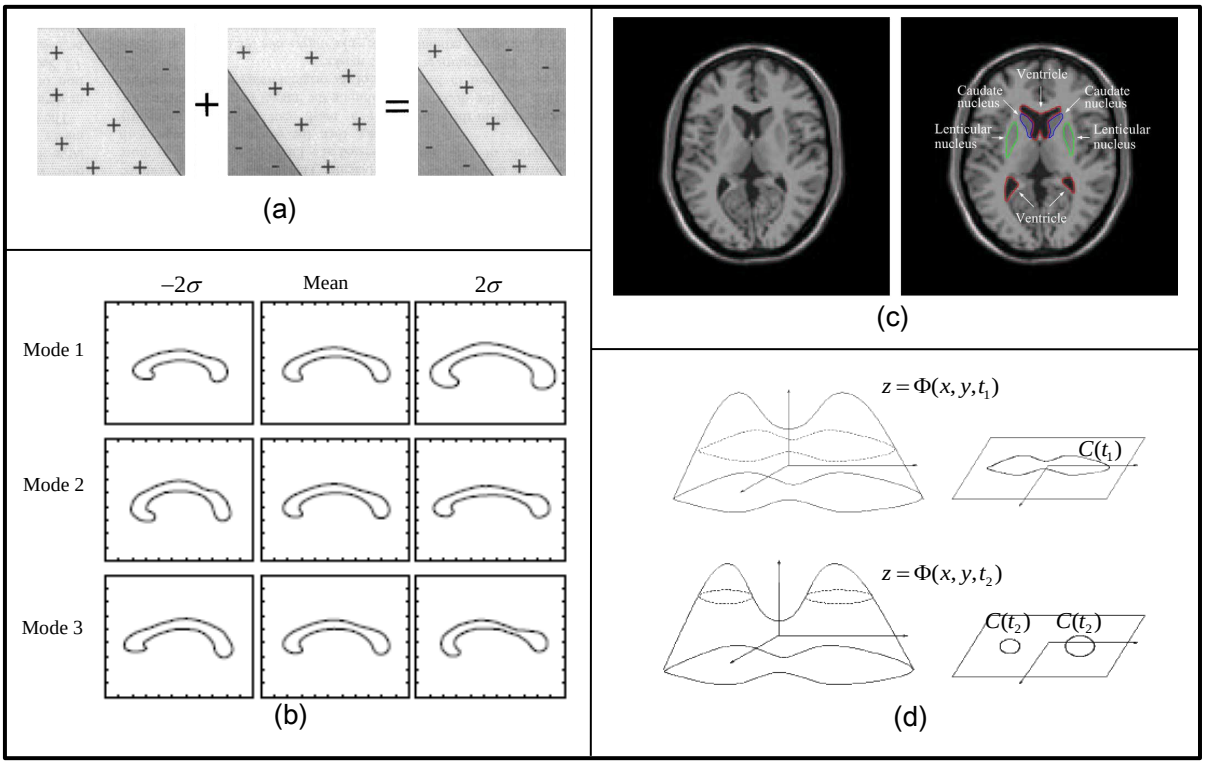


Fig. 4. Pixel based approaches (second group). (a) Polli et al. used template processing on every pixel along the edges [41]. (b) Three primary modes of the corpus callosum. Leventon et al. used a priori knowledge regarding the object [43]. (c) The example of mutual information for medical image segmentation [45]. (d) Level set methods which require derivatives at every pixel on the isosurfaces [47].

et al. used grid lines to find the direction and center point of vessels as shown in Fig. 5(c). In this method, the vascular tracing computations are scheduled to maximize the quality of the partial tracing results within a predefined frame cycle. In each grid line, one frame cycle is used to find the direction and center point of vessels.

Frangi et al. used the Gaussian kernels on the center point on each tracing step to find the direction of vessels [59]. They obtained the eigenvalue closest to zero. The eigenvector corresponding to the eigenvalue is the direction of vessel. Fig. 6(a) shows the Gaussian kernel and the direction of vessels based on the eigenvalue. Aylward et al. used the tangent vectors on the medial axis point of each step as shown in Fig. 6(b) [61]. Given a seed point, the point will extend in the positive and negative tangent directions. Each direction is traversed independently. For the default implementation, at the i^{th} point traversed on a medial axis point, the approximate tangent direction is defined as the eigenvector corresponding to the minimal eigenvalue of the Hessian matrix. The direction of medial axis traversal is maintained by multiplying by the sign of the dot-product of and the previous tangent direction. Carrillo et al. performed local segmentation within the sphere using a clustering algorithm based on both geometric and intensity information [60]. Fig. 6(c) shows the segmentation and tracing of Carrillo et al.'s approach. These methods use vector-based systems to trace curvy fibers containing branches. They are fast compared to the first class of algorithms above due to the fact that every pixel does not need to be checked. However, they face four general difficulties. One issue is that the tracing process is not able to continue tracing at branch points or only one branch is selected for continued tracing. As shown in Fig. 7(a), they select only one branch using some method such as convolution even though there are several branches at a branch point. Haris et al. addressed this issue using a model-based algorithm. However, Haris et al.'s algorithm yields another issue in branch handling (Fig. 7(c)). Moreover, these vector-based

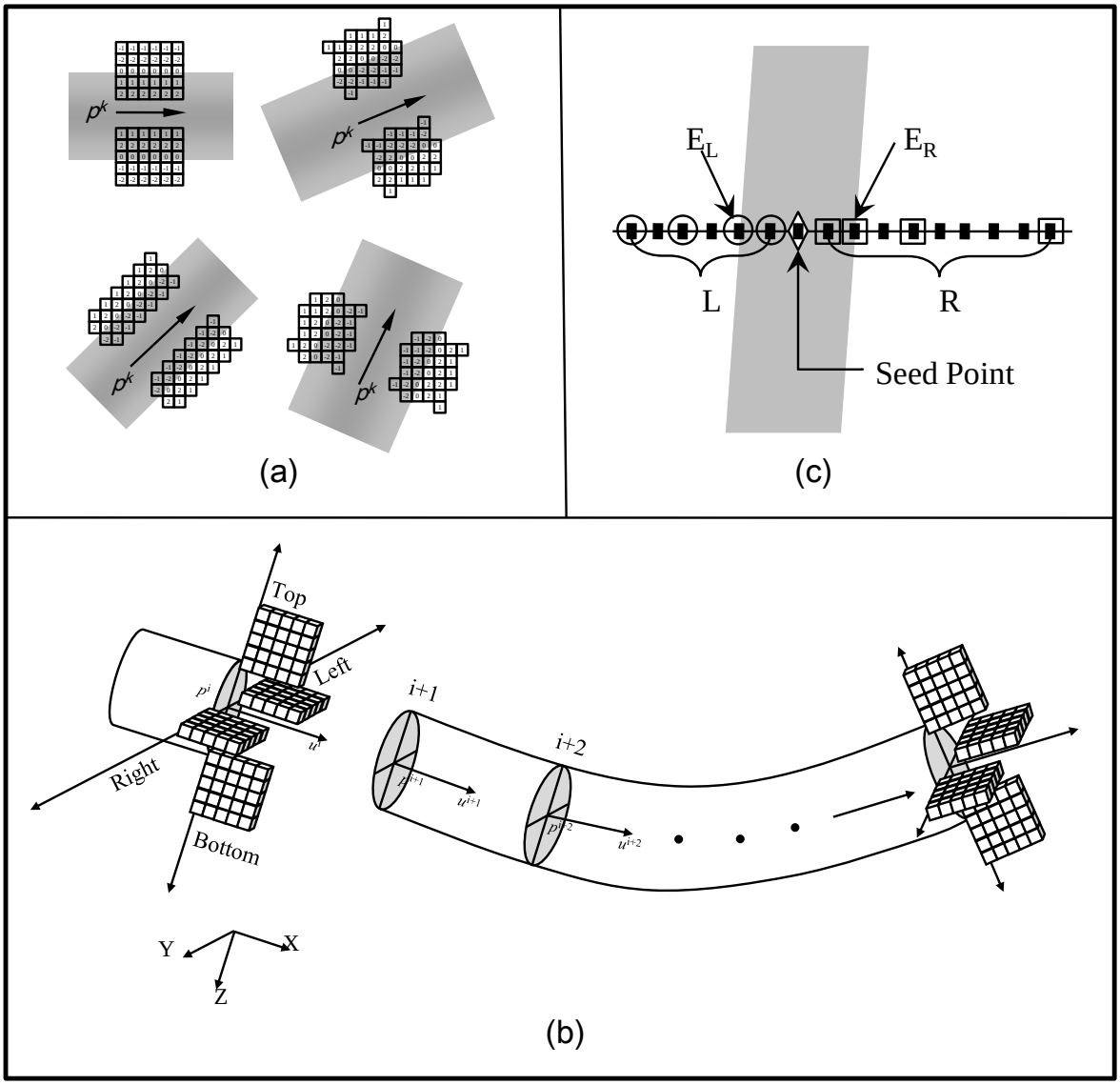


Fig. 5. Vector based approaches (first group). (a) Can et al. used two-dimensional matched kernels to find the direction of blood vessels [51]. (b) Kofahi et al. extended Can et al.'s kernels for 3D vessel tracing [50]. Four directional kernels are used for finding the direction of blood vessels in 3D. (c) Shen et al. obtained vessels using grid lines and derived the center points [54].

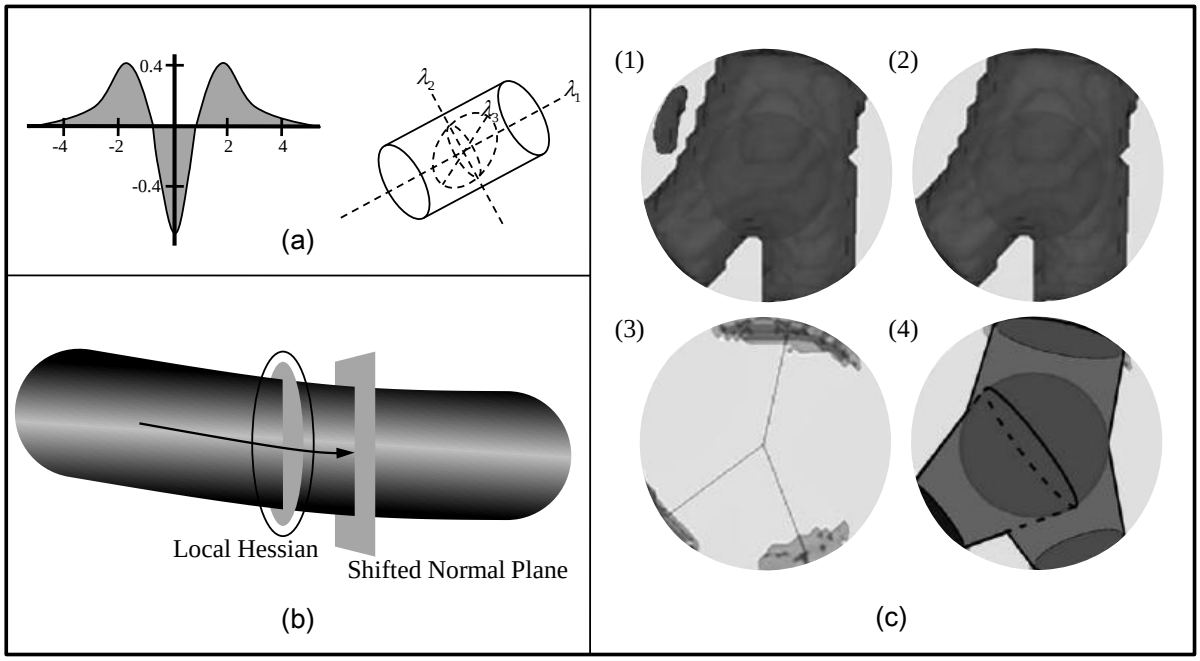
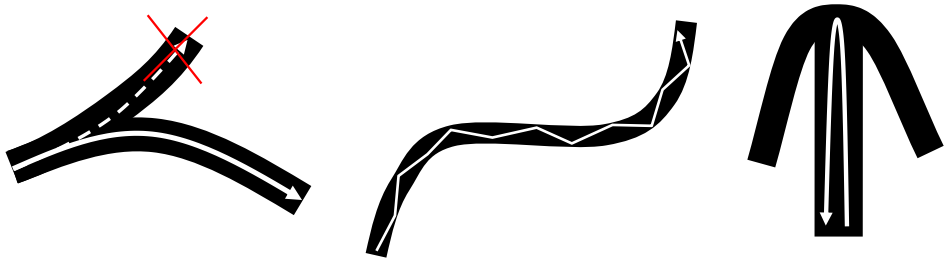


Fig. 6. Vector based approaches (second group). (a) Frangi et al. used eigenvectors of the second order derivative of a Gaussian kernel to find the direction of vessels [59]. (b) Aylward et al. used a step-maximize procedure in order to iteratively traverse a centerline. Eigenvectors of the local Hessian matrix approximate the tangent and normal directions. The shifted normal plane bounds the search for the next ridge point. This method is based on Frangi et al.'s method [62]. (c) Carrillo et al. used a cell to trace vessels. (1) segmentation, (2) removal of non-connected 3D components, (3) connected components on the surface of the cell, and (4) model based on the connected components and on the corresponding distance maps [60].



(a) (b) (c)

Fig. 7. Tracing issues. (a) Ignored branch. (b) Jaggedness in medial axis tracing. (c) Incorrect backing-up at a branch.

tracing methods do not guarantee smoothness (Fig. 7(b)). A detailed overview of existing segmentation algorithms for tubular structures can be found in [63].

C. 3D vs. 2D tracing

Many tracing algorithms have been developed to extract the neurovascular network from 2D images. Most of them have been extended to 3D tracing methods. Frangi et al. proposed 2D and 3D tracing algorithms using a Hessian filter after convolving the input data with a Gaussian kernel [64]. Due to the second order partial derivatives in the Hessian matrix, Frangi et al.'s 3D method has a much higher computational complexity than their 2D method. Friman et al. presented a multi-dimensional approach using vessel profile based on distance function [30]. In 2D, this method uses rotation matrix operations to find the vessel direction, which requires searching only a 2D plane space. However, in 3D, the rotation matrix operation requires searching a 3D spherical space, taking $O(n^2)$ processing time instead of the $O(n)$ time of the 2D method, where n is the number of sampled directions as shown in Fig. 8. Haris et al. presented a 2D tracing method with a circular window to extract a vascular

network [53], and Carrillo et al. extended it to 3D space using spheres [60]. In 3D, the sphere uses connected components to find the vessel direction while only a first order derivative is required on the circular window's edge in 2D. Searching the space in the sphere needs $O(r^2)$ processing time while a circular window requires $O(r)$, where r is the radius. Clearly, most 2D algorithms are more efficient than their 3D versions. In this work, I will provide a local Maximum Intensity Projection (MIP) framework which makes 2D algorithm directly applicable to 3D data. My results show that the computation time of 3D tracing can be reduced by over 60% using this method.

D. Model-based generator

There have been several prior methods to generate a synthetic neurovascular network or synthetic pulmonary arterial trees.

Eberhard et al. introduced a tool (NeuGen) for the generation of realistic morphology of cortical neurons in 3D. NeuGen used parameter distributions derived from anatomical data to construct synthetic neurons of different morphological classes. Fig. 9(a) and (b) show various results of generated neurons and networks using NeuGen. This tool is only for generating synthetic cortical neurons.

Mandegar et al. have shown that the typical cast of a small segment of arterial tree in human lung, as shown in Fig. 9(c) [65]. They also showed the structure of the vascular tree using three schemes such as the Weibel Model [66], the Strahler model [67], and the Diameter-defined Strahlers system [68] for describing this complex structure. All these three schemes were implemented for pulmonary arterial trees by Weibel [66], Singhal et al. [67], and Huang et al. [68].

Ascoli et al. developed a tool using the stochastic L-systems as algorithms for generating virtual neurons as shown in Fig. 9(d) [69, 70]. Similar stochastic growth

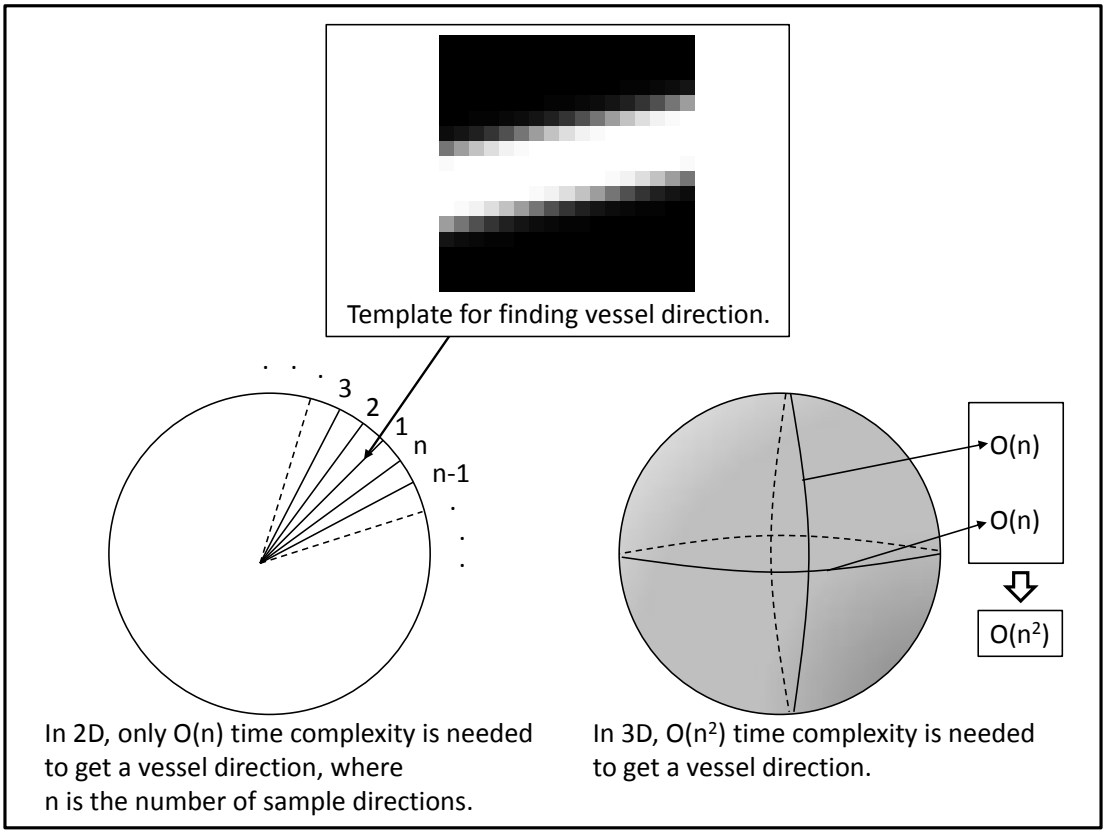


Fig. 8. 2D vs. 3D tracing algorithm. Friman et al.'s tracing algorithm [30]. Top: Template is shown for finding the vessel direction. Lower left: Friman et al.'s 2D tracing has an $O(n)$ time complexity, where n is the number of sampled directions. Lower right: Friman et al.'s 3D tracing has an $O(n^2)$ time complexity, where n is the number of sample directions.

models of dendrite were used by [71, 72]. Examples of mechanical growth models of dendrite were given by [73, 74]. While the algorithms in the synthetic models above are directly or indirectly based on knowledge-based information, growth models need more parameters to obtain an identical neurovascular model. Currently, Ascoli et al.'s algorithm provides a good approach to a high degree of accuracy. However, no prior work has provided full degree of accuracy for human brain microvascular network [75].

In order to obtain an accurate model of the human brain microvascular network, I focused on the analysis of neurovascular networks and implemented a model-based generator using several parameters extracted from the analysis of human brain data as shown in Fig. 10 [18]. Using the model-based generator, I produced an accurate and full neurovascular tree and used it for validating my tracing algorithms. In terms of parameter estimation, comparisons between the results given by 2D and 3D morphometry have shown that only 3D morphometry is able to obtain reliable data on highly complex vascular networks and that 2D morphometry should be limited to the analysis of flat two-dimensional vascular networks to prevent underestimation of parameters [76].

E. Conclusion

In this chapter, I reviewed tracing methods which can be categorized into two classes. The first class uses pixel or segmentation-based processing. The methods within this class yield fewer errors but are slow because they consider volumetric information with many computational steps. The second class is based on vector tracing. They are fast, but have several problems such as branch handling, smoothness, and incorrect backing-up at a branch. I also showed that existing methods have limitations in extending into 3D in terms of complexity and provided motivation for a method based

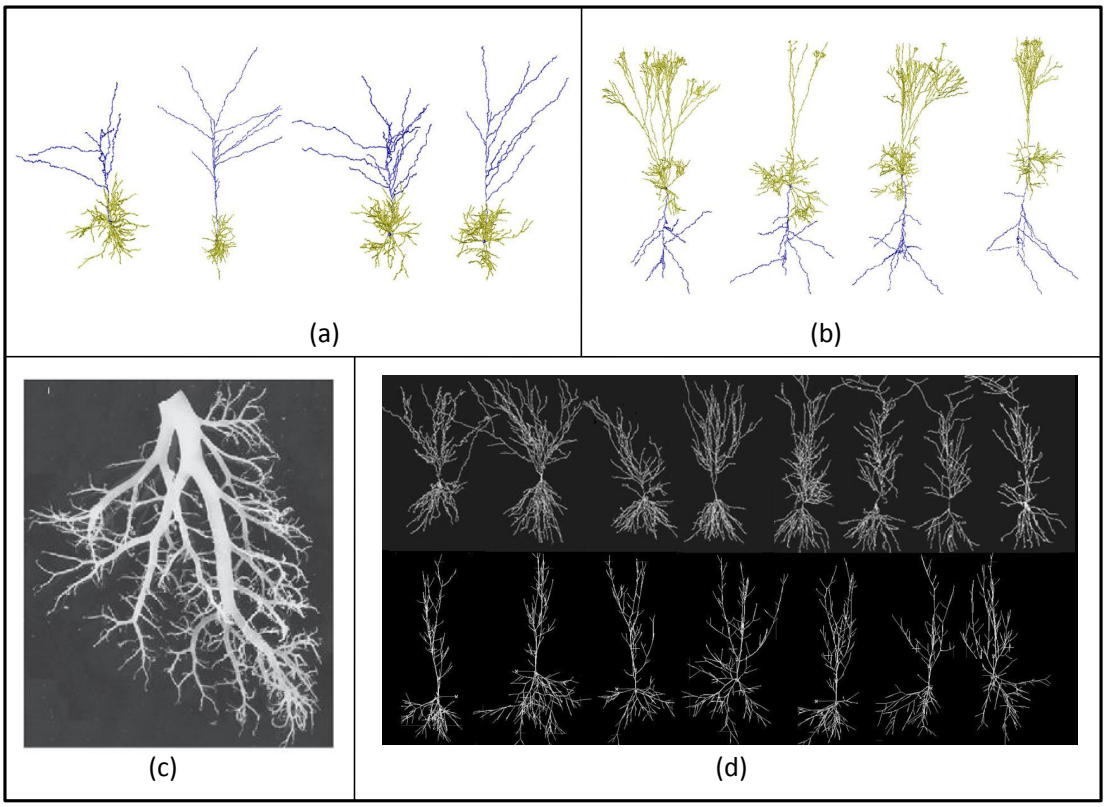


Fig. 9. Synthetic neurons and vascular networks. (a) Four different neurons with 12 dendrites. These are shown by OpenDX visualization tool [77]. (b) Four different pyramidal cells (pyramidal neurons) with 8 dendrites [77]. (c) A small segment of arterial tree in human lung for pulmonary modeling [65]. (d) Top row shows real hippocampal pyramidal cells (pyramidal neurons) obtained from an experimental archive [78]. Bottom row shows virtual neurons generated by Tamori's algorithm [79]. Virtual neurons are not the same as real neurons. However, we can tell that they are in the same morphological class [69].

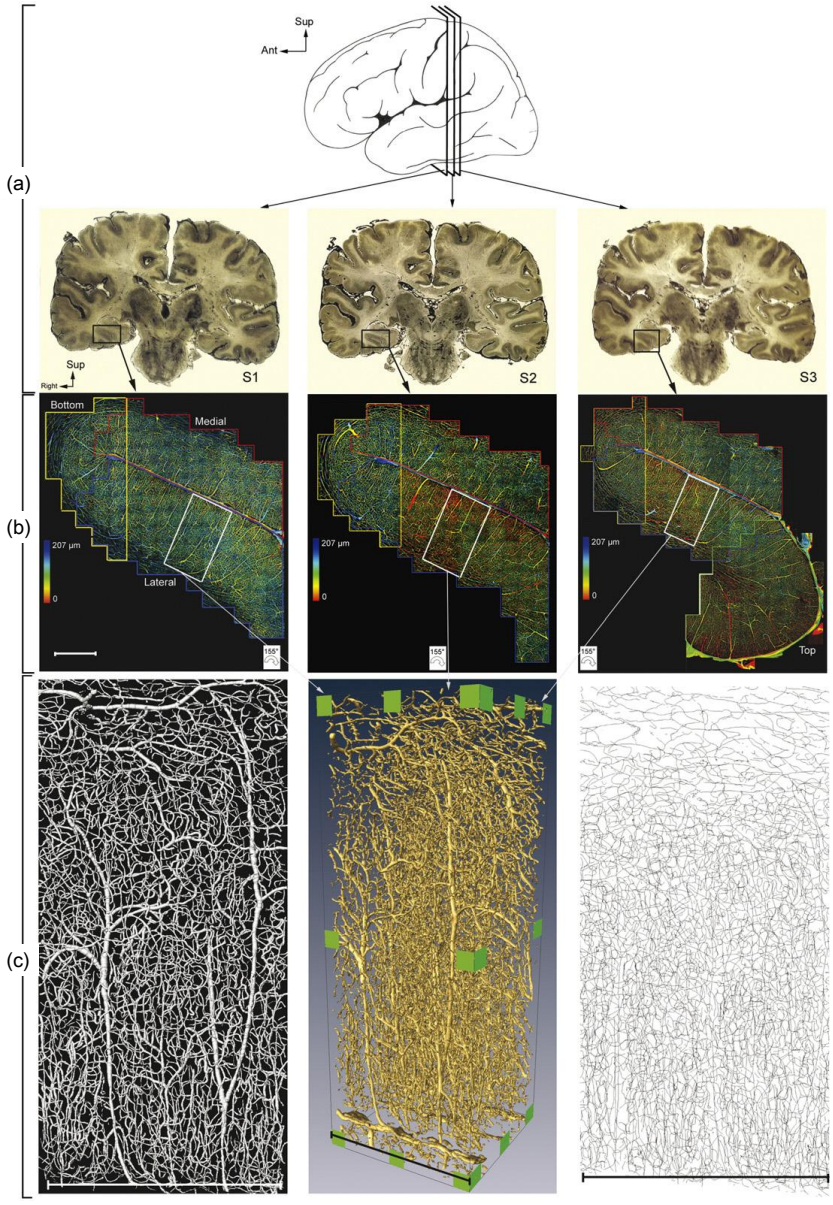


Fig. 10. The acquisition of human microvascular networks in the cerebral cortex. (a) Three adjacent coronal sections (S1, S2 and S3) of India ink-injected human brain used for data acquisition. The region of interest is located around the collateral sulcus in the right temporal lobe. (b) Depth coded projection of the zones is reconstructed by confocal microscopy. (c) 3D volume rendering of a selected zone after alignment of the 3 sections (bottom centre). Vascular skeleton of the same zone on S1 (bottom right). 3D reconstruction of this part of S1 skeleton (bottom left). Scale bar=1 mm. [80].

on local maximum intensity projection which utilizes 2D methods for 3D data. I also reviewed prior methods for generating synthetic vasculatures in the lung or dendritic and axonal arbors in neurons suggesting a model-based generator for microvascular network in human brain. In the four following chapters, a tracing method and a model-based generator motivated by these previous works will be described in detail.

CHAPTER III

2D METHOD FOR TRACING

In this chapter, my 2D tracing method is described. First, I explain a moving window (MW) approach to find the direction of fiber at each tracing step. Second, in order to correctly handle a branch point, I show how to trace the branch point in a geometrically correct manner. Next, I present a cubic tangential trace spline (CTTS) for tracing medial axis using four control points. After that, stopping criteria is overviewed. Finally, the 2D tracing results are shown to see what problems are solved and how good my algorithm is compared to other existing methods.

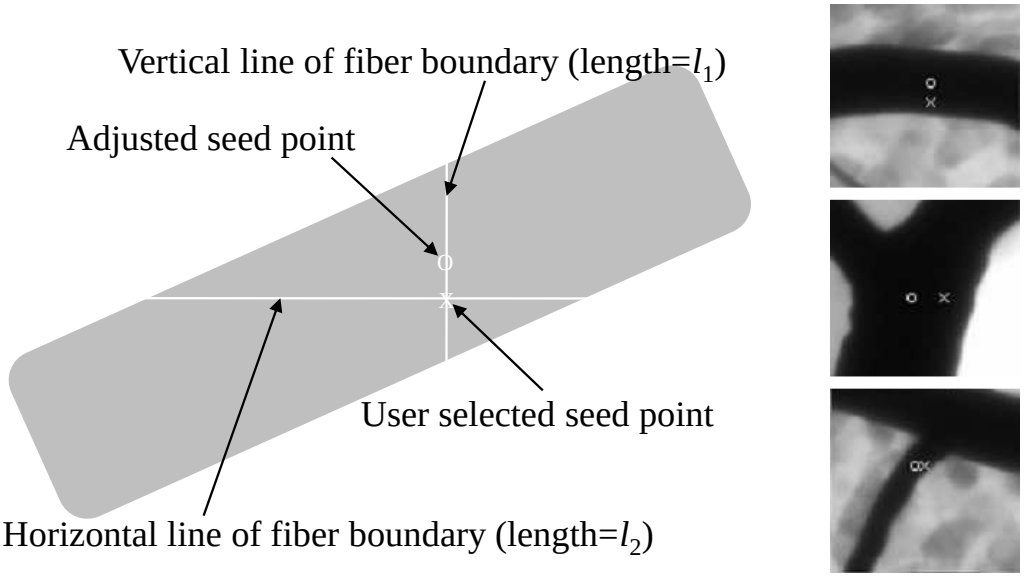
A. Moving window (MW) for selecting fiber direction

My algorithm uses a moving window to keep track of the trace. Initial moving window construction requires a seed point from the user's manual input. Fig. 11(a) shows how to adjust a user selected seed point. The user selected seed point can have two fiber boundary lines (vertical line in vertical direction and horizontal line in horizontal direction as shown in Fig. 11(a)). The fiber boundary can be obtained by some predefined threshold, which is dependent on the input data. If the length of the vertical line is shorter than the length of the horizontal line, the user selected seed point will be adjusted to the middle of the vertical line of the fiber boundary. If the length of the horizontal line is shorter than the length of the vertical line, the user selected seed point will be adjusted to the middle of the horizontal line of the fiber boundary. Fig. 11(b) shows the seed point adjustment in real data. The creation of a moving window centered at the adjusted seed point needs an appropriate

determination of the window's side length r:

$$\begin{aligned} r &= l \times \epsilon \\ l &= \min(l_1, l_2) \end{aligned} \tag{3.1}$$

where l is the shorter length between the length of vertical line of fiber boundary (l_1) and the length of horizontal line of fiber boundary (l_2) as shown in Fig. 11(a). ϵ is a scaling factor usually set between 1.1 and 1.5 based on my experimental results. If experimental data has almost constant fiber radius, I set ϵ to be 1.1. If experimental data has various fiber radii, I set ϵ to be 1.5. In case of the various fiber radii, if I set ϵ to be 1.1, the moving window sometimes cannot easily detect the fiber's cross section (FCS) which is defined in the next paragraph.



(a) (b)

Fig. 11. Seed point adjustment. (a) Illustration. (b) Seed point adjustment in real data ("×" marks the user selected seed point, and "○" the adjusted seed point).

Fig. 12 shows the basic process of my algorithm. A Gaussian filter is applied along the MW border, resulting in a pixel intensity profile sampled only near the

MW border. Using the intensity profile with a predefined threshold, I can detect the fiber's cross section (FCS) and the boundaries of the FCS (Fig. 12). The center of the next trace step's MW is defined as the middle of the boundaries of FCS as shown in Fig. 12. In subsequent steps, newly generated MWs are given a different side length, based on the length of the current FCS:

$$r_{i+1} = l_{FCS_i} \times \epsilon \tag{3.2}$$

where i is the trace step's index and ϵ is the same as for (3.1). r_{i+1} is the side length of the $(i+1)^{th}$ step's MW and l_{FCS_i} is the length of the FCS in the i^{th} trace step as shown in Fig. 13.

The FCS which contains the previous trace result does not generate a new MW in order to avoid backtracking. Finally, I can shift the moving window to the next trace point and change the side length of the moving window.

B. Branch smoothness while maintaining continuity

Interpolation can make the tracing algorithm faster and more accurate if there are correct control points. Here, I use a branch smoothing algorithm on branched fibers to avoid misleading traces as shown in Fig. 14(b) and an interpolation algorithm on unbranched fibers to maintain C^1 or G^1 continuity. The interpolation algorithm is described in section C of chapter III. The branched fibers can be identified when a MW detects three or more FCSs. In order to obtain a correct trace on branched fibers, I should find a parent point on the current MW as shown in Fig. 14(a). The parent point is the point from which the trace starts and produces a visually satisfying result (the left one in Fig. 14(b)). The determination of the parent point is described in the next paragraph. However, some branch points do not give good information

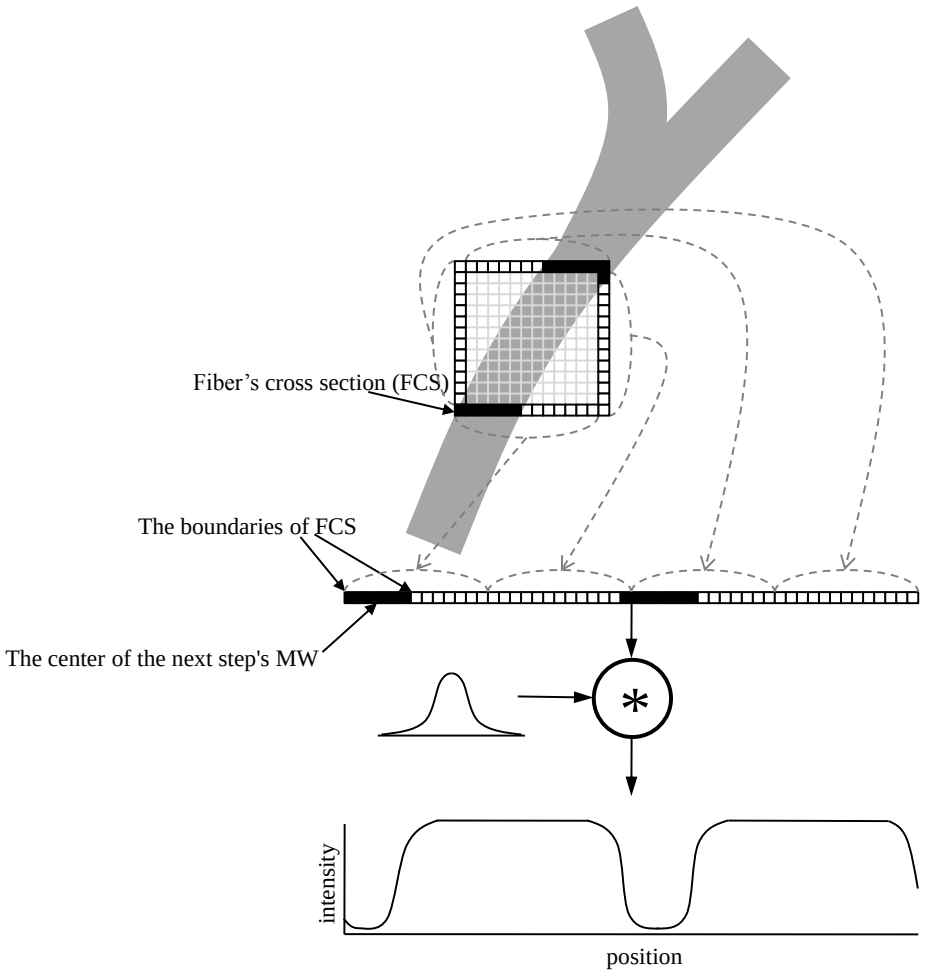


Fig. 12. Overview of moving window. Using the intensity profile with a predefined threshold, the fiber's cross section (FCS) and the boundaries of FCS can be detected. The center of the next step's MW is defined as the middle of the boundaries of FCS.

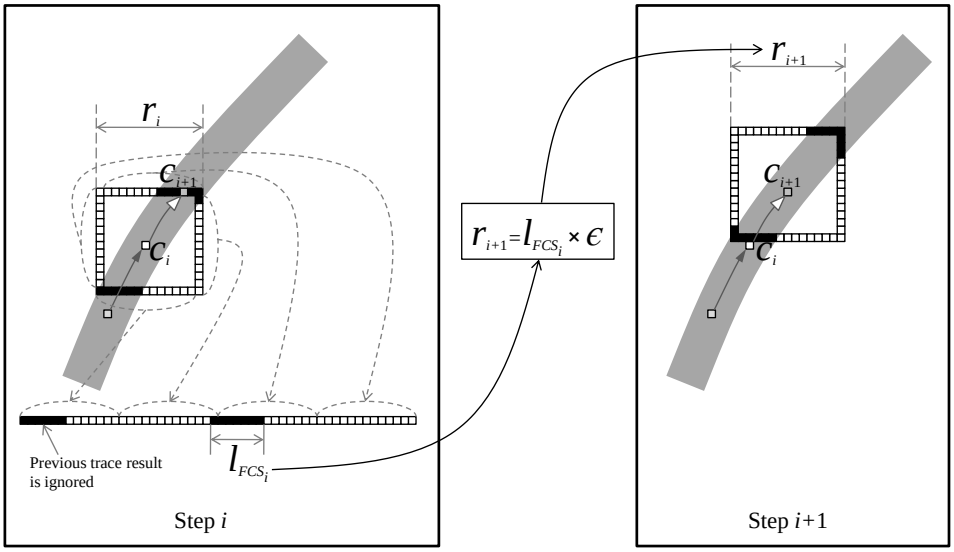


Fig. 13. Calculation of the side length of MW. r_i is the side length of the i^{th} step's MW and c_i is the center of the i^{th} step's MW. l_{FCS_i} is the length of the FCS in i^{th} step. ϵ is the same as (3.1). r_{i+1} is the side length of the $(i+1)^{th}$ step's MW and c_{i+1} (gray pixel) is the center of the $(i+1)^{th}$ step's MW. r_{i+1} is calculated based on l_{FCS_i}.

about which one is the parent point (the right one in Fig. 14(c)). Hereafter, I call this branch an ill-defined branch and the others well-defined branches as shown in Fig. 14(c). In this ill-defined branch case, I simply carry out Bresenham's line drawing to get a visually more satisfying branch trace [81].

In order to find the parent point, I assume that the parent point tends to have the child points on the opposite side of the MW and have a bigger radius than that of the child points. Therefore, the key factors in identifying the parent point are FCS' vector $\vec{v}_i$ and width r_i defined in (3.2). As described in Fig. 15(a), $\vec{v}_i$ is the vector from c to b_i and then normalized as a unit vector $\vec{u}_i$. c is the center point of the current MW and b_i is the ith FCS' center point as defined in Fig. 15(a). In addition to this, I need a weight coefficient $w_{i,j}$ for branch ordering. It is defined as $w_{i,j}$ =min(r_i, r_j).

$w_{i,j}$ causes (3.3) to produce a smaller value when i is on main stem. The parent point b_k is obtained from

$$k = \underset{i\in[1,..,n]}{\operatorname{argmin}} \sum_{j=1,j\neq i}^{n} (w_{i,j}(\vec{u}_i \cdot \vec{u}_j)) \tag{3.3}$$

where n is the number of FCSs detected on the current MW. The above equation has the smallest value when i has the parent point index. Fig. 15(c) shows one example for finding the parent point using the assumption in Fig. 15(b). In this example, (3.3) has the smallest value when i is 3. So, b_3 can be the parent point.

C. Cubic tangential trace spline (CTTS) for tracing medial axis

Cubic Tangential Trace Spline (CTTS) is a piecewise cubic polynomial. In each MW, this piecewise polynomial produces trace results along the medial axis of the fiber. It maintains C^1 or G^1 continuity on the trace results and uses four control points. This means that the polynomial has degree 3 (**cubic**). It also uses the **tangent** information of the previous spline. It is used to **trace** fibers. This is the reason why the piecewise cubic polynomial is called a Cubic Tangential Trace Spline (CTTS).

The Cubic Tangential Trace Spline (CTTS) algorithm interpolates three control points (source, target, and midpoint) and one tangent vector. For example, in order to yield $CTTS_i$, a source (c_i), target (c_{i+1}), and midpoint (s_i) are used as control points and one tangent control vector ($\alpha_i CTTS'_{i-1}(1.0)$) (Fig. 16). c_i and c_{i+1} are source and target points of the i^{th} moving window (MW_i). The midpoint and the tangent vector will be calculated in the following sections.

1. Midpoint calculation

Sun's algorithm is used to get the midpoint [82]. Sun's algorithm works well in detecting medial axis points and it is also robust to noise. However, it is slow due to

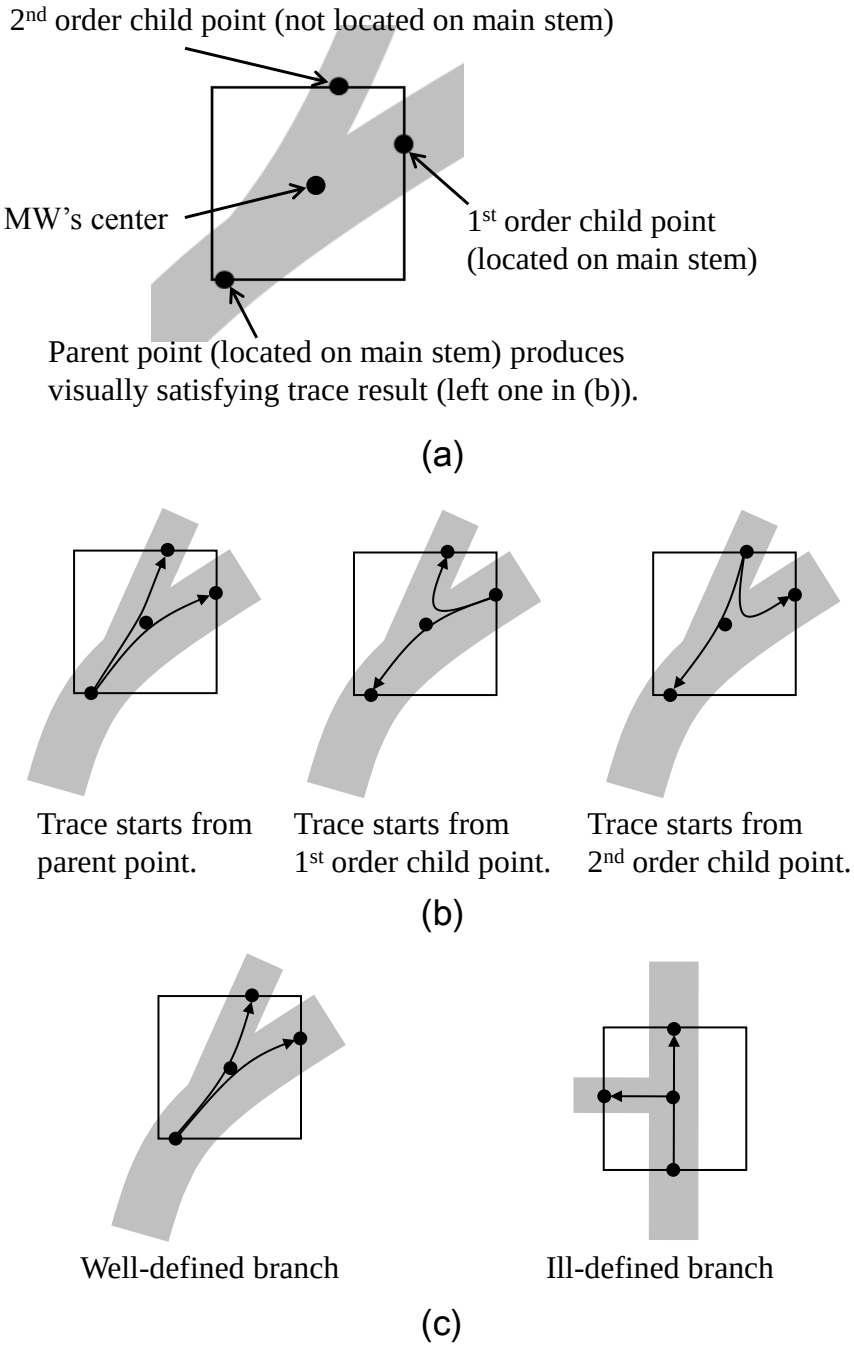
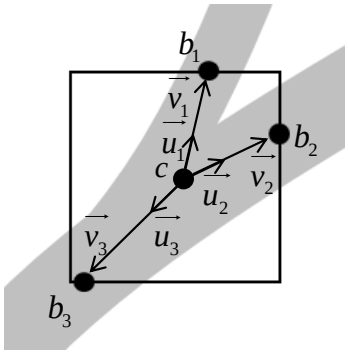


Fig. 14. (a) Definition of parent point and ordered child points. (b) Misleading traces depending on start point when interpolation is used. Only left one has visually satisfying result. (c) Trace results for well-defined branch and ill-defined branch after finding the parent point. If the parent point can be identified correctly, the trace result will produce a visually satisfying result on both well-defined branches and ill-defined branches.



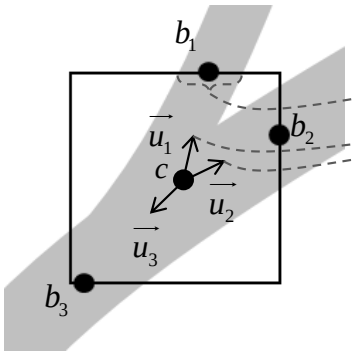
c: center point of current MW

b_i: each FCS' center point

$\vec{v}_i$: vector from c to b_i

$\vec{u}_i$: unit vector of $\vec{v}_i$

(a)



Assumption:

$r_1 = 0.5,\ r_2 = r_3 = 1.0$ for $w_{i,j} = \min(r_i, r_j)$

$\vec{u}_1 \bullet \vec{u}_2 \approx 0.707$

$\vec{u}_2 \bullet \vec{u}_3 \approx -0.980$

$\vec{u}_3 \bullet \vec{u}_1 \approx -0.556$

(b)

From the equation for finding parent point: $\sum_{j=1, j\neq i}^{3} w_{i,j}(\vec{u}_i \bullet \vec{u}_j)$,

In case of i=1: $w_{1,2}(\vec{u}_1 \bullet \vec{u}_2) + w_{1,3}(\vec{u}_1 \bullet \vec{u}_3) \approx 0.5 \times 0.707 + 0.5 \times (-0.556) = 0.0755.$

In case of i=2: $w_{2,1}(\vec{u}_2 \bullet \vec{u}_1) + w_{2,3}(\vec{u}_2 \bullet \vec{u}_3) \approx 0.5 \times 0.707 + 1.0 \times (-0.980) = -0.6265.$

In case of i=3: $w_{3,1}(\vec{u}_3 \bullet \vec{u}_1) + w_{3,2}(\vec{u}_3 \bullet \vec{u}_2) \approx 0.5 \times (-0.556) + 1.0 \times (-0.980) = -1.2580.$

In case of i=3, the above equation produces the minimum value.

Therefore, b_3 will be the parent point.

(c)

Fig. 15. Parent point definition. (a) Definition for finding the parent point equation (3.3). (b) Assumption for the example of finding the parent point in (c). (c) Example for finding the parent point. By using the assumption in (b), we can identify the parent point. The equation for finding the parent point is based on (3.3). It produces the minimum value when i is 3. Therefore, we can think of b_3 as the parent point.

an iterative process for convergence. Therefore, Sun's algorithm is used only for one point (midpoint). The other medial axis points will be interpolated using the midpoint and the other control points (source, target, and tangent vector based point).

2. Tangent vector calculation

One derivative at c_i is calculated per MW to maintain G^1 continuity on the border of the CTTSs in (3.4). In many areas, C^1 is more important than G^1. However, C^1 could cause unnecessary waves where successive MWs have a large size difference because interpolation is dependent on the size of the MWs. To solve this problem, G^1 is used between consecutive CTTSs. This can easily support medial tracing by using the adaptive tangent parameter α, as shown in (3.5). Using a magnitude parameter η and three control points (c_i, c_{i+1}, and s_i), α is calculated as

$$\begin{aligned} \alpha_1 &= 2\eta||s_1 - c_1||/||c_1 - c_0|| \\ \alpha_i &= \eta||s_i - c_i||/||s_{i-1} - c_{i-1}||, \quad i > 1 \end{aligned} \tag{3.4}$$

where η is set between 0.0 and 1.0. If η is 0.0, consecutive CTTSs have only G^0 continuity. As η increases toward 1.0, the continuity, while still only G^0, begins to visually appear closer to G^1. From experimental results, η=0.4 was found to be a good value. Each interpolation polynomial $CTTS_i(t)$ is defined as follows with $c_i = CTTS_i(0.0)$, $s_i = CTTS_i(0.5)$, and $c_{i+1} = CTTS_i(1.0)$:

$$\begin{aligned} CTTS_0(t) &= c_0 L_1(t) + s_0 L_2(t) + c_1 L_3(t) \\ CTTS_i(t) &= \begin{pmatrix} 1 & t & t^2 & t^3 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & \frac{1}{2} & \frac{1}{4} & \frac{1}{8} \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 \end{pmatrix}^{-1} \begin{pmatrix} c_i \\ s_i \\ c_{i+1} \\ e_i \end{pmatrix} \end{aligned} \tag{3.5}$$

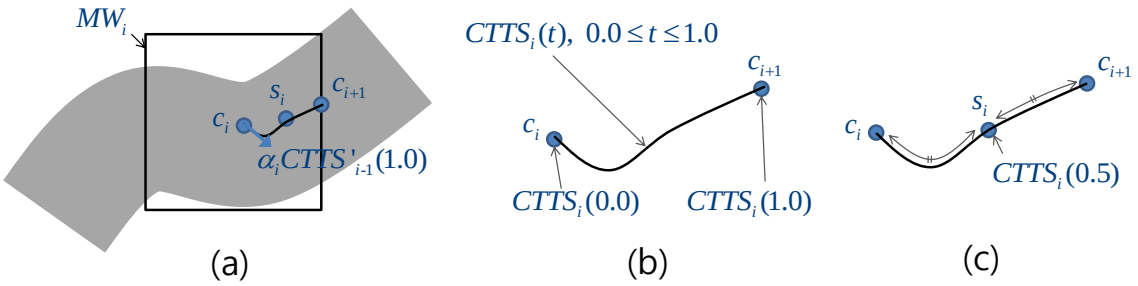


Fig. 16. Cubic tangential trace spline (CTTS) description. (a) Cubic tangential trace spline (CTTS) at step i. (b) $CTTS_i(t)$ starts from c_i and terminates at c_{i+1} on trace step i. So, c_i and c_{i+1} are $CTTS_i(0.0)$ and $CTTS_i(1.0)$ respectively. (c) The midpoint s_i is at $CTTS_i(0.5)$.

where $L_i(t)$ is a Lagrange basis function and e_i is the tangent control vector ($\alpha_i CTTS'_{i-1}(1.0)$).

D. Stopping criteria for tracing

The tracing is terminated if one or more of the following conditions are satisfied.

1. The newly generated MW is outside the image data.
2. The number of FCSs detected on the current MW is 1, which means that the fiber is traced to its end.
3. Local contrast is less than a predefined threshold. A small threshold may cause the detection of many false positives, while a large threshold may cause early termination of tracing.
4. A previously detected fiber intersects the current MW. All the pixels on FCSs detected on the current MW are checked for this test.

The first condition is used in other tracing methods [50, 51]. The second condition indicates that there are no more fibers to be traced from the current tracing step.

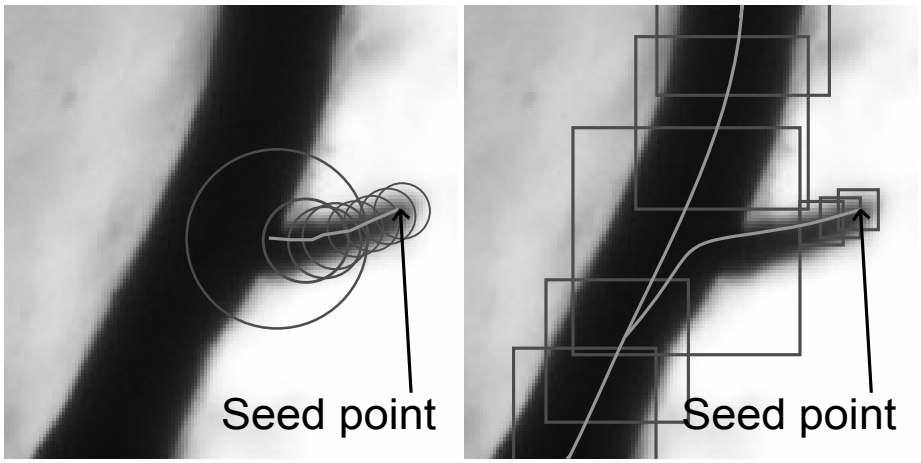
The third condition is used in order to differentiate fibers from background while tracing. If there is no clear fiber based on the local contrast threshold, the tracing process should terminate. The fourth condition is used to avoid backtracking like other tracing methods [50, 51].

E. Results

In this section, I will present the experimental results analyzed in four ways: (a) speed and complexity, (b) branch handling, and (c) smoothness.

1. Speed and computational complexity

Here, I analyze how efficiently my 2D method is able to process large amounts of data. Assuming that one fiber, whose length is k, is located on a MW whose side length is $2n\epsilon$, where n is the width of the fiber, I can calculate the required number of pixels to be processed by the sum of the size of the Gaussian filter, the MW side length, and the number of MW. The number of pixels to be processed by MW, denoted P_{MW}, is then calculated as follows: $P_{MW} = (2n\epsilon \times 4) \times (5 \times 5) \times \frac{k}{2n\epsilon} = 100 \times k$ where $2n\epsilon \times 4$ is the MW side length, 5×5 is the filter size, and $\frac{k}{n\epsilon}$ is the total number of MWs. Consequently, my 2D trace has $O(k)$ (i.e., linear) processing time, depending only on the fiber length (not all pixels in the fiber), which indicates that MW-CTTS is extremely efficient for large-scale data sets without additional overhead. This computational complexity is validated in section C of chapter V using Monte Carlo experiments.



(a) (b)

Fig. 17. Backtracking problem. (a) Haris et al.'s backtracking problem. (b) MW-CTTS' trace without backtracking after branch smoothing.

2. Branch handling while avoiding the backtracking problem

The purpose of the second experiment is to demonstrate the solution for the branch handling problem in MW-CTTS as compared to Haris et al.'s method [53] in branch points shaped like $\ni$ or $\dashv$. Instead of using square windows, Haris et al. proposed a tracing algorithm using circular templates. This algorithm generally works on bifurcation points shaped like $\prec$. However, it has a backtracking problem on bifurcation points shaped $\ni$ or $\dashv$ where the trace goes back where it came from as shown in Fig. 7(c). This problem happens because the center of the next step's tracing circle is not located on the current circle. On the other hand, MW-CTTS can avoid the problem by selecting the next MW's center on the edge of the current MW. All tracing methods based on Haris et al.'s method cause the same problem [55–58, 60]. Fig. 17(a) shows the backtracking problem on $\dashv$ shaped branch points in Haris et al.'s approach while MW-CTTS avoids this problem (Fig. 17(b)).

3. Experiments using neurovascular data

Fig. 18 shows the trace result with real neurovascular data starting from one seed point. These data were acquired from a 200-micron-thick section of mouse cerebellum stained with Nissl and the vasculature filled with India ink. MW-CTTS trace result highlights three important points. First, MW-CTTS starting from one seed point traces much more than Haris et al.'s method and Can et al.'s method. Second, medial axis tracing maintains C^1 or G^1 continuity on all traced points except for branch points. Third, branch point tracing is done using either linear interpolation or branch smoothness to remove misleading traces such as those in Fig. 14(b). The solid arrows in Fig. 18(f) show traces on well-defined branches and the hollow arrows show traces on ill-defined branches. Note that the line drawing looks more appropriate on the ill-defined branches, while the branch smoothness is good for the well-defined branches.

F. Conclusion

In this chapter, I presented the MW-CTTS algorithm to rapidly and correctly trace fibers in 2D images while maintaining C^1 or G^1 continuity. I have shown that (1) probing only on the edge of Moving Window can significantly reduce the trace processing time, (2) Moving Window can avoid backtracking problem on branches, and (3) CTTS can maintain smoothness on the medial axis. I also showed that, in neurovasculature data, the MW-CTTS algorithm can trace, from a single seed point, much more than Haris et al.'s method and Can et al.'s method.

Obtaining vascular networks from a tracing algorithm is important so that we can investigate how the vascular networks are organized using geometric/topological information and diagnose how these networks can affect various neurovascular diseases.

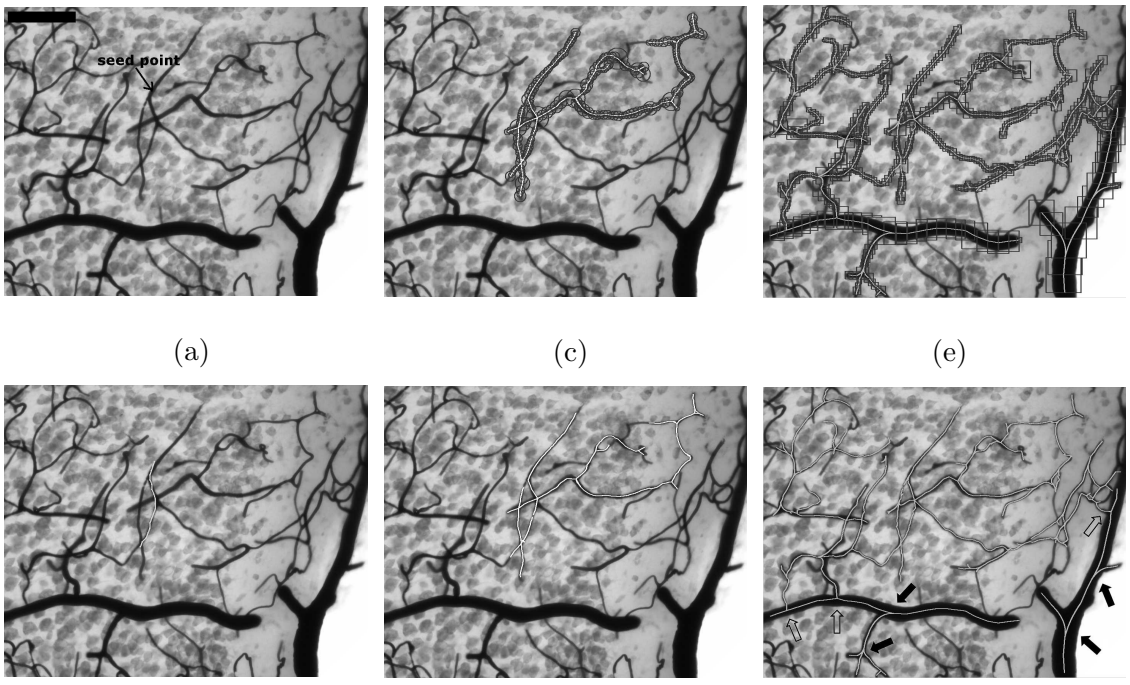


Fig. 18. Trace results from one seed point on mouse cerebellum (India ink stain). (a) Original data showing seed point. Bar=20μm. (b) Can et al.'s trace. (c) Haris et al.'s trace showing circles. (d) Haris et al.'s trace. (e) MW-CTTS trace showing MWs. Note that MW-CTTS starting from single seed point traces much more than Haris et al.'s method and Can et al.'s method. (f) MW-CTTS trace. Branch smoothness on the well-defined branches (solid arrows) and line drawing on the ill-defined branches (hollow arrows).

In the next chapter, I will introduce my 3D tracing algorithm, and show how 2D tracing algorithms can be extended to 3D tracing for obtaining 3D vascular networks.

CHAPTER IV

3D METHOD FOR TRACING

In this chapter, I explain my 3D tracing algorithm. First, I give notations used in this chapter. Second, I describe how the 3D tracing algorithm works. In detail, a fiber boundary detection algorithm is shown in order to obtain a MIP cube. After that, the use of local maximum intensity projection (MIP) for tracing is discussed. Due to the discretization error in MIP, I propose a momentum operator to adjust the center point. I also show how to connect branches after MIP tracing. Finally, in order to prove that the local MIP processing can also be used with other tracing methods, I use Frangi et al.'s 2D method for tracing [59]. The performance comparison shows how the processing time of MIP improves against 3D tracing methods.

A. Definition

Maximum intensity projection (MIP) is a computer visualization method that projects the maximum intensity value along each line perpendicular to the projected plane. In this section, Fig. 19 describes the axis definition and how to get local MIPs. A local MIP is generated within a MIP cube defined by axis length, which will be explained in the next section. In Fig. 19, X, Y, and Z axes are defined as usual and three local MIPs are projected on 2D planes along each axis. Such 2D MIPs can be used for tracing 3D objects. In Fig. 20, MIP cube size is important to get an unambiguous local MIP. If a MIP cube is too big (Fig. 20(b)), local MIPs have unclear projections on each plane. Therefore, the calculation of MIP cube size is important. The calculation of MIP cube size is based on the fiber boundary detection along each axis. The fiber boundary detection along each axis will be explained in the next section.

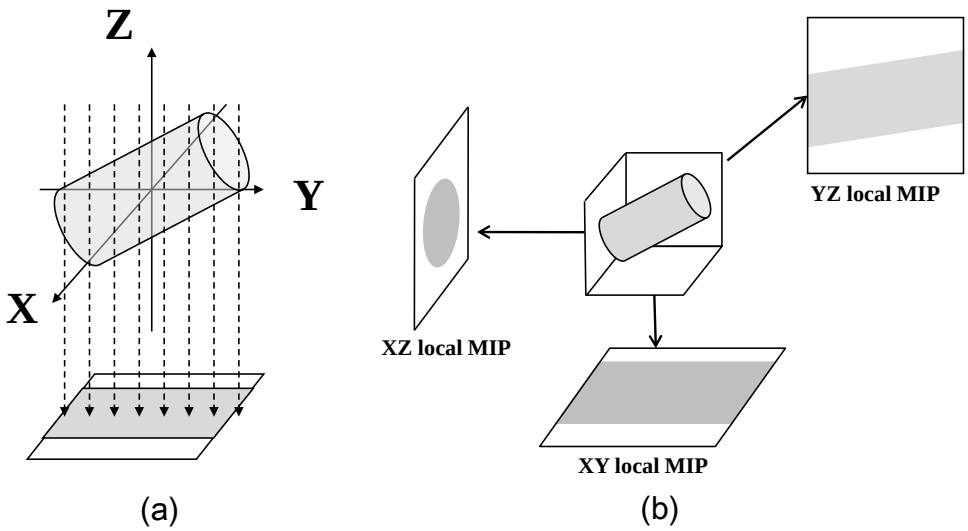


Fig. 19. MIP definition. (a) Local MIP definition. (b) XY, YZ, and XZ local MIPs on 2D planes along each axis.

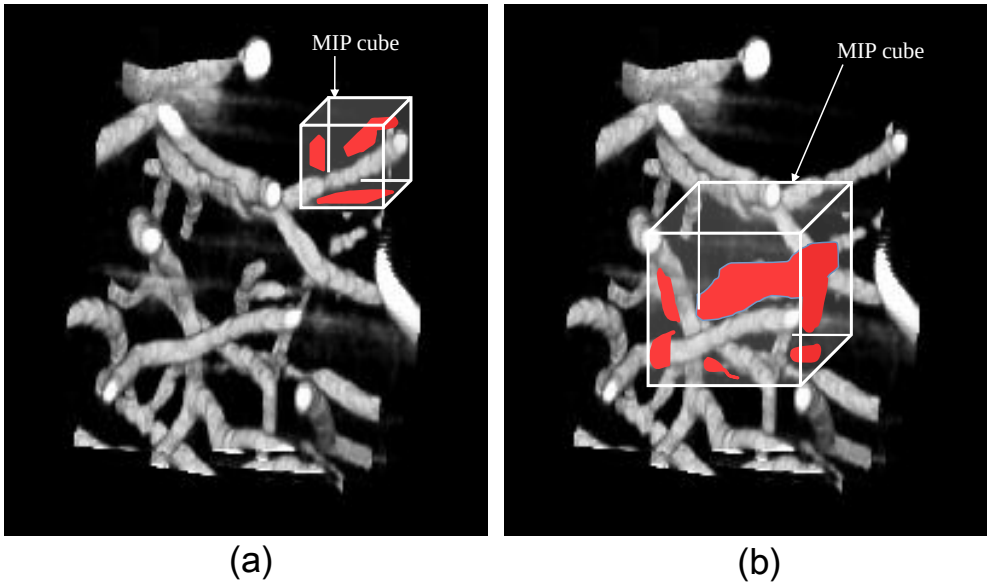


Fig. 20. The importance of MIP cube size. (a) Unambiguous local MIP containing a single fiber. (b) Ambiguous local MIP containing multiple fibers/branches.

B. Fiber boundary detection for obtaining MIP cube size

As discussed in the previous section, fiber boundary detection along each axis is important for correctly obtaining MIP cube size. Moreover, the detection should be correct on both high and low contrast images. Fiber boundary detection consists of three probabilities ($P_{\text{in}}(c,q), P_{\text{out}}(c,q), P_{\text{edge}}(q,\vec{\tau})$) to be able to work on both images. As shown in Fig. 21, c is the center voxel of the current tracing step, q is the boundary check voxel defined as $q \in \{l\vec{u}+c\}$ where l is an integer variable ($1 \leq l \leq m$) and $\vec{u}$ is a unit vector (0,1,0), m is the maximum radius of the fiber, and $\vec{\tau}$ is the vector for finding a boundary defined as $\vec{\tau} = l\vec{u}$. $\vec{u}$ can be changed for checking different directions. One example will be explained in the next paragraph.

Fig. 21 shows how to get a fiber's right boundary voxel from the center voxel c of the current tracing step using the three probabilities.

$$\begin{aligned} &P_{\text{in}}(c,q) = \exp\left(-\tfrac{(I_g(q)-I_g(c))^2}{2\sigma^2}\right) \\ &P_{\text{out}}(c,q) = 1 - \exp\left(-\tfrac{(I_g(q)-I_g(c))^2}{2\sigma^2}\right) \\ &P_{\text{edge}}(q,\vec{\tau}) \\ &\quad = (1 - \exp\left(-\tfrac{(I_g(q+\vec{\tau})-I_g(q-\vec{\tau}))^2}{2\sigma^2}\right))/\|\vec{\tau}\| \end{aligned} \tag{4.1}$$

where σ is constant which can be obtained by experimental result. $I_g(q)$ denotes the image intensity value at voxel q after convolving the data with a Gaussian kernel at one-third the scale of the maximum fiber radius. As shown in Fig. 21, q denotes one voxel from c to $m\vec{u}+c$ along the Y axis. $m\vec{u}+c$ is located m voxels from c along the Y axis. I used the Gaussian function form for detecting fiber boundary because this function can give the highest value when $I_g(q)$ and $I_g(c)$ have the same intensity. In addition to this, as the difference between the value of $I_g(q)$ and $I_g(c)$ increases, we can have gradually decreasing value from the Gaussian. I want to use this gradually

decreasing property for the probability of detecting fiber boundary. There could be other functional forms for the same effect. However, Gaussian function is widely used in image processing. Therefore, I selected Gaussian function for my implementation.

$P_{\text{in}}(c, q)$ denotes the probability that the q voxel is inside a fiber. $P_{\text{in}}(c, q)$ is the Gaussian function as defined in (4.1). If the voxels c and q have similar intensity values, the value of $P_{\text{in}}(c, q)$ will be high. $P_{\text{out}}(c, q)$ denotes the probability that the q voxel is outside a fiber. $P_{\text{out}}(c, q)$ is 1-$P_{\text{in}}(c, q)$. If the voxels c and q have different intensity values, the value of $P_{\text{out}}(c, q)$ will be high. $P_{\text{edge}}(q, \vec{\tau})$ is the probability that the $(q - \vec{\tau})$ voxel is inside a fiber and $(q + \vec{\tau})$ voxel is outside a fiber. If the voxels $(q - \vec{\tau})$ and $(q + \vec{\tau})$ have different intensity values, the value of $P_{\text{edge}}(q, \vec{\tau})$ will be high. In other words, when the voxel q is located on a fiber boundary, $P_{\text{edge}}(q, \vec{\tau})$ will be high.

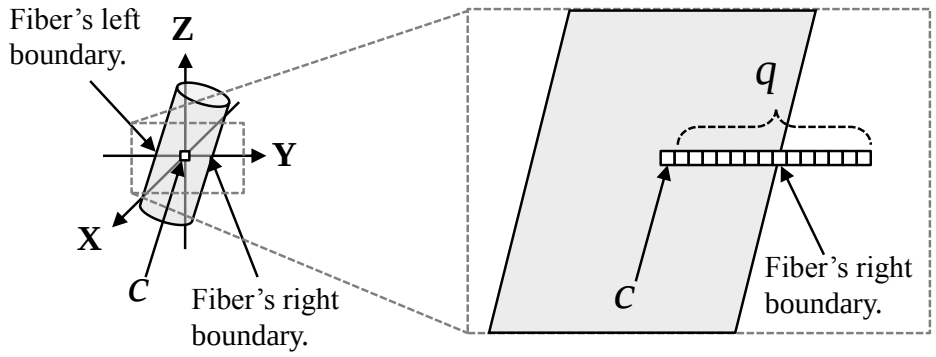


Fig. 21. Detection of fiber's right boundary. c denotes the center voxel of the current tracing step. q is the boundary check voxel defined as $q \in \{l\vec{u} + c\}$ where l is an integer variable $(1 \leq l \leq m)$, $\vec{u}$ is a unit vector (0,1,0), and m is the maximum radius of the fiber.

Using these probabilities, we can get the following edge map $E(c, q)$.

$$\begin{aligned} E(c, q, \vec{\tau}) &= \tfrac{(P_{\text{in}}(c, q-\vec{\tau}) + P_{\text{out}}(c, q+\vec{\tau}) + P_{\text{edge}}(q, \vec{\tau}))}{3} \\ E(c, q) &= \max E(c, q, \vec{\tau}), \text{ for } \vec{\tau} = l\vec{u},\ 1 \leq l \leq m. \end{aligned} \tag{4.2}$$

So, the fiber's right boundary voxel $Q(c)$ of voxel c is obtained by

$$Q(c) = \underset{q \in \{l\vec{u}+c\}}{\operatorname{argmax}} E(c, q) \tag{4.3}$$

If max $E(c, q)$ in (4.2) is close to 0, it means that all checked points are within fibers. In this case, $Q(c)$ is set to $m\vec{u} + c$. The other boundary voxels are obtained in a similar way. Fig. 22 shows how $E(c, q)$ obtains a boundary on low and high contrast images.

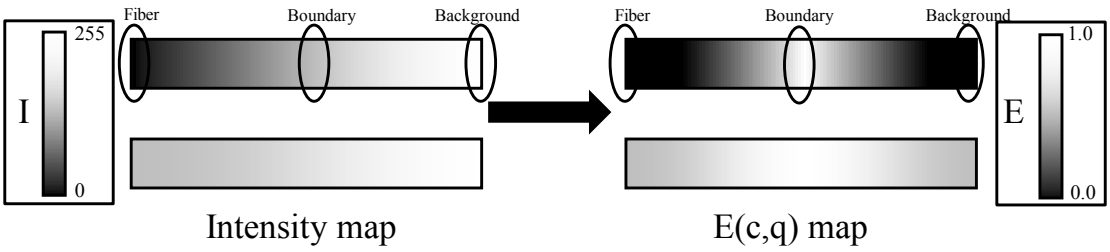


Fig. 22. Fiber boundary detection on high and low contrast data. First row: high contrast case. Second row: low contrast case. Left column: intensity map. Right column: E(c,q) map. In the E(c,q) map, the whitest vertical line is the detected boundary. When voxel q is on the boundary, E(c,q) is closest to 1.0, which is the whitest one in E(c,q) map.

From the detection of fiber boundary, I define the local Y axis length on a voxel c as the length from the fiber's left boundary voxel of c to the fiber's right boundary voxel of c. The local X axis length and local Z axis length are defined in a similar way as shown in Fig. 23(a). The MIP cube size is set to be (medium axis length+ζ)^3. If the longest axis length is set to be the MIP cube size, an ambiguous local MIP can be obtained. If the shortest axis length is set to be the MIP cube size, background will not be contained in local MIPs.

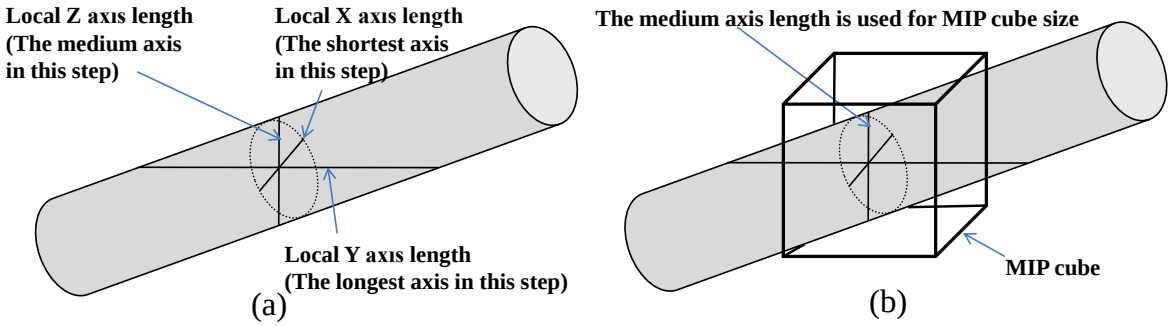


Fig. 23. MIP cube size calculation. (a) Local axis length definition. In this step, local Z axis length is medium. This length will be used for calculating this step's MIP cube. (b) MIP cube size calculation. MIP cube size is dependent on the medium axis length.

C. Local MIP tracing on projected 2D planes

In each tracing step, the local MIP along the longest axis will be ignored. The local MIP along the longest axis has less valuable information on the projected plane than the other local MIPs as shown in Fig. 24. Let us take an example that has local Y axis length as the longest one. In this case, XY and YZ local MIPs are considered for tracing. Note that any 2D tracing method can be used for getting fiber direction on these projected planes. On the XY local MIP, the direction $\vec{V}_{\mathrm{XY}}$ can be interpreted as (x_1,y_1,0). In the same way, on the YZ local MIP, the direction $\vec{V}_{\mathrm{YZ}}$ has (0,y_2,z_2). These two vectors are combined into $(x_1, (y_1 + y_2)/2, z_2)$ for $\vec{V}_{\mathrm{XYZ}}$, which is the 3D fiber direction on the current voxel. By using this, the next candidate center (denoted as c_i for the center on ith step and $c_{i-1}^{'}$ for the adjusted center on (i-1)th step) is calculated as follows:

$$c_i = \frac{\vec{V}_{\mathrm{XYZ}}}{\|\vec{V}_{\mathrm{XYZ}}\|} \times \omega + c_{i-1}^{'} \tag{4.4}$$

where ω is the tracing step size. The adjusted center $c_{i-1}^{'}$ will be explained in the next subsection.

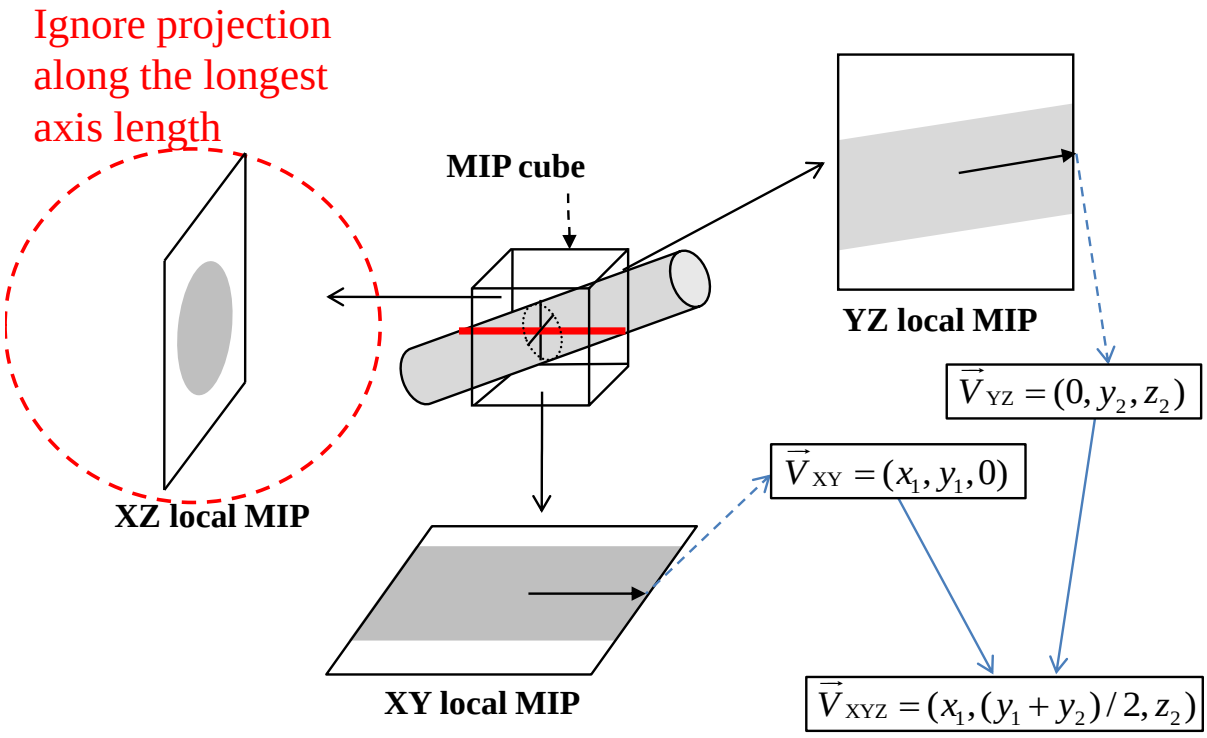


Fig. 24. Local MIP tracing. The local MIP along the longest axis length is ignored because the local MIP has no valuable information. In this step, XZ local MIP will be ignored. By using the other two local MIPs, 2D tracing directions will be obtained. The 2D tracing directions will be combined as 3D tracing directions after simple combination.

1. Center point adjustment using a momentum operator

Due to the discretization error in MIP, a momentum operator is used to adjust the center point within a spherical window as shown in Fig. 25 [83]. The maximum fiber radius is set to be the radius of the spherical window. From the following equation, we can get the adjusted center point $c_i^{'}$ at the ith tracing step.

$$c_i^{'} = (\frac{M_{100}}{M_{000}}, \frac{M_{010}}{M_{000}}, \frac{M_{001}}{M_{000}}) \tag{4.5}$$

where $M_{\chi\psi\kappa}$ is a momentum operator at the ith tracing step in the spherical window, where $\chi + \psi + \kappa = L$ is the order of the moments [83]. Therefore, $c_i^{'}$ can be the new center of mass (moment M_{000}) of the spherical window. The momentum operator $M_{\chi\psi\kappa}$ is given by

$$M_{\chi\psi\kappa} = \int_z \int_y \int_x x^{\chi} y^{\psi} z^{\kappa} f(x, y, z) dx dy dz, \tag{4.6}$$

where $f(x, y, z)$ is the intensity value at location (x, y, z) within the spherical window.

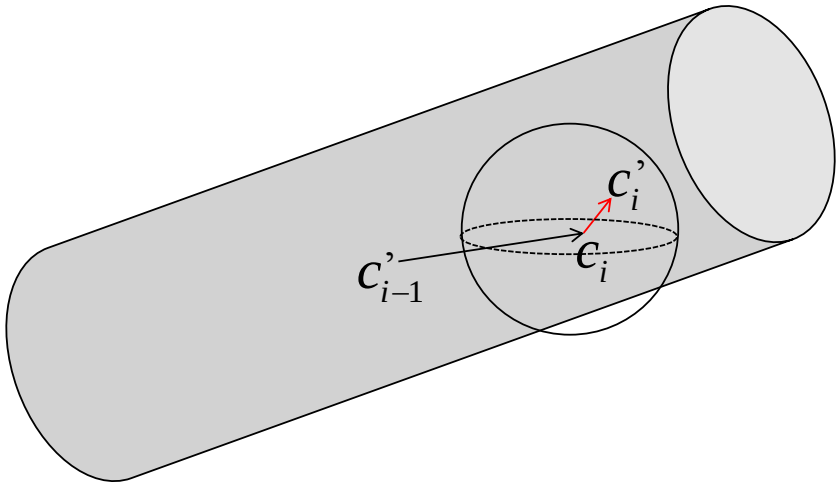


Fig. 25. 3D vessel direction adjustment using a momentum operator.

2. Branch connection

In practice, the local MIP algorithm traces along the maximum response of the template, which does not handle cases with branches. All branches should be traced with each seed point. The tracing stops if the previously traced voxels are detected within a sphere of a predefined radius and can be connected to the current step's center voxel. It can be said that two voxels are connected if all voxels on the Bresenham's line between the two voxels have higher intensity than background intensity assuming that the background is dark [81].

3. Stopping criteria

After MIP processing, any 2D tracing algorithm can be used on the projected 2D planes. It means that each algorithm's stopping criteria can be used as the stopping criteria of the proposed MIP algorithm. The algorithm proposed by Haris et al. stops when there are no detected fibers and Can et al.'s tracing stops when maximum response of template is lower than a predefined threshold [51, 53]. The tracing algorithm presented by Friman et al. is terminated if a score function measuring the goodness of fit to the fiber does not satisfy a predetermined threshold [30]. I will use the stopping criteria of my MW-CTTS 2D tracing algorithm for experiments as described in chapter III.

D. Frangi tracing

In this section, I will review Frangi et al.'s method to show that the local MIP processing can work with the well-known tracing algorithm [59, 64]. Frangi et al.'s tracing algorithm is a good source for validating my MIP processing because Frangi et al.'s tracing algorithm has both a 2D and a 3D version. I will compare the results

from Frangi's 3D version to that of the 2D version using my MIP processing to see if there is any difference in performance with respect to accuracy and speed.

A multiscale filter using the Hessian matrix is used for fiber direction detection in a 2D image. Let $H^2_\sigma(c)$ denote the Hessian matrix at a pixel c with scale σ:

$$H^2_\sigma(c) = \begin{bmatrix} I_{xx}(c) & I_{xy}(c) \\ I_{yx}(c) & I_{yy}(c) \end{bmatrix} \tag{4.7}$$

$$I_{mn}(c) = \sigma^2 \frac{\partial^2 (G(c,\sigma) I(c))}{\partial m \partial n} \tag{4.8}$$

where $G(c,\sigma)$ is the Gaussian kernel with center c and standard deviation σ. The eigenvector v_1 corresponding to the smallest magnitude eigenvalue λ_1 indicates the fiber direction. The fiber likeliness function $T^2(c,\sigma)$ in the 2D version uses only one geometric ratio R_B [64]:

$$T^2(c,\sigma) = \begin{cases} 0 & \text{if } \lambda_2 > 0 \\ \exp\left(-\frac{R_B^2}{2\beta^2}\right)\left(1 - \exp\left(-\frac{S^2}{2\gamma^2}\right)\right) & \text{otherwise,} \end{cases} \tag{4.9}$$

with

$$R_B = \frac{|\lambda_1|}{|\lambda_2|}, S = \sqrt{\sum_{i=1}^{2} \lambda_i^2}, \tag{4.10}$$

where R_B is the blobness measure and S is the second order structure measurement. The parameters β and γ were set to 0.5, and 0.25 respectively. The maximum value among $T^2(c,\sigma)$ is given as follows:

$$T^2(c) = \max_{\sigma_{\min} \le \sigma \le \sigma_{\max}} T^2(c,\sigma). \tag{4.11}$$

Fig. 26 shows that the eigenvector v_1 approximates the local fiber direction in 2D.

In a similar way, the 3D version of Frangi et al.'s tracing algorithm is as follows.

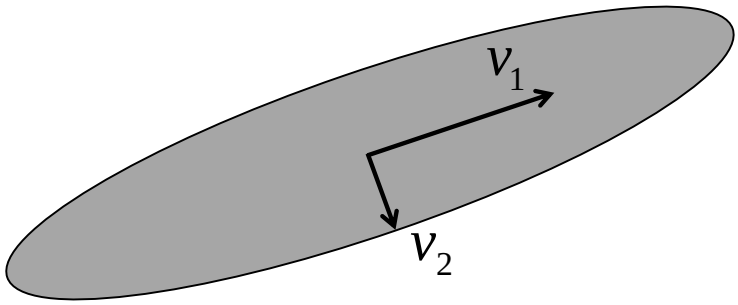


Fig. 26. The direction of the fiber in 2D using the Hessian matrix. The eigenvector v_1 of the Hessian matrix corresponding to the smallest eigenvalue λ_1 is the direction of the fiber in 2D.

Let $H^3_\sigma(c)$ denote the Hessian matrix at a voxel c with scale σ:

$$H^3_\sigma(c) = \begin{bmatrix} I_{xx}(c) & I_{xy}(c) & I_{xz}(c) \\ I_{yx}(c) & I_{yy}(c) & I_{yz}(c) \\ I_{zx}(c) & I_{zy}(c) & I_{zz}(c) \end{bmatrix} \tag{4.12}$$

$I_{mn}(c)$ is defined as in 4.8. The eigenvector v_1 corresponding to the smallest magnitude eigenvalue λ_1 indicates the fiber direction. Frangi et al. presented fiber likeliness function $T^3(c,\sigma)$ using geometric ratios R_A and R_B [59]:

$$T^3(c,\sigma) = \begin{cases} 0 & \text{if } \lambda_2 > 0 \text{ or } \lambda_3 > 0 \\ \left(1 - \exp\left(-\frac{R_A^2}{2\iota^2}\right)\right) \exp\left(-\frac{R_B^2}{2\beta^2}\right) \left(1 - \exp\left(-\frac{S^2}{2\gamma^2}\right)\right) & \text{otherwise,} \end{cases} \tag{4.13}$$

with

$$R_A = \frac{|\lambda_2|}{|\lambda_3|}, R_B = \frac{|\lambda_1|}{|\lambda_2\lambda_3|}, S = \sqrt{\sum_{i=1}^{3} \lambda_i^2}, \tag{4.14}$$

where R_A is used to distinguish between a plate-like or a tube-like structure. R_B and

S have the same meaning as in the 2D version. The parameters ι, β, and γ control the sensitivity of the Hessian filter and they were set to 0.5, 0.5, and 0.25 respectively. The maximum value among $T^3(c,\sigma)$ at scale σ gives the eigenvector v_1 which aligns with the fiber direction as follows:

$$T^3(c) = \max_{\sigma_{\min} \leq \sigma \leq \sigma_{\max}} T^3(c,\sigma). \tag{4.15}$$

Fig. 27 shows that the eigenvector v_1 approximates the local fiber direction in 3D.

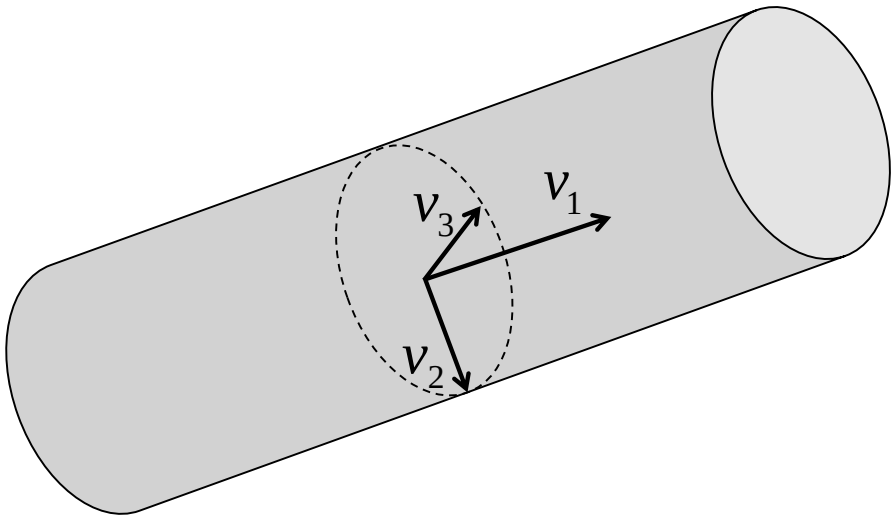


Fig. 27. The direction of the fiber in 3D using the Hessian matrix. The eigenvector v_1 of the Hessian matrix corresponding to the smallest eigenvalue λ_1 is the direction of the fiber in 3D.

E. Results

In order to test my local MIP algorithm, I used the 2D-version of Frangi et al.'s method [59]. I also used the local MIP algorithm with my MW trace algorithm for generating 3D results and validation. Tracing in low contrast data will be shown to test the boundary detection approach.

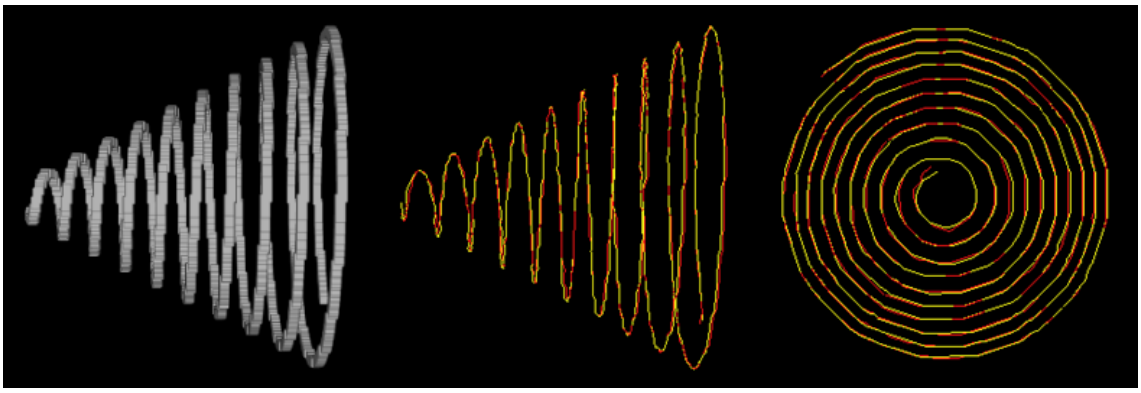


Fig. 28. Local MIP traces with well-known method. Local MIP traces. Left: Synthetic data. Middle: Trace result. The trace result of 2D version of Frangi et al.'s method with my local MIP processing is overlayed on top of the trace result of the 3D version of Frangi et al.'s method. Right: The same trace result seen from a different view.

1. 3D Frangi vs. 2D Frangi with MIP cube

In this subsection, the trace result from 3D Frangi et al.'s method is compared to the trace result from the MIP algorithm with 2D Frangi et al.'s method. First, simple synthetic data are considered to see how similar the results are. Second, performance comparison is done with real mouse cerebellum data. Finally, validation experiments were conducted using human-labeled ground truth.

a. Synthetic data tracing result

The synthetic data size was $256 \times 256 \times 256$ voxels and the data had curvature ranging from 0.001 to 0.1 with a width of 5 voxels. Fig. 28 shows the result of my method (Frangi et al.'s 2D tracing method with local MIP processing) and the result of Frangi et al.'s 3D tracing method. Note that the 2D tracing result with local MIP is almost identical to the 3D tracing result.

Fig. 29. Trace comparison. Left: Mouse cerebellum data obtained by the KESM. The volume size was $128 \times 128 \times 128$ voxels. Voxel size was $0.6\mu m \times 0.7\mu m \times 1.0\mu m$. Middle: The trace result of the 3D version of Frangi et al.'s method. Right: The trace result of the 2D version of Frangi et al.'s method with my local MIP processing.

b. Performance comparison

The size of the mouse cerebellum vascular volume acquired by KESM was $128 \times 128 \times 128$ voxels. Fig. 29 shows one volume with the trace results. In this case, my method took 188 seconds to trace 1188 voxels while the 3D tracing method took 484 seconds for 1065 voxels. I performed the experiment on a PC with Intel Pentium 4 (2.4 GHz) processor, 512MB of memory under Windows XP operating system in Debug mode of Microsoft Visual Studio 2005. Note that Fig. 30 shows how significantly my method reduces processing time compared to a 3D method. This improvement is due to the fact that the 3D method uses a 3×3 Hessian matrix with 3D Gaussian kernel ($O(n^3)$) while my method uses a 2×2 Hessian matrix with 2D Gaussian kernel ($O(n^2)$), where n is the scale of Gaussian filter. If n is a small value such as 2 or 3, there is not much of a performance improvement. Other tracing algorithms can have similar results due to the dimension reduction. For example, Friman et al.'s method will have a similar performance with local MIP processing because the 3D version has $O(n^2)$ processing

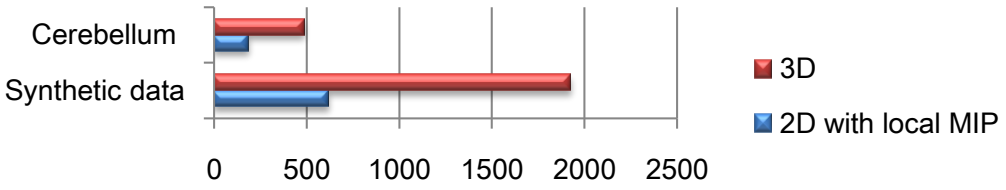


Fig. 30. Performance comparison for the cerebellum and the synthetic data regarding MIP tracing. Horizontal axis shows seconds. 2D with local MIP tracing is about three times faster than 3D tracing in both the cerebellum and the synthetic data. The cerebellum and the synthetic data have 1188 voxels and 3911 voxels, respectively. My method takes 34.78% and 35.40% of 3D processing time for the cerebellum and the synthetic data, respectively. This experiment shows that the performance improvement is similar in both the cerebellum (34.78%) and the synthetic data (35.40%).

time while the 2D version is $O(n)$, where n is the number of sampled directions in 2D space [30].

c. Validation

The following experiments were conducted for validating the Frangi et al.'s 2D method with local MIP processing. Two individuals manually selected the centerline points for 10 pieces of vascular data. Two measurements (length difference:ϕ and centerline deviation:φ) were used to quantitatively evaluate the difference between my method's result (A) and two manual results (R1, R2) based on Zhang et al.'s validation [84]. The length difference (ϕ) is derived as

$$\phi = |1 - L_R/L_A| \tag{4.16}$$

where L_R and L_A are the length of vascular data extracted by person and my method respectively. The centerline deviation (φ) is defined as

$$\varphi = V_{ox}(l_R, l_A)/L_A \tag{4.17}$$

where $V_{ox}(l_R, l_A)$ is the total number of voxels between l_R and l_A. l_R and l_A are the manual vessel centerline and my method's centerline respectively. Table I shows how different my method's result is from that of the manual results. The p-values are obtained using a two-sided paired t-test, and they show that the two manual results are similar (no significant difference). Table II shows that there is strong correlation between my method's result and the manual results.

Table I. The mean (μ) and standard deviation (σ) of length difference (ϕ) and centerline deviation (φ) values for two manual results (R1 & R2).

	ϕ			φ		
	μ	σ	p-value	μ	σ	p-value
R1	0.1518	0.1762	0.4188	1.2131	0.3529	0.6853
R2	0.1325	0.1804		1.1294	0.3016	

Table II. Correlation coefficients between my method (A) and the manual ground truths (R1 & R2). Note that the tables are symmetric, so repeated values are not shown.

	ϕ			φ		
	A	R1	R2	A	R1	R2
A	-	0.9917	0.9904	-	0.9552	0.9628
R1	-	-	0.9982	-	-	0.9740
R2	-	-	-	-	-	-

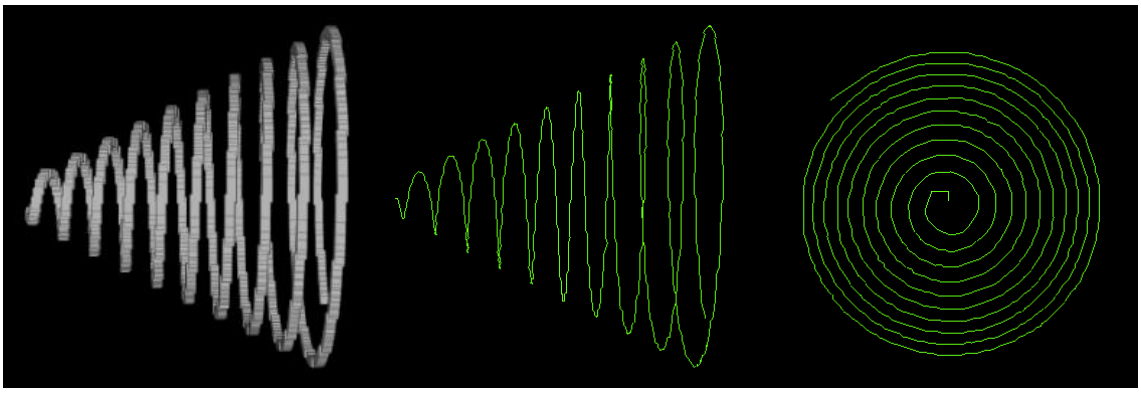


Fig. 31. Local MIP traces. Left: Synthetic data. Middle: Trace result. The trace result is extracted from my MW method using my local MIP processing. Right: The same trace result seen from a different view.

2. MW with MIP cube

In this subsection, the trace result from my MW method with MIP processing is shown. First, simple synthetic data are considered, similar to the previous subsection. Second, trace results from real mouse cerebellum data are shown. Finally, validation experiments are conducted using human-labeled ground truth.

a. Synthetic data tracing result

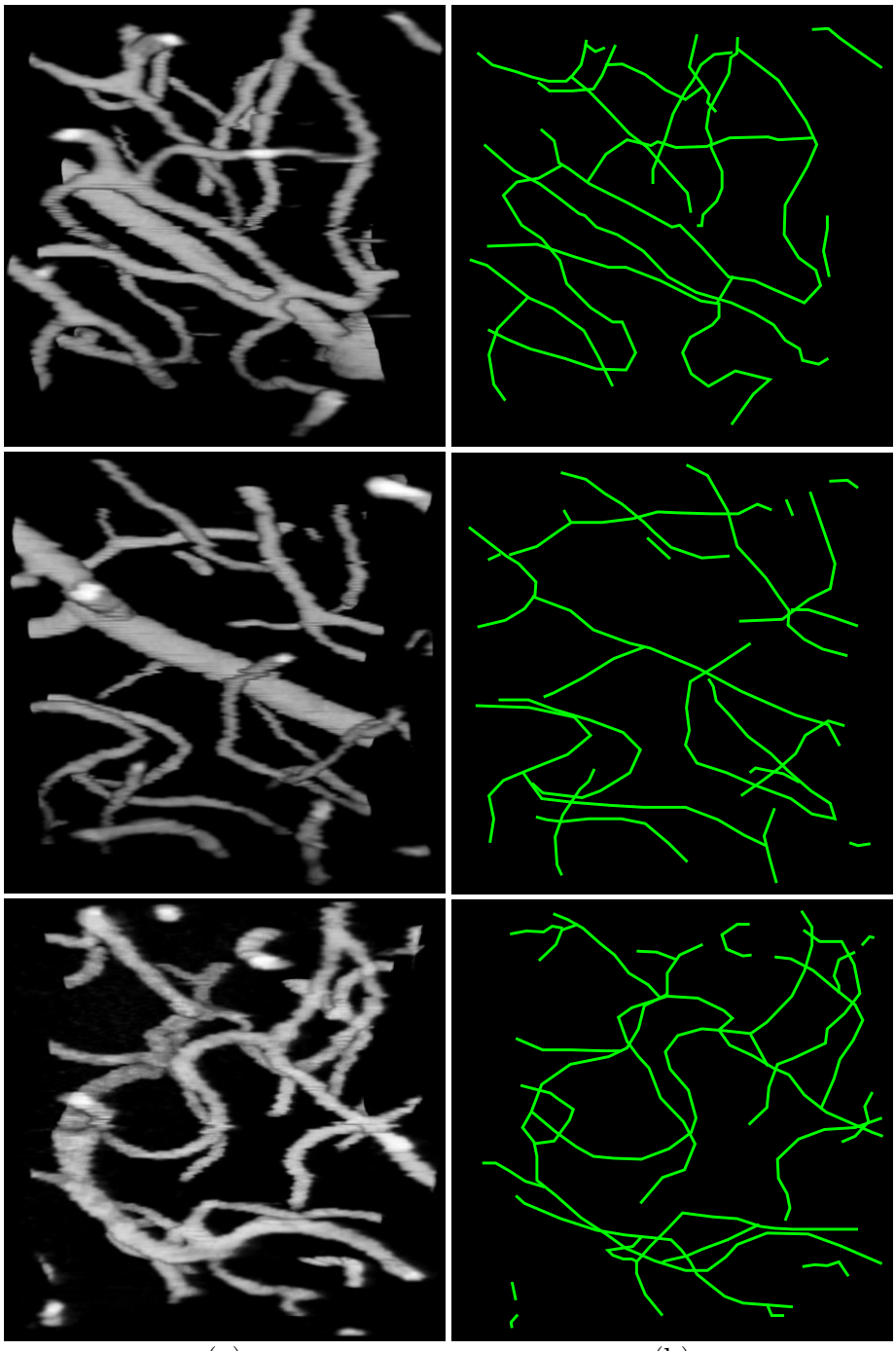
The synthetic data size was $256 \times 256 \times 256$ voxels and the data had curvature ranging from 0.001 to 0.1, with a width of 5 voxels. Fig. 31 shows the result of the my MW method. The result is almost identical to the Frangi et al.'s tracing result as shown in Fig. 28. It took 569 seconds to trace this data. Note that the Frangi et al.'s 2D tracing method with local MIP processing took 619 seconds to trace the same synthetic data, while the 3D version took 1926 seconds.

b. Performance comparison

The size of the mouse cerebellum vascular volume acquired by KESM was 128 × 128 × 128 voxels. Fig. 32 shows three volumes with trace results. In this case, my method took 193, 177, and 194 seconds for 1326, 1214, and 1334 voxels respectively. I performed the experiment on a PC with Intel Pentium 4 (2.4 GHz) processor, 512MB of memory under Windows XP operating system in Debug mode of Microsoft Visual Studio 2005. All experiments in my dissertation were conducted using this environment if there is no other specification. Even though Release mode gives a faster processing time than Debug mode, I used Debug mode to find all bugs in my implementation during all the experiments. After compiling it in Release mode, I checked that my implementation worked well in Release mode. For your information, in Release mode, there was about 80% reduction in execution time.

c. Validation

In this validation part, the same experiments were conducted as in the Frangi et al.'s validation part. Two individuals manually selected the centerline points for 10 pieces of vascular data. We used two measurements, length difference (ϕ) and centerline deviation (φ), to quantitatively evaluate the difference between my method's result (A) and two manual results (R1, R2) based on Zhang et al.'s validation [84]. Length difference (ϕ) is defined in Eq. 4.16. Centerline deviation (φ) is defined in Eq. 4.17. Table III shows how different my method's result is from the manual results. The p-values are obtained using a two-sided paired t-test, which shows that the two manual results are similar (no significant difference). Table IV shows that there is strong correlation between my method's result and the manual results.



(a) (b)

Fig. 32. Tracing result of cerebellum data using MIP processing. (a) Cerebellum data. (b) MW trace with MIP processing.

Table III. The mean (μ) and standard deviation (σ) of length difference (ϕ) and centerline deviation (φ) values for two manual results (R1 & R2).

	ϕ			φ		
	μ	σ	p-value	μ	σ	p-value
R1	0.4238	0.6296	0.5028	1.5392	0.3529	0.6913
R2	0.5901	0.8142		1.7170	0.5406	

Table IV. Correlation coefficients between my method (A) and the manual ground truths (R1 & R2). Note that the tables are symmetric, so repeated values are not shown.

	ϕ			φ		
	A	R1	R2	A	R1	R2
A	-	0.9832	0.9814	-	0.9613	0.9582
R1	-	-	0.9921	-	-	0.9934
R2	-	-	-	-	-	-

3. Tracing in low contrast data

Due to uneven illumination, KESM sometimes yields low contrast images. The boundary detection part should evaluate correct local volumes so that my method can trace a 3D volume correctly. Fig. 33 shows the tracing result on a low contrast volume using the boundary detection algorithm in 4.1.

F. Conclusion

In this chapter, I explained how to use maximum intensity projection (MIP) in 3D tracing for the reusability of existing 2D tracing algorithms and the subsequent reduction of computational complexity. I have also shown (1) the fiber boundary detection

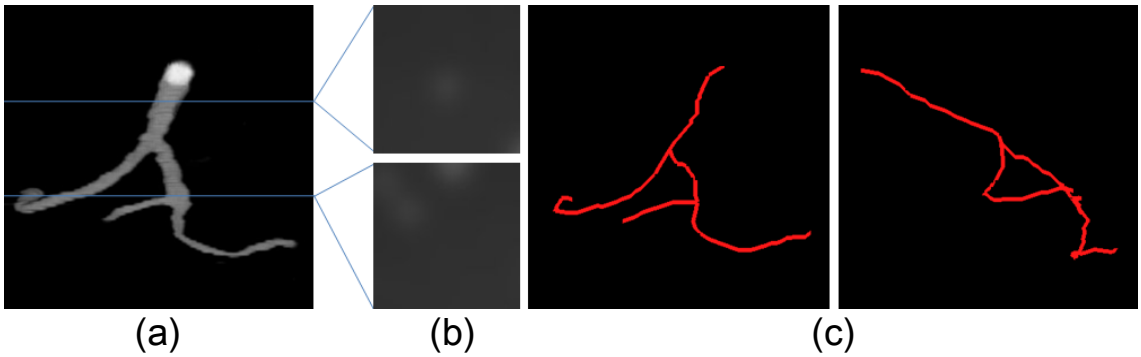


Fig. 33. Low contrast data trace. (a) 3D vascular data. (b) Two sample slices from (a). Gaussian blur (σ=30) was applied in all slices to simulate low contrast. The first sample slice had a mean intensity of 47.9 (μ) with standard deviation 4.5 (σ). The second sample slice had mean 49.1 (μ) with standard deviation 6.9 (σ). (c) Trace result with my method from two views.

algorithm for low contrast data, (2) center adjustment using a momentum operator to correct the discretization error of MIP processing, and (3) the use of Frangi et al.'s tracing method for validating my MIP processing. I also conducted a validation experiment with human-labeled ground truth to see how accurate the tracing results are.

In the conclusion section of the previous chapter, I explained the importance of obtaining 3D reconstructions of vascular networks. Using my MIP processing, the existing 2D tracing methods can be used for reconstructing 3D vascular networks. Due to the low computational complexity of 2D tracing algorithms compared to 3D tracing algorithms, the existing 2D tracing algorithms with my MIP processing took less time than that of 3D tracing algorithms (e.g.: about 65 % reduction of processing time compared to Frangi et al.'s 3D algorithm). Even though my tracing algorithm gives good results, we need to validate the algorithm to see how robust it is to noise and contrast. In the next chapter, I validated my tracing algorithm with a Monte Carlo method using various noise levels and contrasts.

CHAPTER V

VALIDATION FOR THE 2D AND 3D TRACING METHODS

In this chapter, I validated my tracing algorithm using synthetic data. First, I describe the performance measures that I use. Second, I introduce the method of generating synthetic data. Next, I explain the 2D validation method that uses the synthetic data. The synthetic data covers noise levels representative of medical images such as MR, CT, and ultrasound. I also validate my 3D tracing algorithm with 3D synthetic data. The 3D synthetic data also has the same noise level as the 2D synthetic data.

A. Measurement of performance and validation

In order to quantify the speed and accuracy of my method, Monte Carlo experiments were conducted using synthetic data. The total number of synthetic data sets was 84 (three data types×seven intensity profiles×four noise levels) for 2D performance and validation and for 3D performance and validation. The data types were line, curve, and spiral in 2D images. The data types were branch, stacked curve, and spiral in 3D volumes. The data types will be described in detail in section C and D of this chapter. For each data set, 100 random seed points were used within 2 pixels along the medial axis, thereby avoiding the placement of seed points in the background.

Four measures were used in each data set during the Monte Carlo runs. I calculated the averages and standard deviations for each measure. The first two of these measures quantify how fast my method can trace. The next two of them quantify the noise robustness and accuracy. The measures are as follows:

1. *Average Processing Time*: Mean processing time of the tracing for each data.

2. *Average Time/Distance*: Mean trace processing time per traced pixel.

3. *Average Error*: Mean distance between a point on the traced result and its closest medial axis point.

4. *Percent of Points within 2 Pixels*: Percent of points on the traced result within 2 pixels of their closest medial axis point.

Note that the first two measures are for performance and the last two measures are for validation. These measurements are based on the Aylward et al.'s measurement [61].

B. Synthetic data degradation

In this section, I described how to add noise to the synthetic data. For example, in 2D image, a degraded image $g(x, y)$ can be derived as

$$g(x, y) = f(x, y) + h(x, y) \tag{5.1}$$

where $f(x, y)$ is the original image and $h(x, y)$ is an additive noise term at pixel (x, y). There are many noise models such as Gaussian, Rayleigh, Erlang (Gamma), uniform, and exponential noise. For example, the Gaussian noise arises in an image due to factors such as electronic circuit noise and sensor noise due to poor illumination and/or high temperature [42]. Therefore, in this experiment, I used the Gaussian noise model. Fig. 34 shows how to add the Gaussian noise to synthetic data.

For the Gaussian noise, the PDF of a Gaussian random variable, z, is given by

$$p(z) = \frac{1}{\sqrt{2\pi}\sigma} \exp^{-(z-\mu)^2/(2\sigma^2)} \tag{5.2}$$

where z represents gray level, μ is the mean of z, and σ its standard deviation. Approximately, 70% of the z values will be in the range $[(\mu - \sigma), (\mu + \sigma)]$, and about 90% of its values will be in the range $[(\mu - 2\sigma), (\mu + 2\sigma)]$.

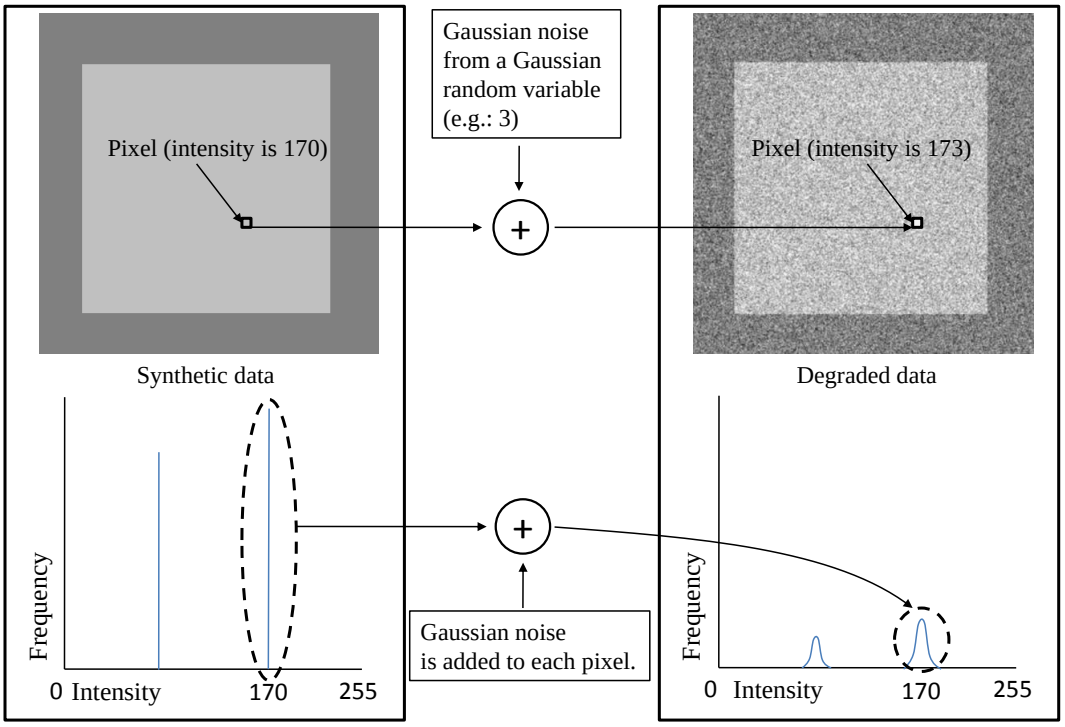


Fig. 34. Illustration of degrading image using the Gaussian noise model. Left column: Synthetic data with histogram. Right column: Degraded data with histogram after adding Gaussian noise.

C. 2D validation using Monte Carlo experiments

For the synthetic images, I generated three 256×256 pixel data sets. Each data set contained linear, curvy, and spiral fiber, respectively. Generally, lines and curves most frequently appear in blood vessel data. The spiral data contained curvatures ranging from 0.17 to 0.0044 to cover most vessels. The ideal traversals of these fibers' centerlines result in the extraction of 200, 230, and 2900 centerline points for line, curve, and spiral data in the ground truth, respectively. The fiber's radius ranged from 2.0 to 4.0 pixels and the background intensity was 0. All images had one of the following combination of intensity profiles and Gaussian noise as shown in Fig. 35.

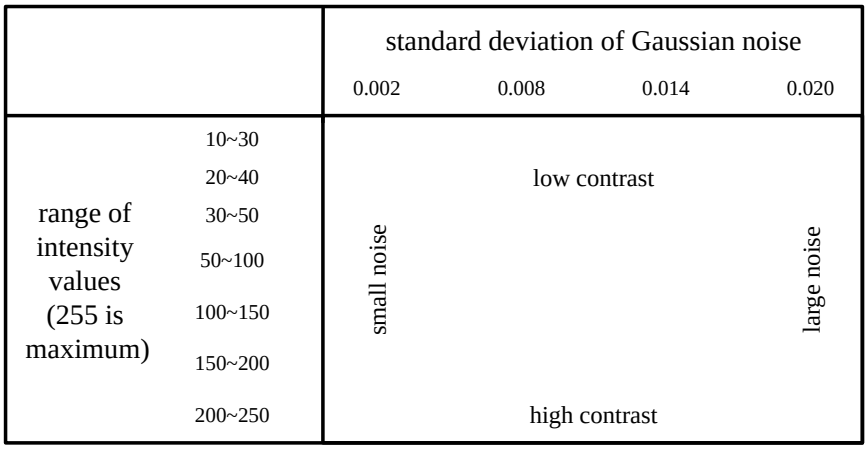


Fig. 35. Combination of intensity profiles and Gaussian noise for synthetic data.

The Gaussian noise had standard deviation σ of 0.002, 0.008, 0.014, and 0.020. This is very small noise on a 0∼255 range. However, my intention in this experiment was to test synthetic images from the lowest contrast to the highest contrast with noise levels representative of MR, CT, and ultrasound. The data with $\sigma = 0.006$ is representative of the noise level in MR and CT data and the one with $\sigma = 0.02$ is similar to the noise in ultrasound data. I tested the lowest contrast image with intensity profiles from 10 to 50 to investigate my method's performance in a worst

case scenario as shown in Fig. 36. The lowest contrast has three finer ranges (10∼30, 20∼40, and 30∼50). From now on, I will call these three ranges the lowest contrast.

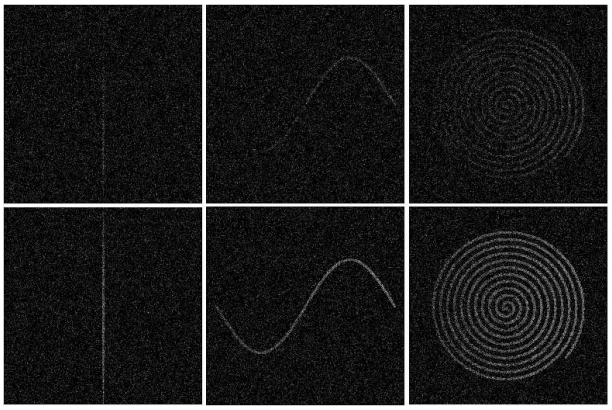


Fig. 36. Synthetic data with Gaussian noise. It has standard deviation σ of 0.020. The noise-free fibers have intensity profiles from 10 to 50 on the first row and from 50 to 100 on the second row. The first row contains the low contrast which is less than 30 percent of the gray level contrast that human eye can distinguish.

Fig. 37 shows the tracing result of 2D synthetic data. The intensity profile is from 50 to 100 and the Gaussian noise has a standard deviation of 0.020. Based on the measurements defined in section A of this chapter, there are two main parts: performance and validation.

1. Performance of my algorithm

In Fig. 38, each series represents the measurement of each image data having different intensity profiles, respectively. From these measurements, I can confirm that the MW-CTTS time complexity is linear with fiber length. The number of the medial axis points of line, curve, and spiral data were 200, 230, and 2900, respectively. In other words, the spiral fiber length was ten times longer than the other's length. The spiral data processing time (about 0.5 sec) was almost ten times larger than the line and

curve (about 0.055 and 0.04 sec, respectively) in Fig. 38, which ensures that the time complexity is linear independent of data type, noise, and intensity levels. Only low contrast data whose intensity values range from 10 to 50 had very low processing time. This is due to the early termination of tracing caused by the combination of noise and low contrast. However, the average processing time/distance (about 0.15 μsec in Fig. 39) was almost constant over all image data.

2. Validation of my algorithm

The second experiment was conducted to show noise robustness of the MW-CTTS algorithm. There are two measurements to be considered for noise robustness as defined in section A of this chapter. First, most data produced good results independent of noise level, as shown in Fig. 40. Except the lowest contrast, all the other data showed that average error was less than 1 pixel. Second, all experiments (except the lowest contrast data) showed that more than 97% of the traced points were within two pixels of the medial axis as shown in Fig. 41. However, except the lowest one, the other data contain intensity profiles from 50 to 250. Therefore, my MW-CTTS method can be applied to a wide variety of medical images such as CT, MR, and ultrasound.

D. 3D validation using Monte Carlo experiments

I generated three 256×256×256 voxel data sets for 3D validation. Each data set contained branch, stacked curve, and spiral, respectively. Generally, branch and stacked curve cases most frequently appear in blood vessel data. The ideal traversals of these fibers' centerlines result in the extraction of 180, 5220, and 2890 centerline points for branch, stacked curve, and spiral data in the ground truth, respectively. The fiber's radius ranged from 2.0 to 4.0 pixels and the background intensity was

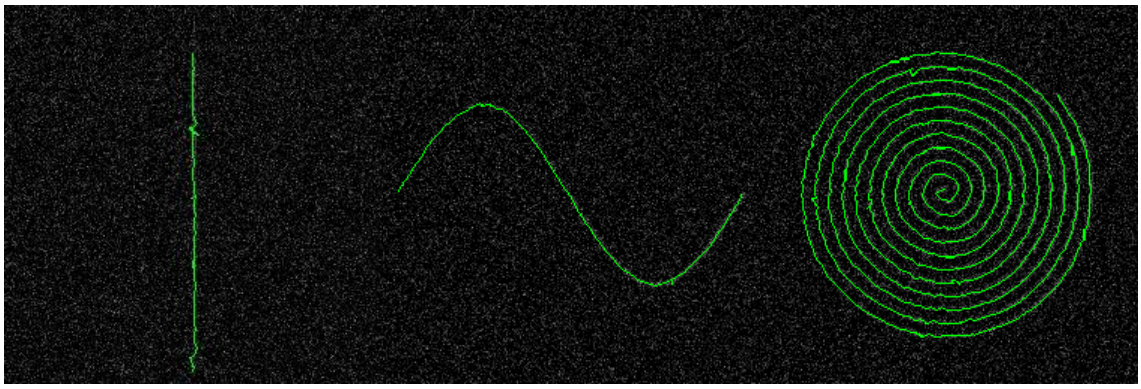


Fig. 37. Tracing result of 2D synthetic data. Left: Line data. Middle: Curve data. Right: Spiral data. The standard deviation of the Gaussian noise is 0.020. Intensity profile is from 50 to 100.

0. The combination of intensity profiles and Gaussian noise was the same as the 2D data combination for covering the noise level representative of medical images (MR, CT, and ultrasound).

The left column of Fig. 42 shows the branch, stacked curve, and spiral data. The simplest data (branch) was tested first. Each fiber in the stacked curve data was a sine curve and was rotated about the Z and the Y axis. The spiral data contained curvatures ranging from 0.001 to 0.1. The right column of Fig. 42 shows the tracing result from the 3D synthetic data. The intensity profile was from 50 to 100 and the Gaussian noise had a standard deviation of 0.020.

In Fig. 43, each series represents the measurement of each volume data having different intensity profiles. From these measurements, as in 2D validation, I can confirm that the MW with MIP processing has linear time complexity relative to fiber length. The number of the traced medial axis points for the branch, stacked curve, and spiral data were 180, 5220, and 2890, respectively. For example, the spiral fiber length was sixteen times longer than the branch fiber length. The spiral data processing time (about 337 sec) was almost sixteen times greater than the branch

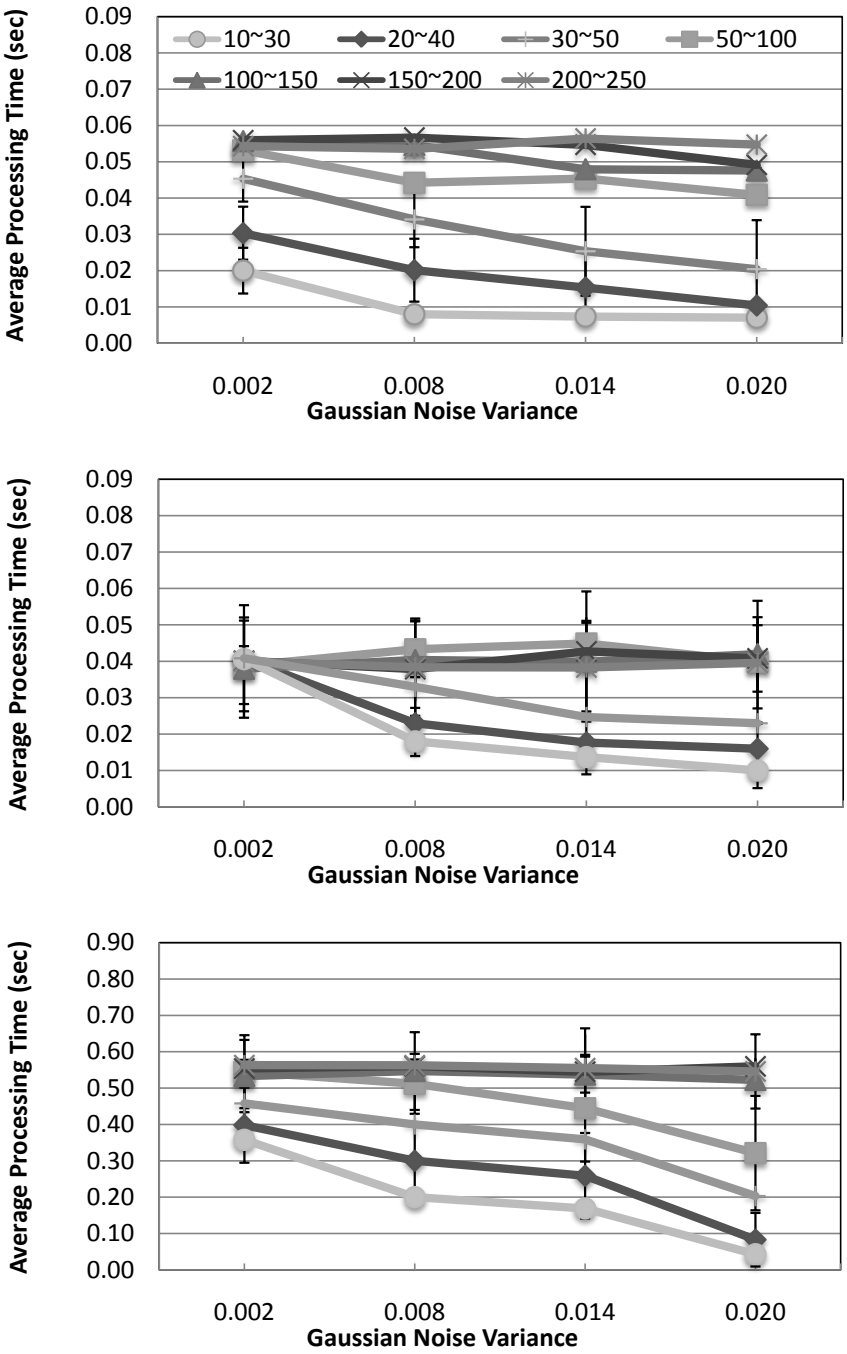


Fig. 38. Means and standard deviations of the three Monte Carlo measures for average processing time (2D). Each series represents fibers' different intensity profiles (10∼30, 20∼40, 30∼50, 50∼100, 100∼150, 150∼200, and 200∼250). Top: Line data. Middle: Curve data. Bottom: Spiral data.

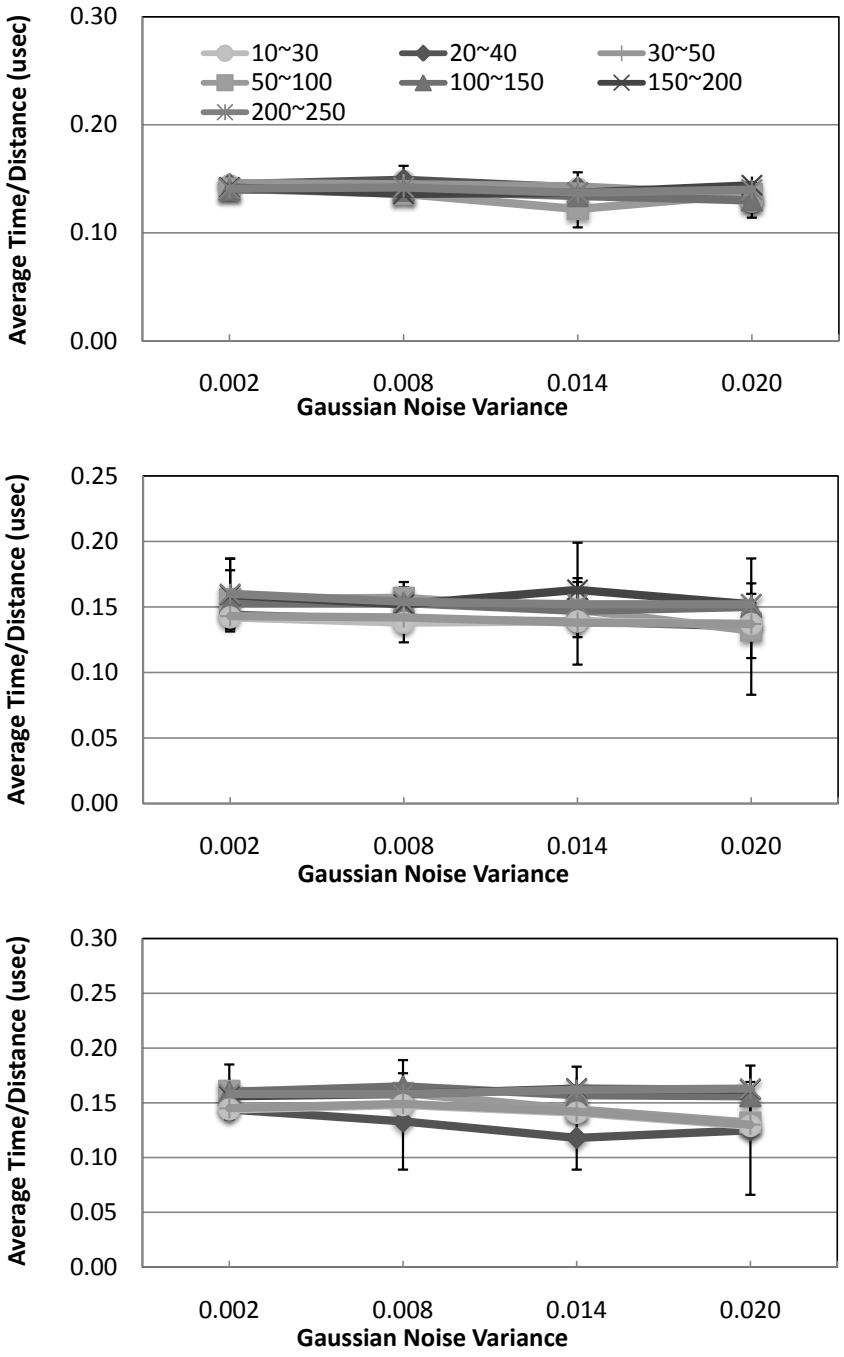


Fig. 39. Means and standard deviations of the three Monte Carlo measures for average time/distance (2D). Each series represents fibers' different intensity profiles (10∼30, 20∼40, 30∼50, 50∼100, 100∼150, 150∼200, and 200∼250). Top: Line data. Middle: Curve data. Bottom: Spiral data.

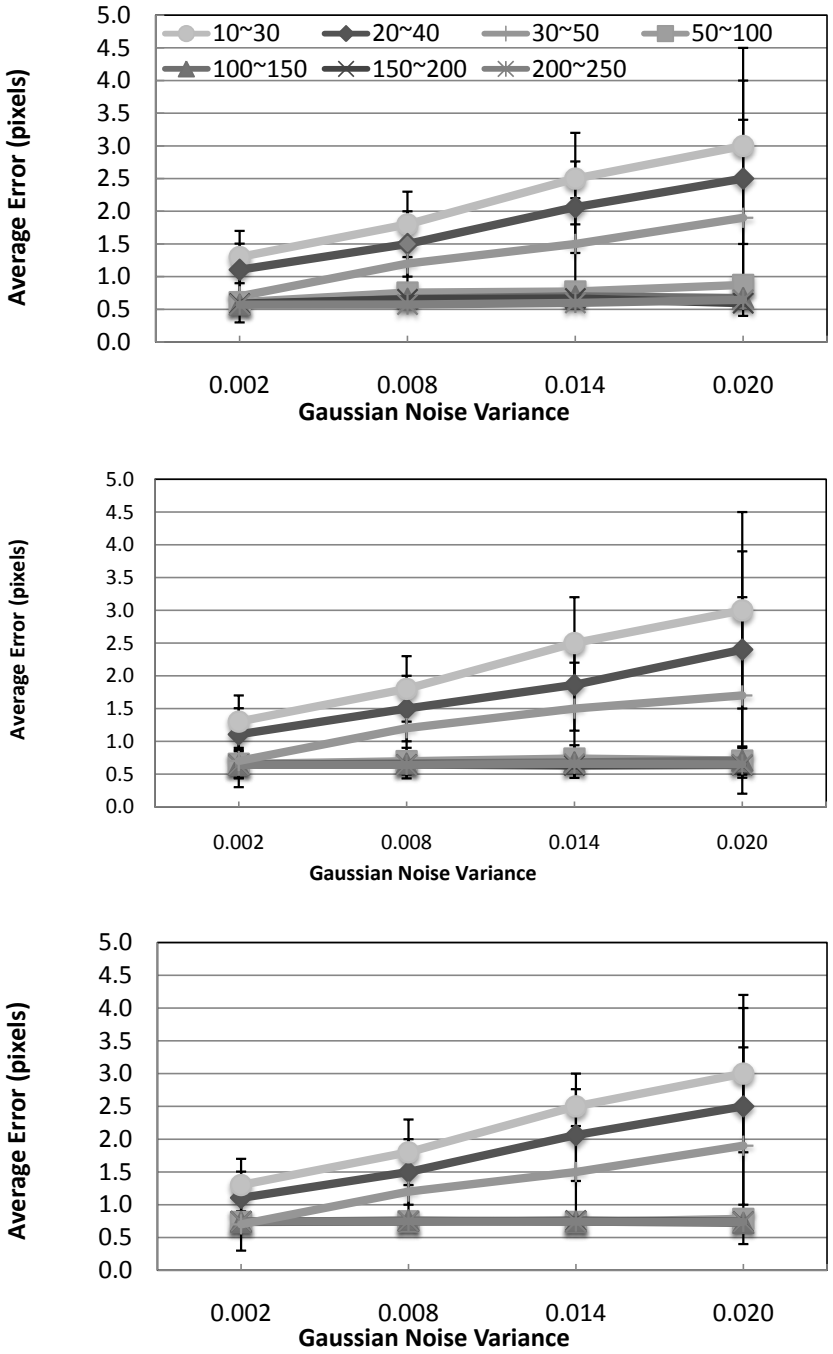


Fig. 40. Means and standard deviations of the three Monte Carlo measures for average error (2D). Each series represents fibers' different intensity profiles (10∼30, 20∼40, 30∼50, 50∼100, 100∼150, 150∼200, and 200∼250). Top: Line data. Middle: Curve data. Bottom: Spiral data.

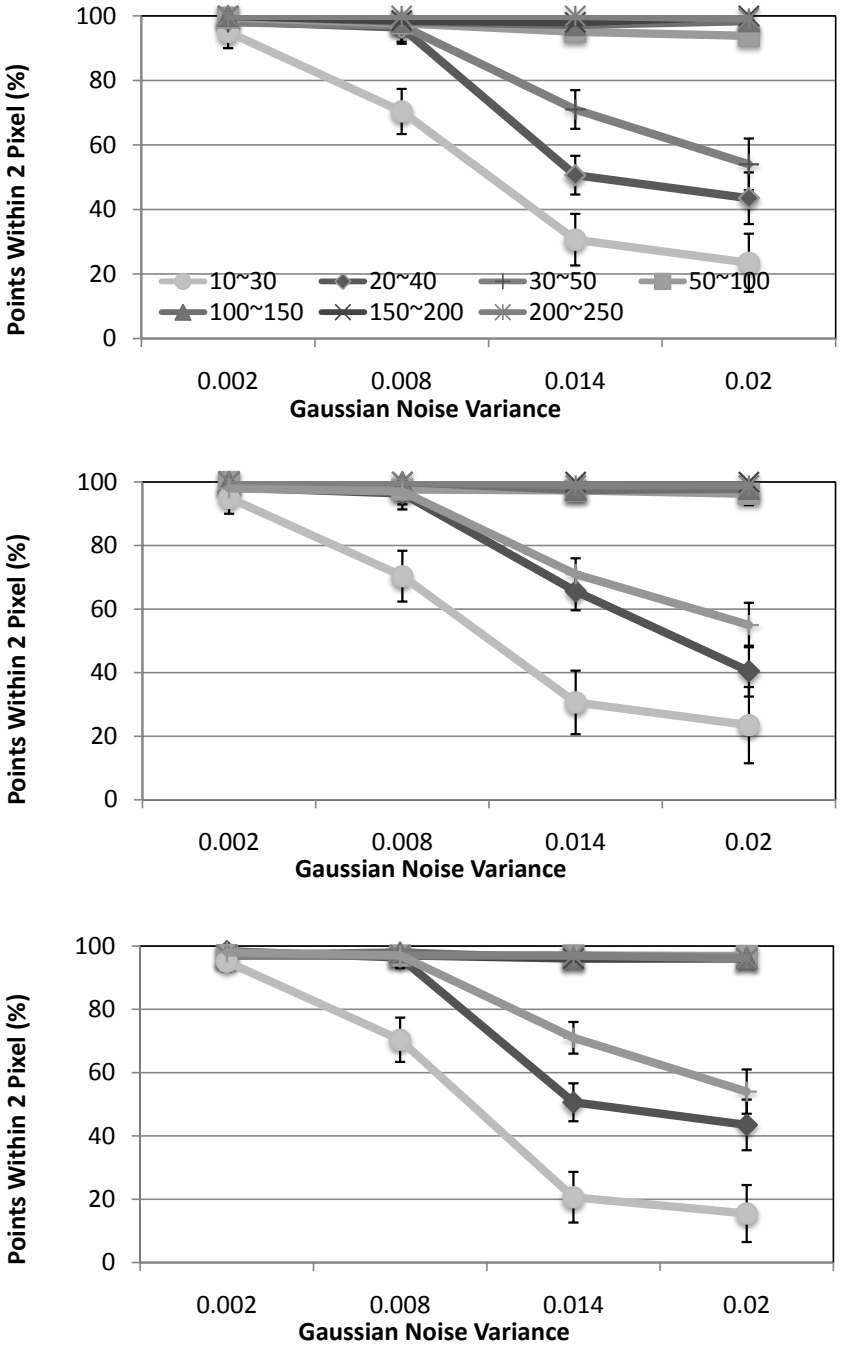


Fig. 41. Means and standard deviations of the three Monte Carlo measures for percent of points within 2 pixels of their closest medial axis point (2D). Each series represents fibers' different intensity profiles (10∼30, 20∼40, 30∼50, 50∼100, 100∼150, 150∼200, and 200∼250). Top: Line data. Middle: Curve data. Bottom: Spiral data.

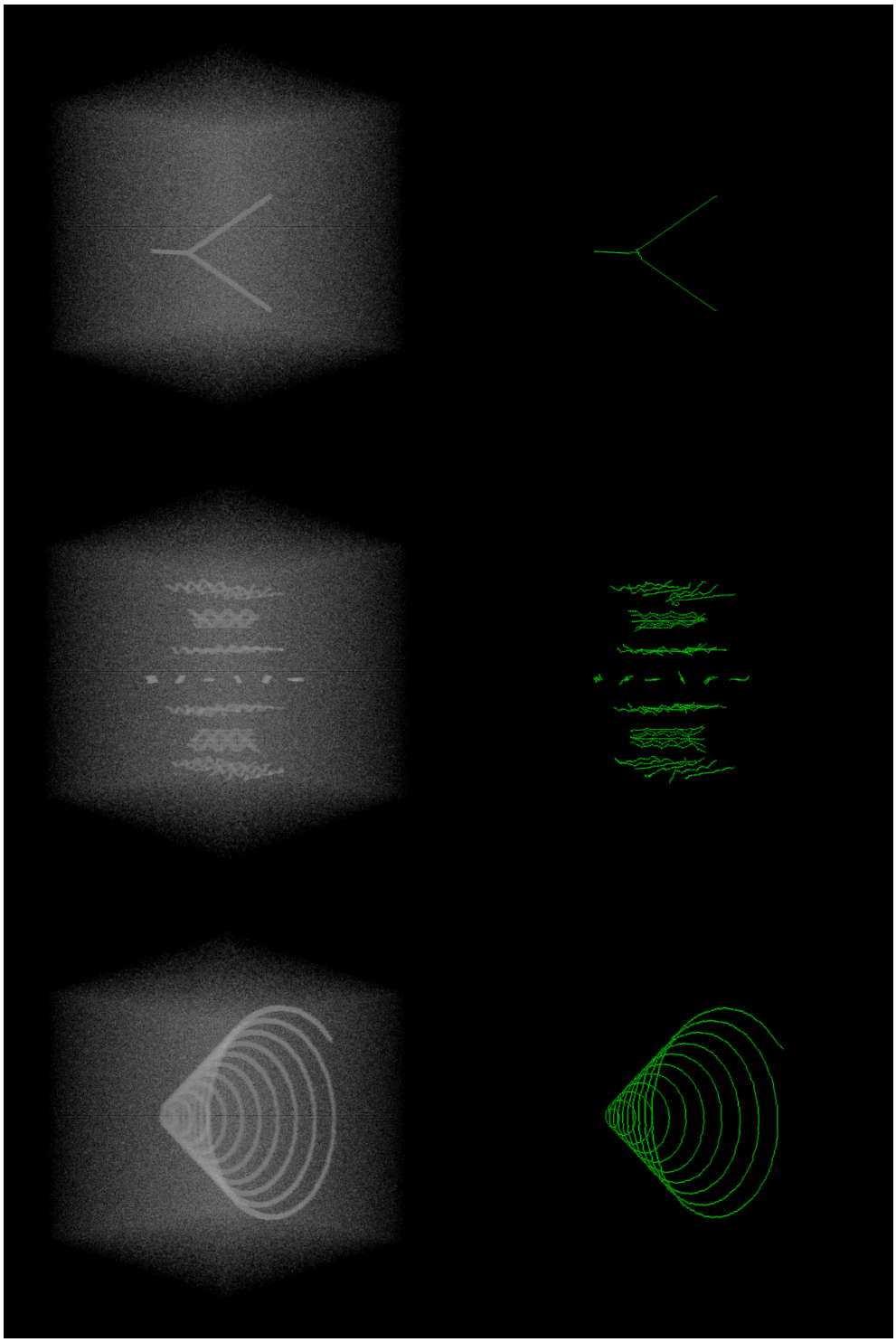


Fig. 42. Synthetic 3D data. Left column: Original data. Right column: Traced result. Top: Branched data. Middle: Stacked data. Bottom: Spiral data. The standard deviation of the Gaussian noise was 0.020. Intensity profile ranged from 50 to 100.

data (about 20 sec) in Fig. 43, which demonstrates that the time complexity is linear, independent of data type, noise, and intensity levels. Like 2D validation, only low contrast data whose intensity values range from 10 to 50 had very low processing time. This is due to the early termination of tracing caused by the combination of noise and low contrast. However, the average processing time/distance (about 0.125 sec of Fig. 44) was almost constant over all volume data. Fig. 45 shows all the tracing results have less than average error of 1 voxel except the lowest contrast data. Fig. 46 shows more than 95.6% of all the tracing points are within 2 voxels of their closest medial axis point except the lowest contrast data. However, except the lowest one, the other data contain intensity profiles from 50 to 250. These results show that my tracing method with MIP processing can be used for a wide variety of medical volume data.

E. Conclusion

In this chapter, for 2D/3D tracing validation, I made six types of digital phantoms (2D: line, curve, and spiral data, 3D: branch, stacked curve, and spiral data) with various contrasts and noise levels. Using these digital phantoms, I conducted Monte Carlo experiments for checking the processing time and noise robustness. The experiments showed that (1) my method had linear computational processing time in terms of fiber length and (2) my method can effectively deal with noise representative of medical data (MR, CT, and ultrasound). These digital phantoms have limited size (2D: 256×256 pixels, 3D: 256×256×256 voxels) and there is no quantitative evidence about how close the phantoms are to real vascular data, which will be addressed in the next chapter.

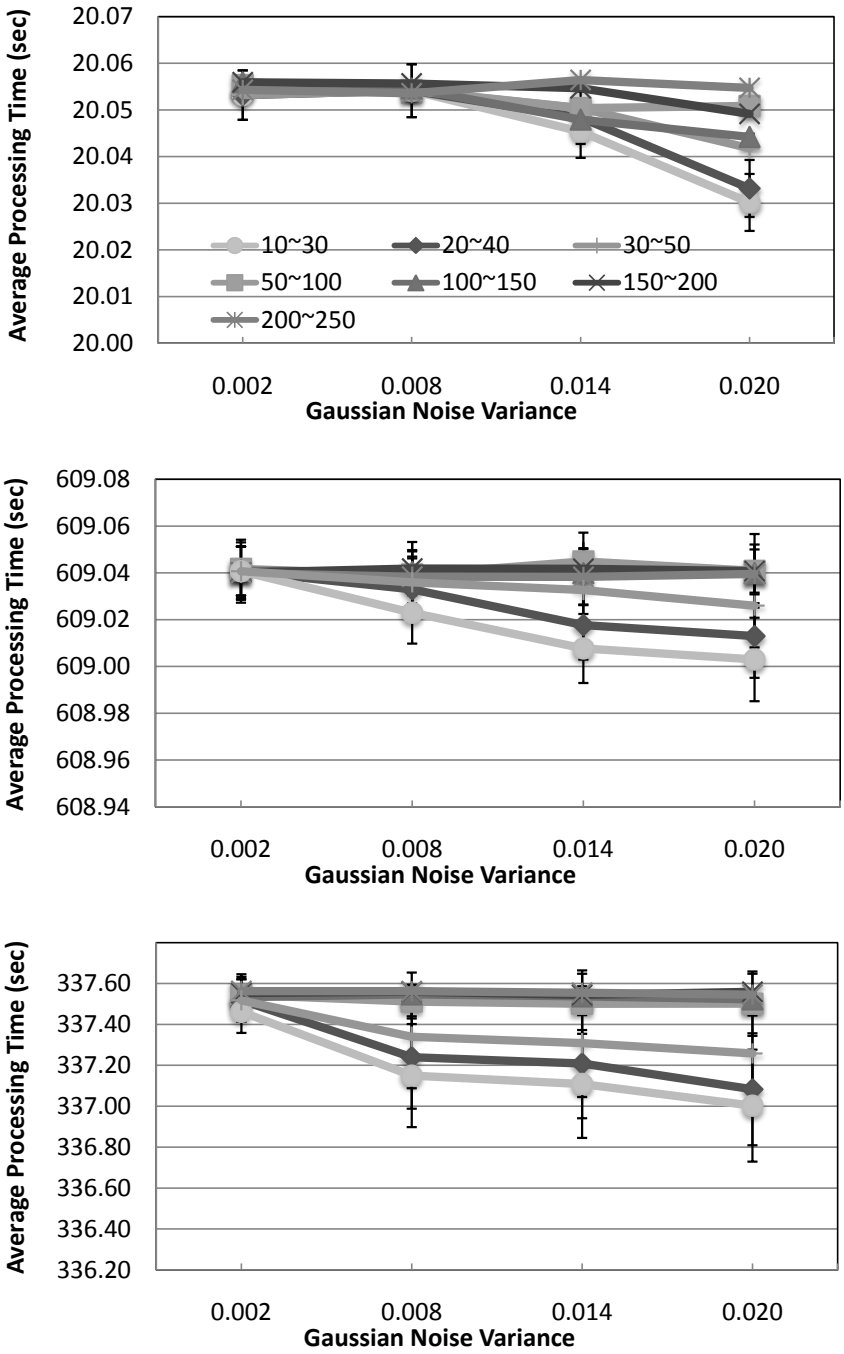


Fig. 43. Means and standard deviations of the three Monte Carlo measures for average processing time (3D). Each series represents the fiber's different intensity profile (10∼30, 20∼40, 30∼50, 50∼100, 100∼150, 150∼200, and 200∼250). Top: Branch data. Middle: Stacked curve data. Bottom: Spiral data.

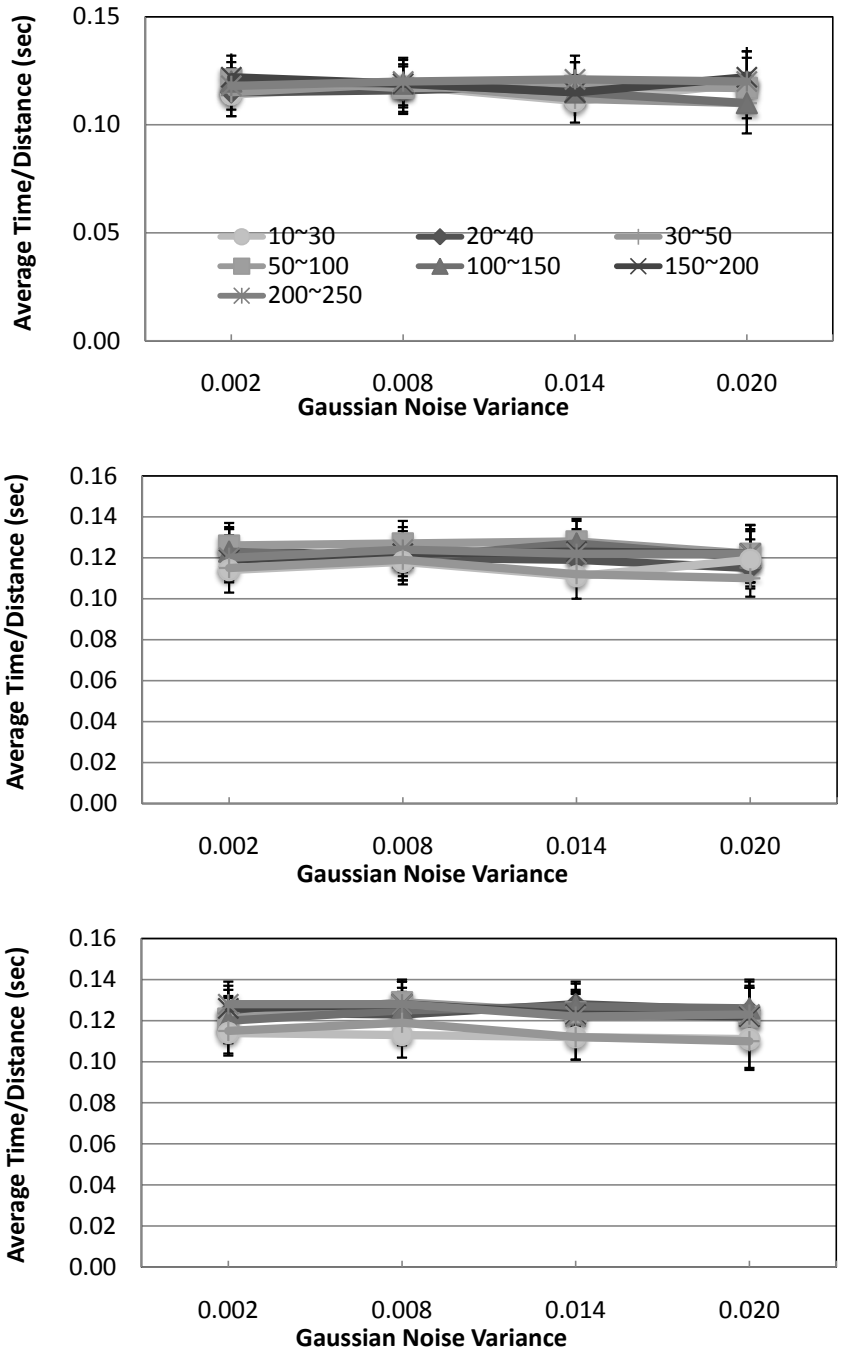


Fig. 44. Means and standard deviations of the three Monte Carlo measures for average time/distance (3D). Each series represents the fiber's different intensity profile (10∼30, 20∼40, 30∼50, 50∼100, 100∼150, 150∼200, and 200∼250). Top: Branch data. Middle: Stacked curve data. Bottom: Spiral data.

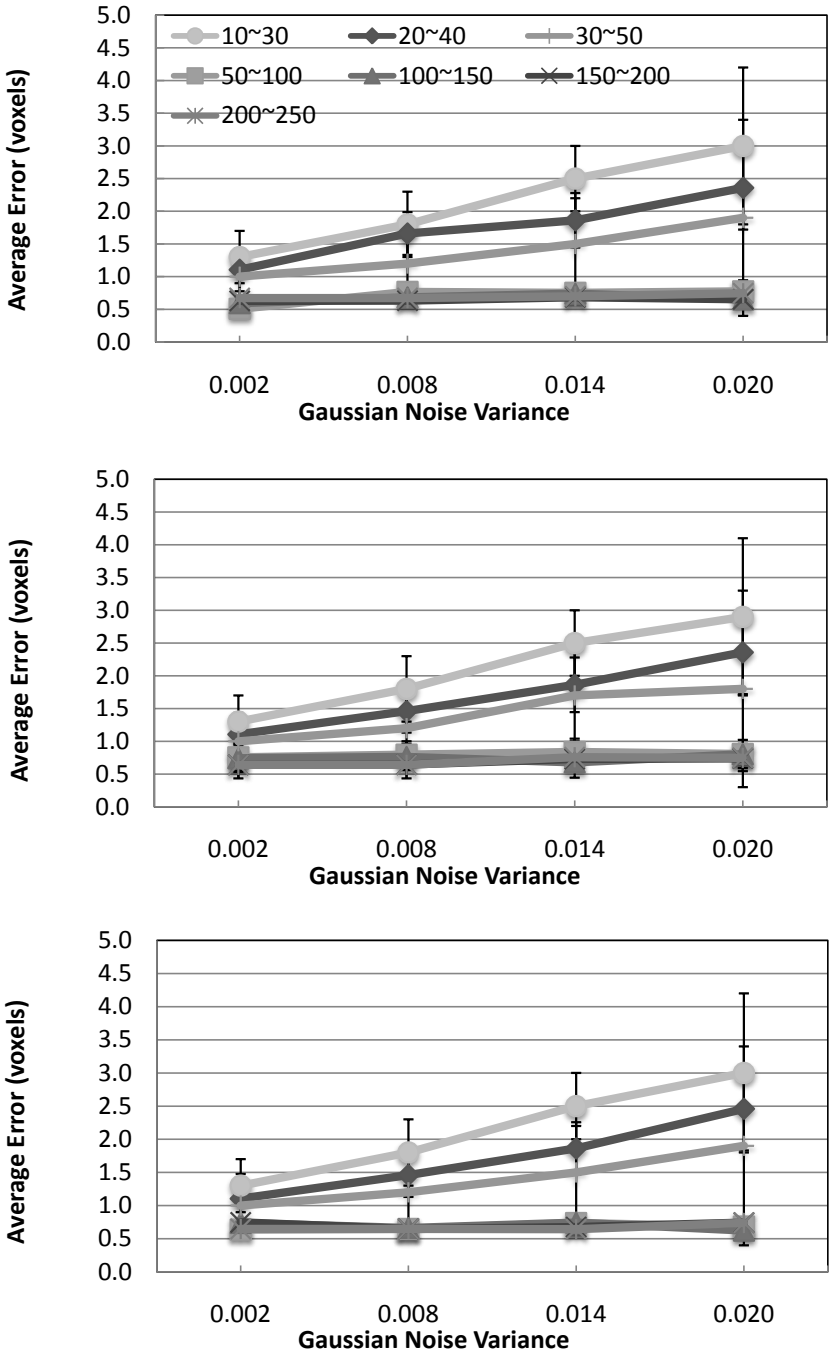


Fig. 45. Means and standard deviations of the three Monte Carlo measures for average error (3D). Each series represents the fiber's different intensity profile (10∼30, 20∼40, 30∼50, 50∼100, 100∼150, 150∼200, and 200∼250). Top: Branch data. Middle: Stacked curve data. Bottom: Spiral data.

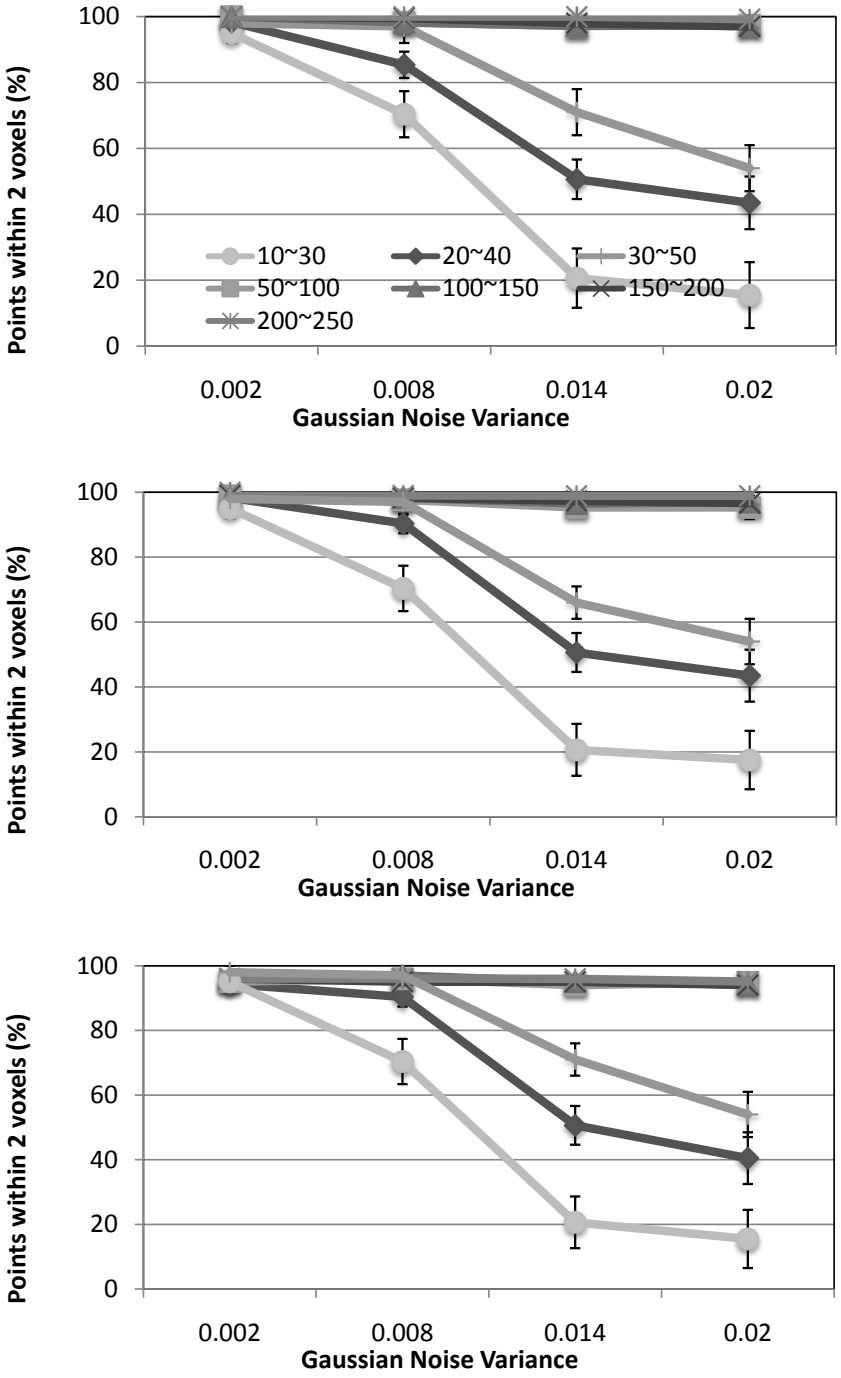


Fig. 46. Means and standard deviations of the three Monte Carlo measures for percent of points within 2 voxels of their closest medial axis point (3D). Each series represents the fiber's different intensity profile (10∼30, 20∼40, 30∼50, 50∼100, 100∼150, 150∼200, and 200∼250). Top: Branch data. Middle: Stacked curve data. Bottom: Spiral data.

CHAPTER VI

MODEL-BASED VALIDATION

In the previous chapter, I introduced two validation methods. First, I made synthetic data for 2D/3D algorithms. These synthetic data cover most cases of neurovascular models in terms of geometry. As stated above, I also combined noise representative of medical data. This combination is enough for my algorithm to be validated with Monte Carlo method. This validation is based on the validation method conducted in [61]. Second, I carried out validation with human-labeled ground truth based on [84]. It is also applicable to 2D/3D algorithms. Human-labeled validation compares the human-labeled ground truth with the tracing results. To assess the difference/similarity, I checked the p-values using two-sided paired t-test.

However, the first validation method does not guarantee that the synthetic data can cover all the real data types. The second validation method is not suitable for large-scale networks. Manual labeling requires too much time for large-scale data. Therefore, I need a new validation method for large-scale realistic data. Motivated by this, I will introduce a new model-based validation method.

There are two main parts of the validation system:

1. Validation of the model-based generator: Fig. 47 shows how to validate the model-based generator. For the ground truth, I will use statistical information extracted from neurovascular data using a thinning method [18]. In order to make the information from the thinning method into ground truth, two additional methods are needed: (1) a threshold that separates vessels from the background and (2) distance map computation that computes for each vessel's point its shortest distance to the background. Cassot et al. set thresholds manually by experts on all slices (70 sections for 207μm thick data) of the experiment

data to obtain the ground truth [18]. In a similar manner, Cassot et al. obtained the radius of vasculatures using the distance map. Therefore, the statistical information will include the number, length, and diameter of vasculatures. I will also define several parameters based on the analysis of the thinning results. These parameters will be used by the model-based generator for yielding other statistical information including the number, length, and diameter of vasculatures. The two sets of statistical information will be compared for checking the correctness of the model-based generator.

2. Validation of 3D tracing algorithm using the model-based generator: Fig. 48 shows how to validate my 3D tracing algorithm. Using the synthetic ground-truth data set generated from the model-based generator, I will generate a realistic synthetic vascular data set. The synthetic ground-truth data set contains precise geometric information such as length and diameter of fibers, and the realistic synthetic vascular data set is image/volume data constructed from the synthetic ground-truth. My 3D algorithm will then trace the vasculatures from the realistic synthetic vascular data set. Then, the tracing results will be compared to the synthetic ground-truth data set to validate my 3D tracing algorithm.

In the following sections, I introduced parameters for the model-based generator and showed how to yield synthetic ground-truth vasculature data sets using the model-based generator. Next, I validated the model-based generator using statistical information. After that, I validated my tracing method using the model-based generator with noise levels used in the previous chapter. Finally, I showed the tracing results of the whole mouse brain obtained by KESM.

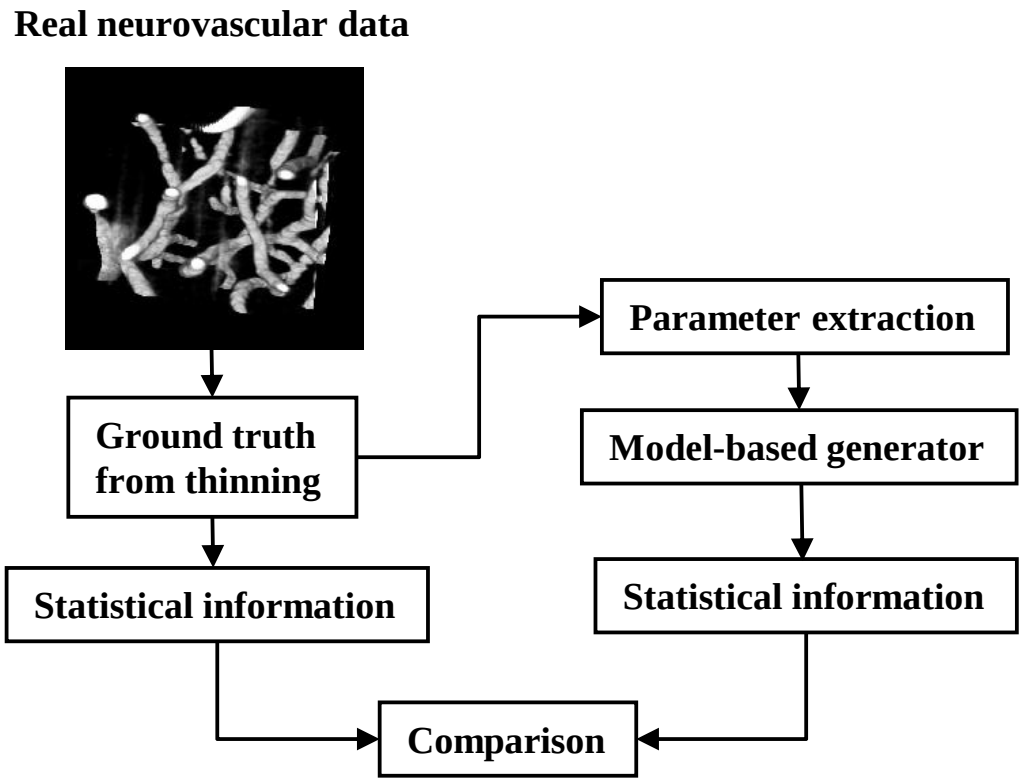


Fig. 47. Validation of the model-based generator. Statistical information will be extracted from the ground truth of real neurovascular data using a thinning method [18]. The statistical information will include the number, length, and diameter of vasculatures. The parameters will be defined based on the analysis of the thinning method. Using the parameters, the model-based generator will also yield statistical information. The two sets of statistical information will be compared to check the correctness of the model-based generator.

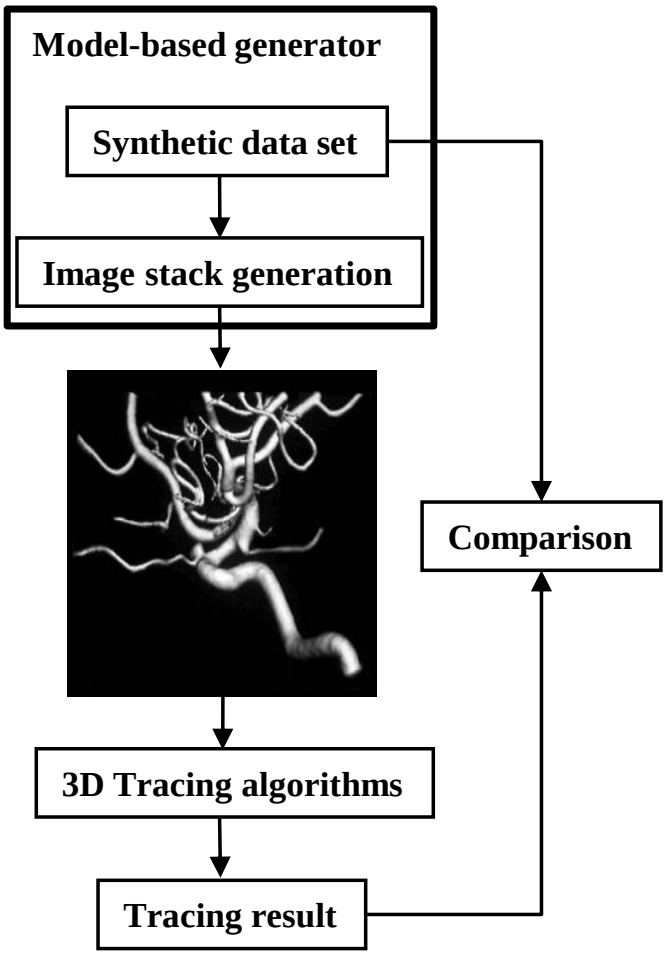


Fig. 48. Validation of 3D tracing algorithm using the model-based generator. The model-based generator will yield the synthetic ground-truth data set. Based on the synthetic ground-truth data set, I can generate a realistic synthetic vascular data through the image generation process. My 3D algorithm will trace the vasculatures from the realistic synthetic vascular data. Then, the tracing results will be compared to the synthetic ground-truth data set for validating my 3D tracing algorithm.

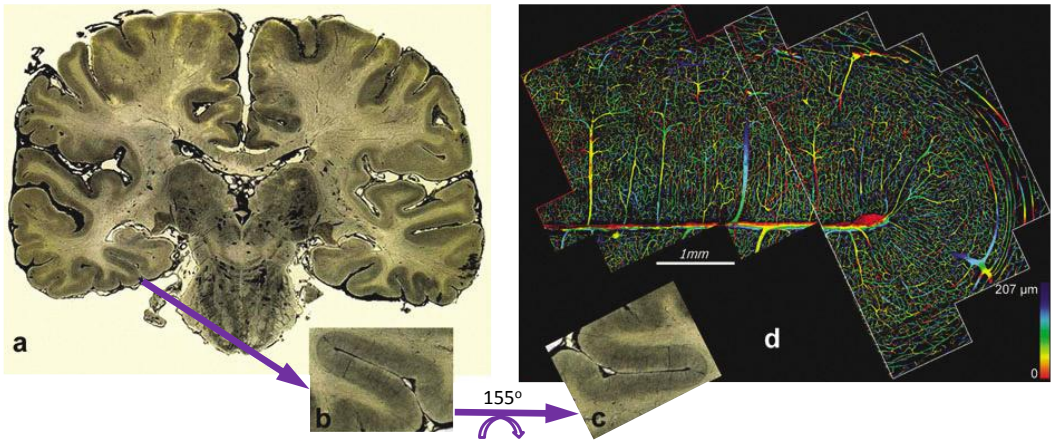


Fig. 49. A thick section of India ink-injected human brain. (a) View of the whole section. (b) The collateral sulcus in the temporal lobe. (c) Rotated view of (b). (d) Depth coded projection of the zone imaged by confocal microscopy [18].

A. Parameter extraction from real vascular data

In order to get biologically meaningful parameters, as described in Fig. 47, I used human brain microvascular networks extracted from the collateral sulcus in the temporal lobe (Fig. 49). Cassot et al. developed algorithms adapted to very large data sets to automatically extract and analyze center lines together with diameters of thousands of brain microvessels within a large cortex area from thick sections of india ink-injected human brain, using confocal laser microscopy [18]. From this method, the authors could easily obtain the geometry of human brain microvessels as shown in Fig. 50.

Using the same method, we can obtain 2D and 3D morphometry information. However, comparison between the results given by 2D and 3D morphometry has shown that only 3D morphometry is able to obtain reliable data on highly complex vascular networks and that 2D morphometry should be limited to the analysis of flat two-dimensional vascular networks [76]. Therefore, I use 3D morphometry information extracted from a thinning method used by Cassot et al. in order to get

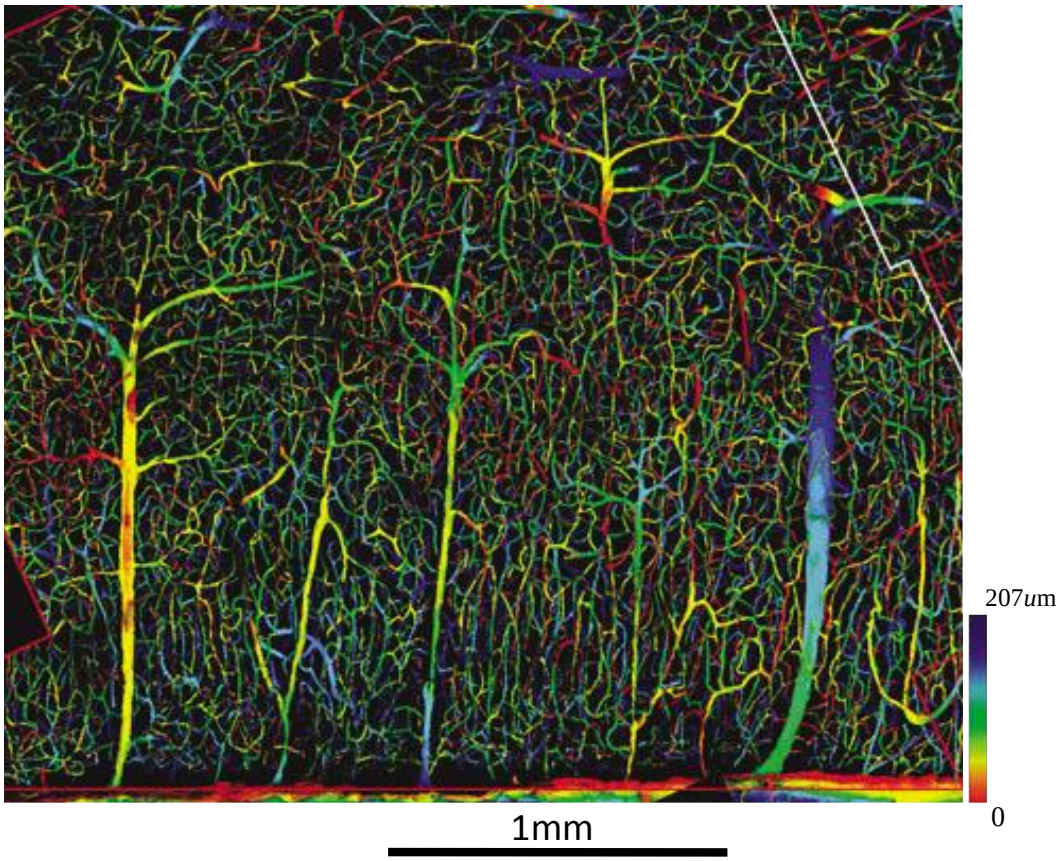


Fig. 50. Depth-coded projection of a part of Fig. 49 [18]. Cassot et al. provide an algorithm which gives a representation of the vascular network as a set of cylinders centered at the center line points, the radii of which corresponded to the distance map values at these points. This line set can be visualized and superimposed for different types of 3D visualization of the original confocal microscope images for representation and control purposes (Iso-surface, volume rendering, and projection views with a color scale with the line set information).

the parameters. This 3D morphometry will be considered as a ground truth for my model-based generator.

Among the many parameters that can be analyzed by this method, the frequency distributions of diameters and lengths of the vessels (elements), and the number of vessels are all vital features for an adequate model of cerebral microcirculation. These three parameters are the base information of constructing a neurovascular network to globally assess the microvascular architectural complexity [85]. These parameters can also characterize the geometry of such a network with the centerline information [86]. Therefore, the parameters to be used by the model-based generator are:

- the diameter of a vessel element.
- the length of a vessel element.
- the number of vessel elements.

In order to get the diameter, length, and number of vessels, we need to separate the vessels into segments based on the criteria as shown in Fig. 51 [87]. This criteria give an order number to every vessel segment. This ordering scheme starts with order 0 for the terminal vessel. When two segments of the same order n meet with each other, the confluent is called a vessel of order $n+1$ if and only if the diameter of the confluent segment is greater than $[(D\mu_n + D\sigma_n) + (D\mu_{n+1} - D\sigma_{n+1})]/2$, where $D\mu_n$ and $D\sigma_n$ denote the mean and standard deviation of the diameters of the vessels of order n as shown in Fig. 51(b). Several vessel segments of the same order are connected in a series. I called their combination an element [18].

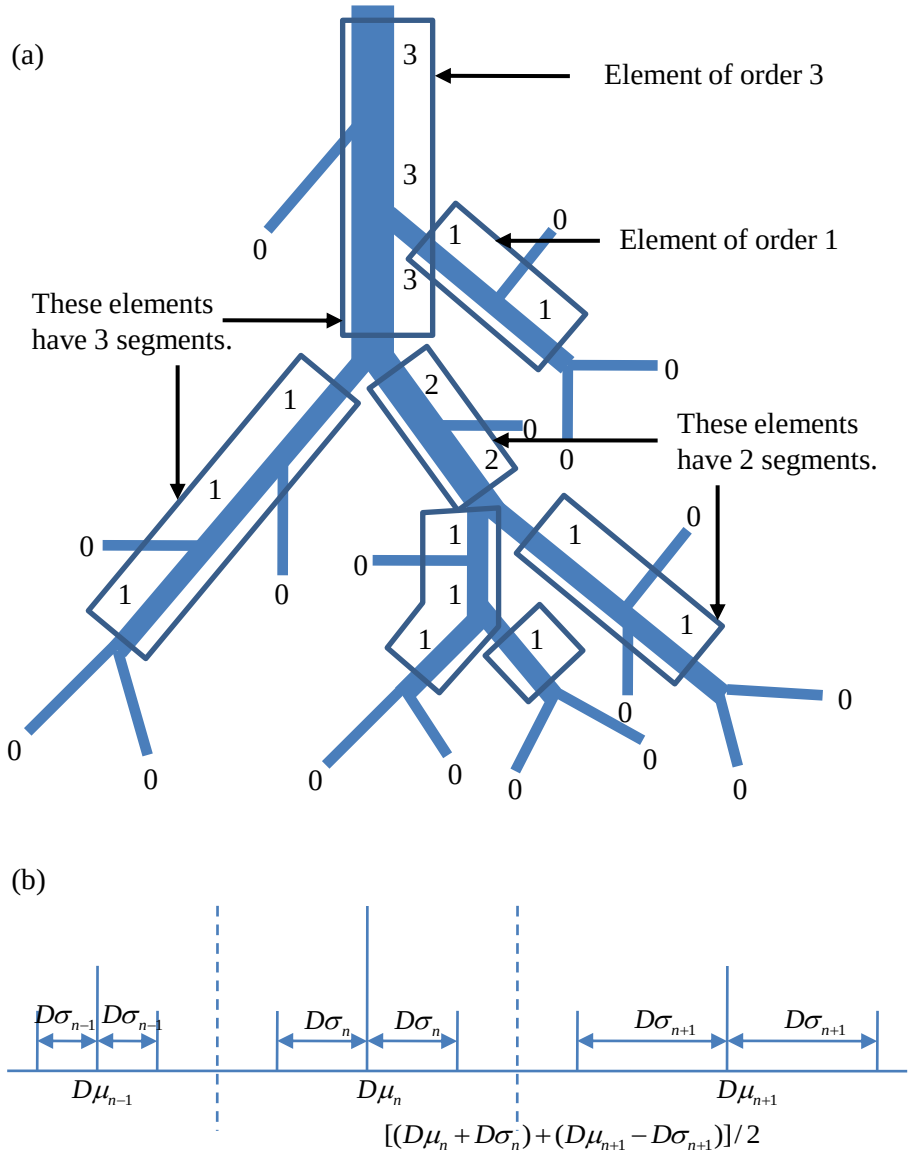


Fig. 51. The definition of segments and elements. Segments and elements are described based on the order numbers [87]. (a) Vessel order numbers are determined by their connection and diameters. Each vessel between 2 successive points of bifurcation is defined as a segment. Segments of the same order connected in series are defined as an element. Smallest segments and elements are of order 0. (b) When two elements meet, order number is that of larger element (assumed to be n) increased by 1 (to order n+1) if diameter of confluent element is larger than $[(D\mu_n + D\sigma_n) + (D\mu_{n+1} - D\sigma_{n+1})]/2$ (right dashed line), where $D\mu_n$ and $D\sigma_n$ denote the mean and standard deviation of the diameters of the vessels of order n. Order numbers are defined using the dashed lines.

B. Model-based generator

In this section, an algorithm for the construction of a 3-D vascular trees is presented. The model is constructed utilizing a random number generator for the three parameters such as the diameter, length, and number of elements defined in section A of chapter VI. The basic principle of the vascular model is to have branches at some location and to have proper diameter for the branches. The branch's location and the branch's diameter are obtained from the information of length and diameter of the vessel elements [18]. After getting the branch's diameter, the branch's order number should be assigned. The branch's order number can decrease by 1 based on the diameter of the branch. We begin with the main vessel with order number 4 based on the ground truth [18]. The variation of each parameter is given by a Gaussian distribution based on the process of morphogenesis [88].

The following is the generation procedure of a neurovascular tree using the above information (the branch's location and the branch's diameter).

1. Initial vasculature

Initial vessel's diameter and order number are determined from Cassot et al.'s ground truth. In Fig. 52(a), the main stem has two branches. I set the main stem's order number to 4 and two branches' order number to 3. The main stem's diameter is assigned following a Gaussian distribution ($\mu = 0.0$ and $\sigma = 1.0$) around $D\mu_4$ where $D\mu_4$ is the mean of the diameter of element of order 4. The main stem's length is assigned following a Gaussian distribution ($\mu = 0.0$ and $\sigma = 1.0$) around $L\mu_4$ where $L\mu_4$ is the mean of the length of element of order 4. Two branches' diameter and length are assigned following a Gaussian distribution ($\mu = 0.0$ and $\sigma = 1.0$) around $D\mu_3$ and $L\mu_3$ respectively. All $D\mu_n$ and $L\mu_n$ ($n = 0, .., 4$) are obtained from Cassot

et al.'s ground truth [18].

2. Branch generation for elements of order 3 and 4

In Fig. 52(b), based on the percentage of location of branch point's ground truth in main stem, branch points are assigned on both the main stem and the two branches [18]. The diameter of each branch is assigned following a Gaussian distribution ($\mu =$ 0.0 and $\sigma = 1.0$) around $D\mu_n$, where n is order number of each branch and $D\mu_n$ is the mean of diameter of element of order n.

3. Generation step size for elements of order 0, 1, and 2

Fig. 53(a) shows how to calculate a generation step size. ϑ_i is the i^{th} generation step size and it is randomly selected from $5 \leq \vartheta_i \leq (5 + D\mu_n \times 3)$ where $D\mu_n$ is the mean of diameter of element of order n.

4. Generation step direction for elements of order 0, 1, and 2

Fig. 53(b) shows how to calculate a generation step direction. $\vec{v}_{i+1}$ is the $(i+1)^{th}$ generation step direction and it is derived from $\vec{v}_{i+1} = \vec{u}_{i+1} \times \vartheta_{i+1}$ where $\vec{u}_{i+1} = (\vec{v}_i + \vec{\alpha}_i)/|\vec{v}_i + \vec{\alpha}_i|$ and $\vec{\alpha}_i$ is a random vector on the i^{th} generation step. $\vec{\alpha}_i$ is defined as $\vec{\alpha}_i = (\alpha_x, \alpha_y, \alpha_z)$, α_x is randomly selected from -5 to 5, α_y is randomly selected from -5 to 5, and α_z is randomly selected from -10 to 10. These ranges can be changed by the user for more diversity.

5. Calculation of length for elements of order 0, 1, and 2

The element's length is assigned following a Gaussian distribution ($\mu = 0.0$ and $\sigma = 1.0$) around $L\mu_n$ where $n = 0, 1, 2$.

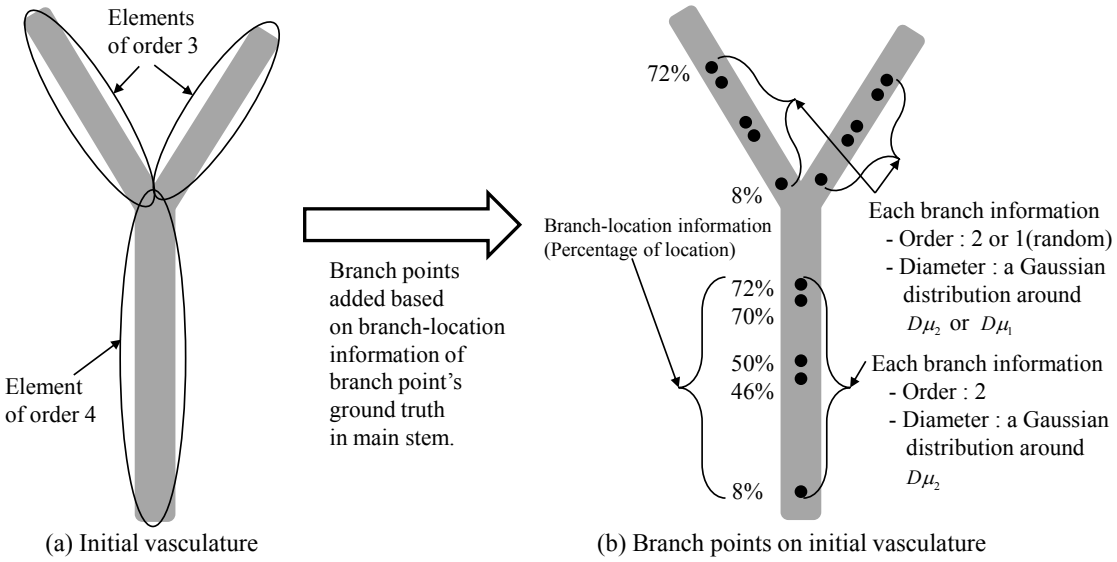
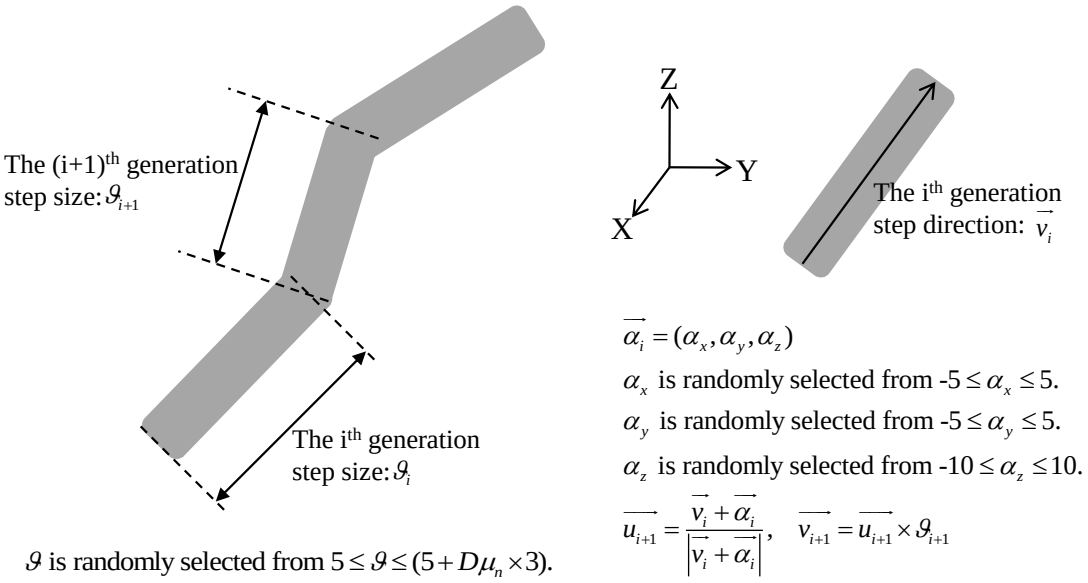


Fig. 52. Initial vasculature generation (order $n = 3, 4$). (a) Main stem (order 4) has two branches (order 3). The main stem's order number is set to 4 and two branches' order number to 3. The main stem's diameter is assigned following a Gaussian distribution ($\mu = 0.0$ and $\sigma = 1.0$) around $D\mu_4$ where $D\mu_4$ is the mean of the diameter of order 4. The main stem's length is assigned following a Gaussian distribution ($\mu = 0.0$ and $\sigma = 1.0$) around $L\mu_4$ where $L\mu_4$ is the mean of the length of order 4. Two branches' diameter and length are assigned following a Gaussian distribution ($\mu = 0.0$ and $\sigma = 1.0$) around $D\mu_3$ and $L\mu_3$ respectively. (b) Based on the branch-location information of the branch point's ground truth in main stem, branch points are assigned on both the main stem and the two branches [18]. Diameter of each branch is assigned following a Gaussian distribution ($\mu = 0.0$ and $\sigma = 1.0$) around $D\mu_n$, where n is order number of each branch and $D\mu_n$ is the mean of the diameter of element of order n.



(a) Generation step size calculation (b) Generation step direction calculation

Fig. 53. Vessel element generation (order $n = 0, 1, 2$). (a) Generation step size (ϑ_i) calculation for the i^{th} generation step. ϑ_i is randomly selected from $5 \le \vartheta_i \le (5 + D\mu_n \times 3)$ where $D\mu_n$ is the mean of diameter of element of order n. (b) Generation step direction ($\vec{v}_{i+1}$) calculation for the $(i+1)^{th}$ generation step. $\vec{v}_{i+1}$ is derived from $\vec{v}_{i+1} = \vec{u}_{i+1} \times \vartheta_{i+1}$ where $\vec{u}_{i+1} = (\vec{v}_i + \vec{\alpha}_i)/|\vec{v}_i + \vec{\alpha}_i|$ and $\vec{\alpha}_i$ is a random vector on the i^{th} generation step.

6. Branch generation for elements of order 0, 1, and 2

Each element has its own branches based on the information of percentage of location. From now on, I will call the percentage of location the branch-location information. Even though the branch-location information of branch point's ground truth is defined for main stem (order 4), I will use this same branch-location information for elements of order 1 and 2 (there is no branch-location information of branch point's ground truth for elements of order 1 and 2). Element of order 0 does not have branches. The order n element's branch diameter and length are assigned following a Gaussian distribution around $D\mu_{n-1}$ and $L\mu_{n-1}$ respectively.

By using the procedure above, the model-based generator yields vascular trees using different random seed numbers as shown in Fig. 54. The order number ranged from 0 to 4. The volume size was $528\times52\times625$ voxels. Each voxel was 0.25μm$\times0.25\mu$m$\times0.25\mu$m based on the ground truth [18]. The average values regarding the length, diameter, and number of elements in Fig. 54 were as follows (Table. V).

Table V. The mean values for the length, diameter, and number of elements from the model-based generator.

order	length (μm)	diameter (μm)	number of elements
0	184.8	7.8	163.9
1	270.2	10.7	48.3
2	350.2	16.0	15.1
3	830.4	18.2	2.0
4	1559.7	31.8	1.0

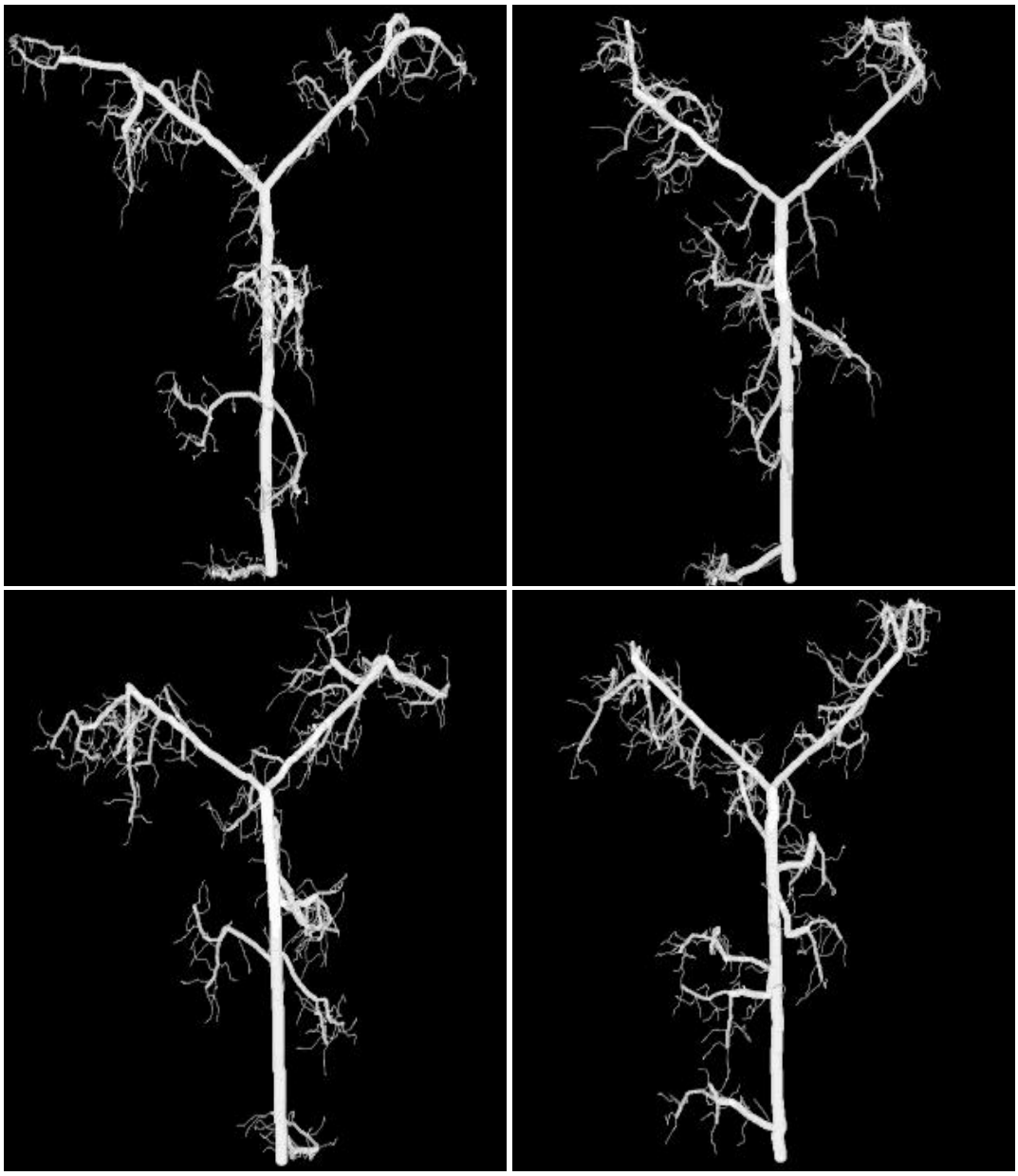


Fig. 54. Vasculatures generated from the model-based generator. See Fig. 56 for tracing results.

C. Validation of the model-based generator

In this section, I validated the model-based generator comparing the ground truth to the synthetic data generated from the model-based generator. As described in section VI.A, the ground truth was obtained from Cassot et al.'s human brain microvascular network using a thinning method. They provided the vascular tree information regarding these three parameters: diameter, length, and number of vessel elements based on the order number.

Fig. 55(a) shows the relationship between the total number of elements N_n and the order number n. The synthetic data has slightly more elements at order 2 and 1. This is due to the fact that some collisions, during generating vasculatures, make other branches. These branches increase the number of elements. Fig. 55(b) shows the relationship between the mean vessel diameter D_n and the order number n. Compared to the ground truth, the synthetic data has higher diameter at order 1 and lower diameter at order 3. However, the overall trend still looks similar. Fig. 55(c) shows the relationship between the mean vessel element length L_n and the order number n for the synthetic vs. ground truth. We can say that they also look similar.

In these figures, we can have the curves fitted by an equation of the form

$$\log\hat{\beta}_n = a + bn \tag{6.1}$$

where $\hat{\beta}_n$ represents N_n, D_n, and L_n and a and b are constants. The coefficients a and b are obtained by linear regression, as follows:

$$b = \frac{\sum_{i=0}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=0}^{n}(x_i - \bar{x})^2} \quad \text{and} \quad a = \bar{y} - b\bar{x} \tag{6.2}$$

where $\bar{x}$ is the mean of the n values and $\bar{y}$ is the mean of $\log\hat{\beta}_n$, x_i is the value of the i^{th} order number, and y_i is the value of $\log\hat{\beta}_i$. By using this regression, b is positive

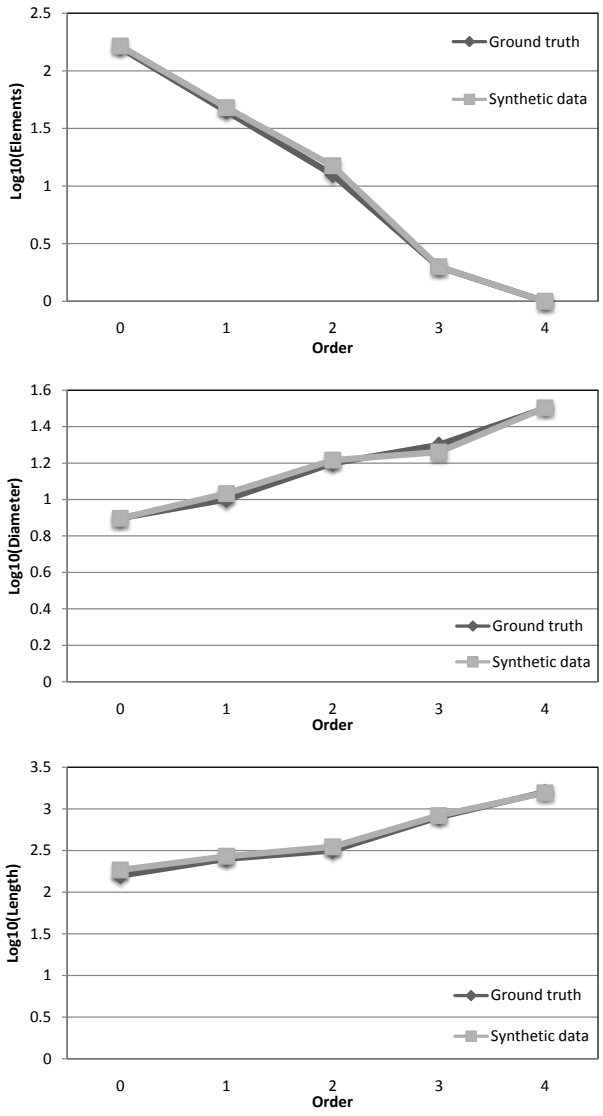


Fig. 55. The model based graphs. (a) The number of vessel elements of successive orders. Order 0 is for the terminal vessel and order 4 is for the main stem [87]. Two serieses (the ground truth from [18] and synthetic data) are compared. (b) The mean diameter (μm) of vessel elements of successive orders. (c) The mean length (μm) of vessel elements of successive orders.

for D_n and L_n, which increases with the order number, and negative for N_n, which decreases with this order. The constants a and b are listed in Table VI.

Table VI. Constants a and b for the relationships between the order number and mean diameter, length, and number of vessel elements, for ground truth and synthetic data.

Ground truth

	Diameter	Length	Number of elements
a	0.8547	2.1138	2.1942
b	0.1582	0.2875	-0.5765

Synthetic data

	Diameter	Length	Number of elements
a	0.8938	2.2031	2.2354
b	0.1441	0.2339	-0.5809

D. Validation of tracing method with model-based generator

Fig. 56 shows the traced results for the synthetic vasculatures from the model-based generator in Fig. 54. As stated earlier, the volume size of each data was (528×52×625) voxels. Each voxel was 0.25μm×0.25μm×0.25μm based on the ground truth [18]. Fig. 57 shows the close-up of the traced results for the synthetic vasculatures from the model-based generator. The following experiments were conducted for validating my MW method with local MIP processing. From the model-based generator, I used the medial axis of the synthetic data as a ground truth. I used two measurements, length difference(ϕ) and centerline deviation(φ), to quantitatively evaluate the difference between my method's result (A) and ground truth (R) based on Zhang et al.'s validation [84]. The length difference (ϕ) is defined in Eq. 4.16, and the centerline de-

viation (φ) in Eq. 4.17. As shown in Table VII, the mean (μ) and standard deviation (σ) of the length difference (ϕ) were 0.3029 and 0.5837 voxels respectively. The mean (μ) and standard deviation (σ) of the centerline deviation (φ) were 0.8675 and 0.4733 voxels respectively. Even though the mean value indicates that most trace results are less than or equal to one voxel, the mean value is greater than 0.5 voxel. To reduce execution time, I conducted a center point adjustment only once using a momentum operator at each step. If I use a momentum operator more than one time, I expect that the centerline deviation will be decreased. This result shows that most tracing results are in fiber using one momentum operator.

Table VII. The mean (μ) and standard deviation (σ) of length difference (ϕ) and centerline deviation (φ) values for my tracing method (unit=voxel).

ϕ		φ	
μ	σ	μ	σ
0.3029	0.5837	0.8675	0.4733

In addition to this, I added various noise levels to the synthetic vasculatures with several contrast levels defined in section V.C. Fig. 58 shows the tracing results from the synthetic vasculatures generated from the model-based generator. The intensity profile ranged from 50 to 100 and the Gaussian noise had a standard deviation of 0.020.

Fig. 59 shows the means of five synthetic neurovascular trees generated from the model-based generator for processing time (top) and average time/distance (bottom). It took about 1665 seconds to trace about 13700 voxels. The average time/distance is about 0.12 seconds.

The top chart in Fig. 60 shows most tracing results have less than average error of 0.8 voxel except the lowest contrast data. The bottom chart in Fig. 60 shows more

Fig. 56. Traced results of vasculatures extracted from the model-based generator in Fig. 54.

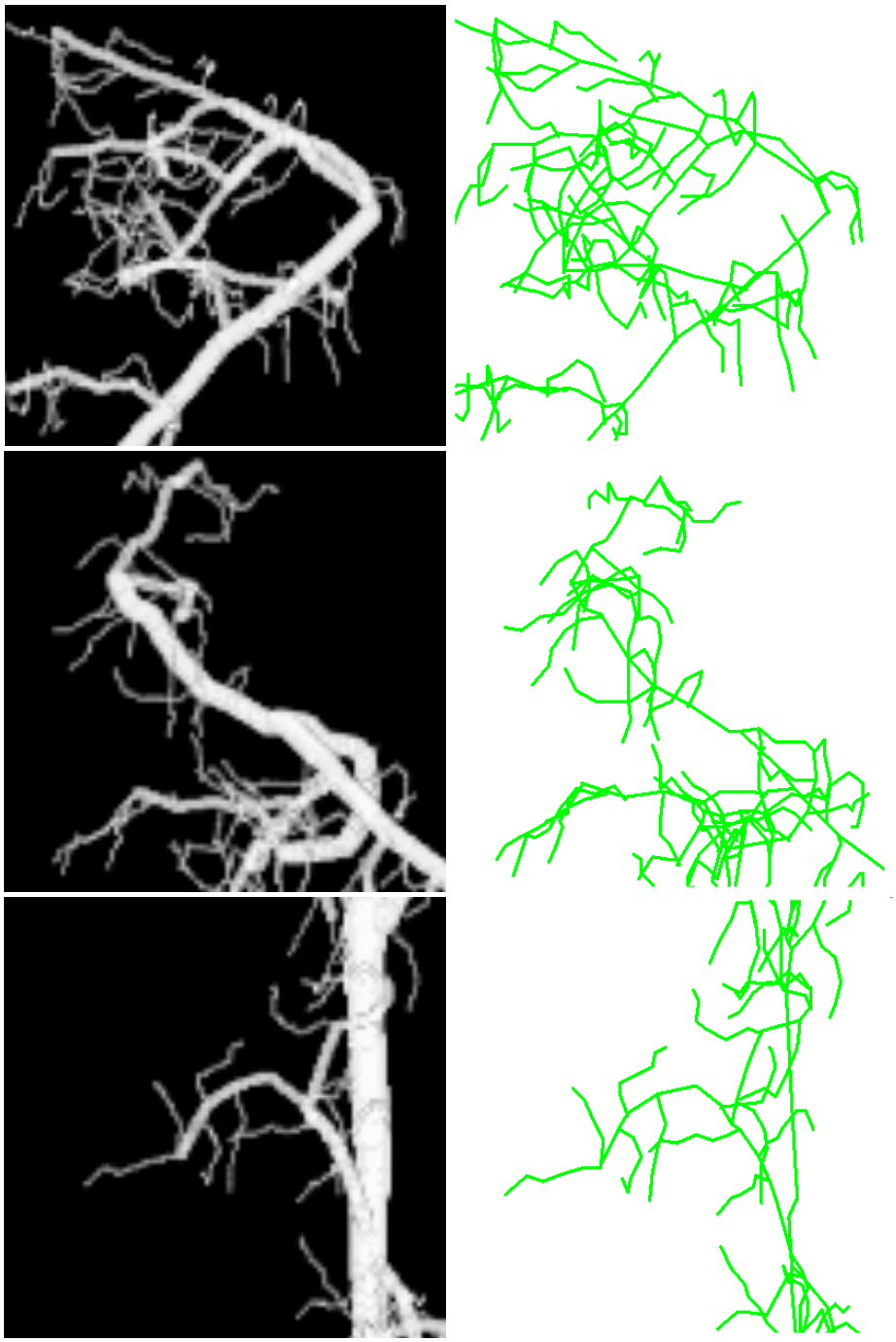


Fig. 57. Traced results using my MW with local MIP processing. Left: Close-up of vasculatures generated from the model-based generator. Right: Traced results using my MW with local MIP processing.

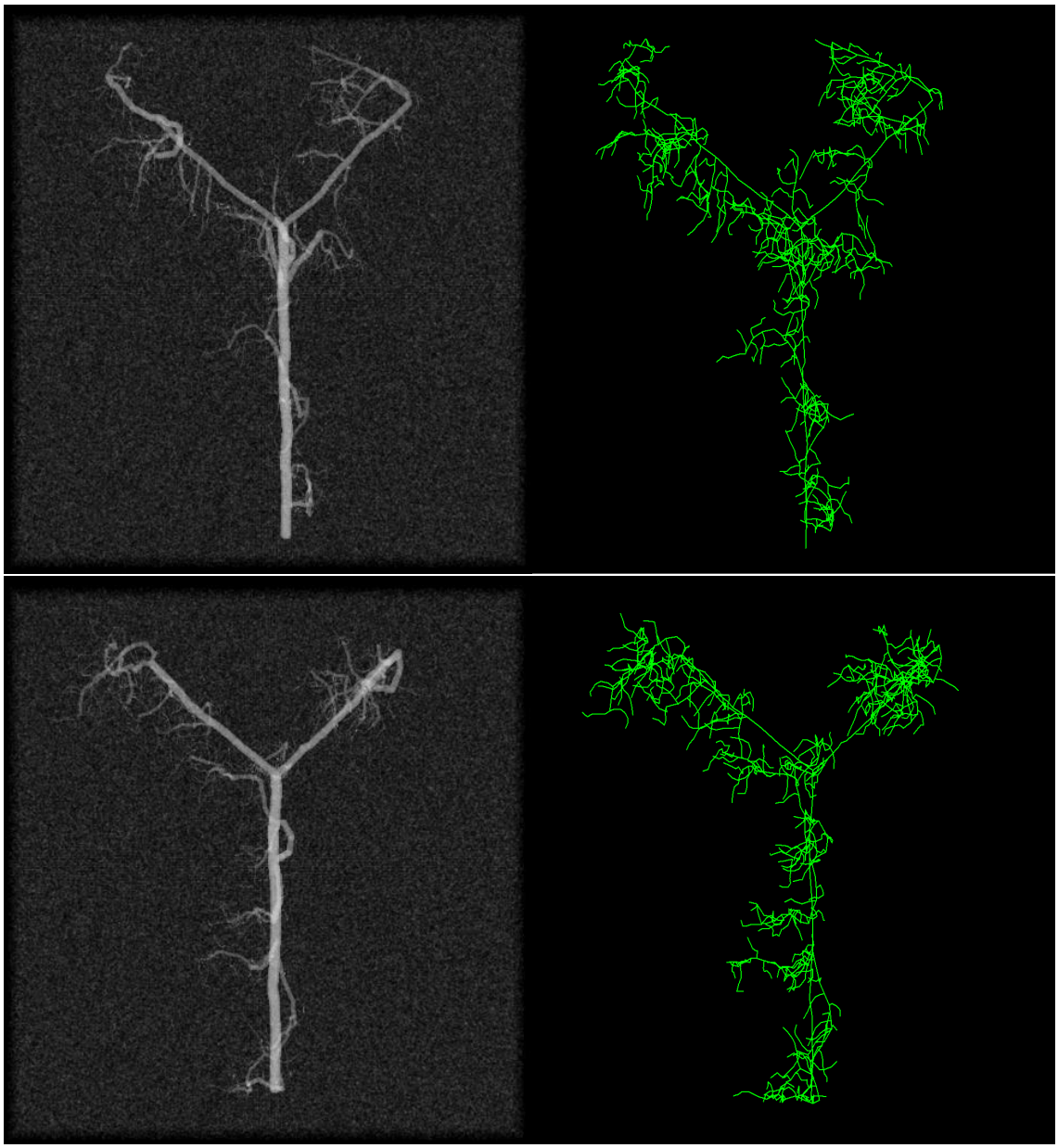


Fig. 58. Synthetic vasculatures generated from the model-based generator. Left column: Original data. Right column: Traced result. The standard deviation of the Gaussian noise was 0.020. Intensity profile ranged from 50 to 100.

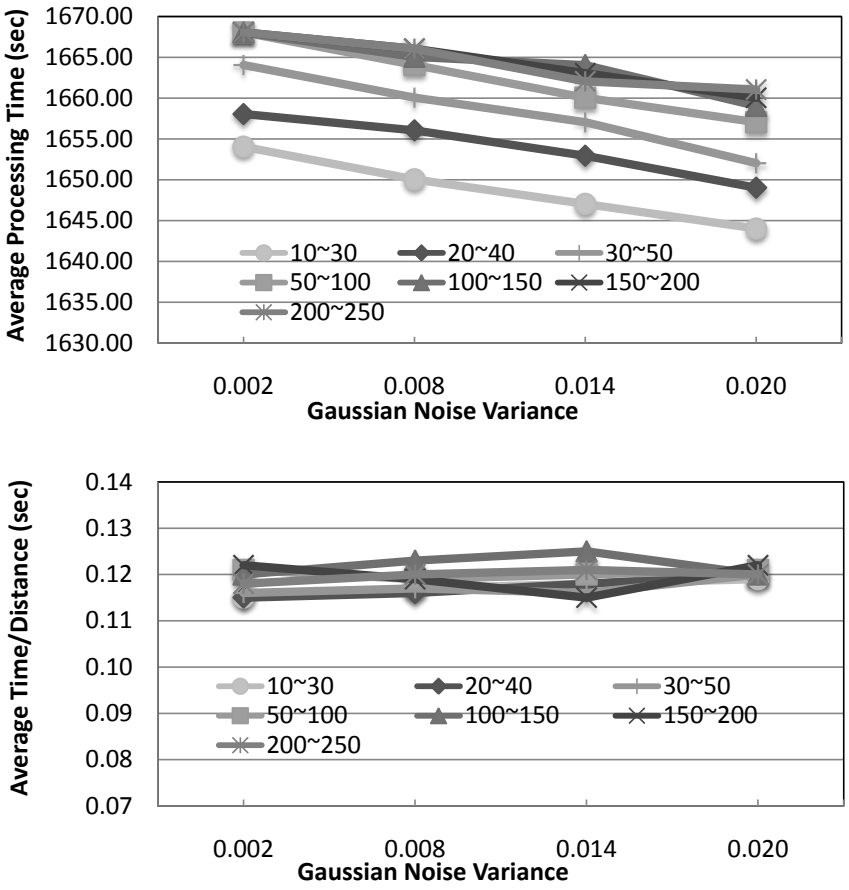


Fig. 59. Means of five synthetic neurovascular trees measures for processing time (top) and average time/distance (bottom). Each series represents the fiber's different intensity profile (10∼30, 20∼40, 30∼50, 50∼100, 100∼150, 150∼200, and 200∼250).

than 95.1% of all the tracing points are within 2 voxels of their closest medial axis point except the lowest contrast data. However, except the lowest one, the other data contain intensity profiles from 50 to 250. These results show that my tracing method with MIP processing can be used for a wide variety of medical volume data.

E. Whole mouse brain traced result

Using my 3D tracing algorithm, I conducted tracing experiment using the whole mouse brain data obtained by KESM. This whole mouse brain data were subsampled (1/32) and the tested data set was 375×375×290. Fig. 61 shows close-up of the traced result of a whole mouse brain. During subsampling, many thin microvasculatures could disappeared or become blurred. However, as we can see, in the subsampled data (the left column in Fig. 61), there are still many vasculatures to be traced even though they look unclear due to subsampling.

The tracing result (the right column in Fig. 61) shows that my tracing algorithm can trace these vasculatures. Note that I did not do any preprocessing regarding noise elimination for KESM data, which means that my algorithm can be used for tracing KESM data without any noise elimination process. The tracing of these thin and blurred vasculatures gave us the tracing result of the whole mouse brain data as shown in Fig. 62. The left column in Fig. 62 shows the iso-surface visualization of the whole mouse brain and the right column shows the traced result of the whole mouse brain data after subsampling.

F. Conclusion

In this chapter, I have shown how the diameter, length, and number of vessel elements were selected as parameters for modeling vasculatures. Using these parameters, the

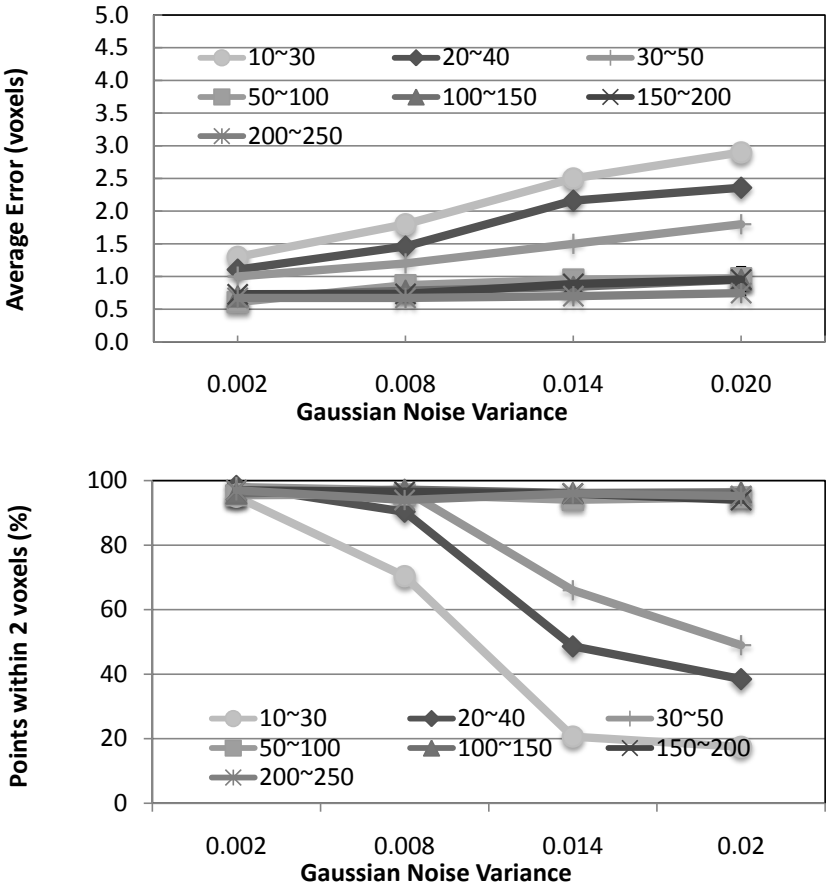


Fig. 60. Means of five synthetic neurovascular trees measures for average error (top) and percent of points within 2 voxels of their closest medial axis point (bottom). Each series represents the fiber's different intensity profile (10∼30, 20∼40, 30∼50, 50∼100, 100∼150, 150∼200, and 200∼250).

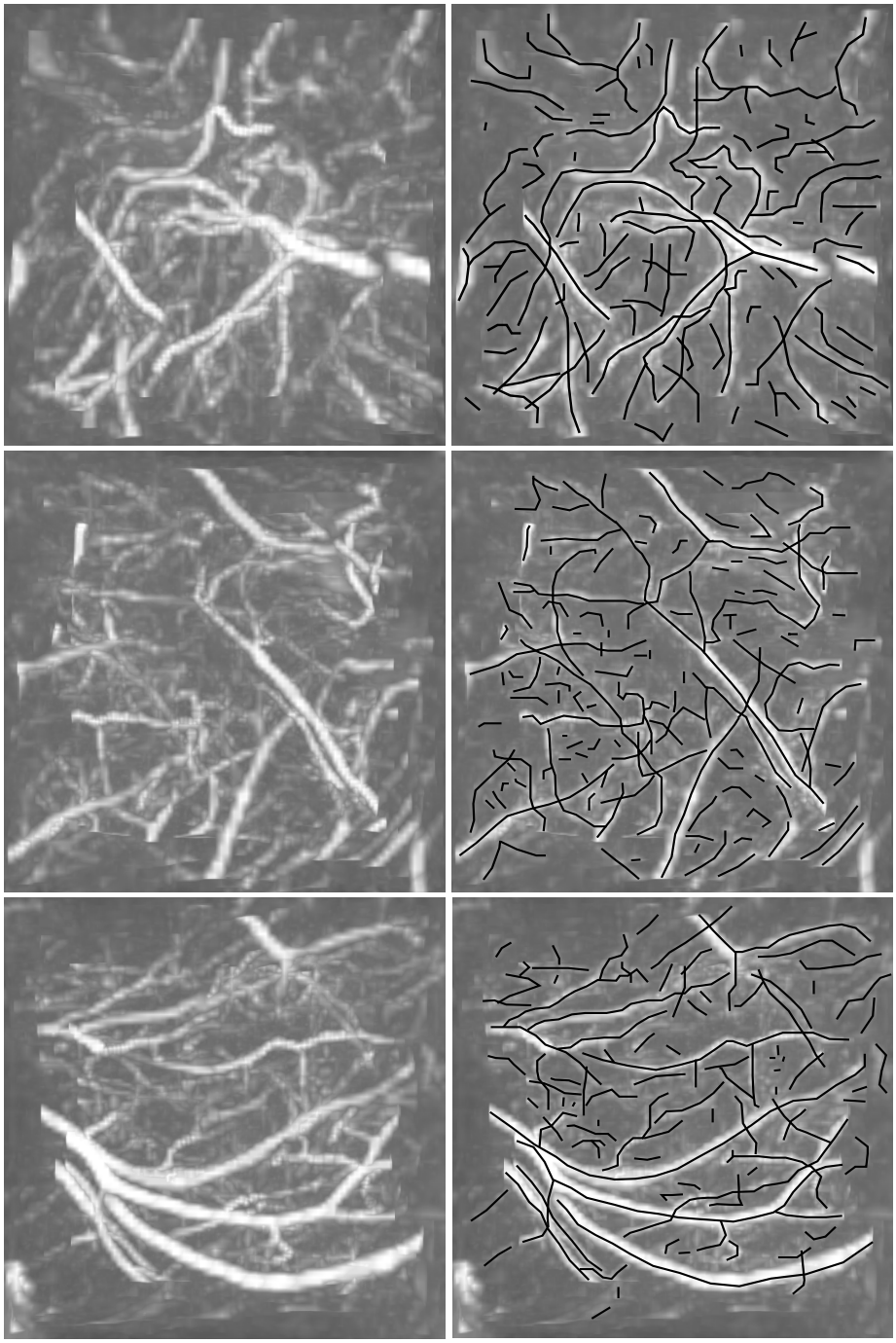


Fig. 61. Close-up of whole mouse brain traced result. Left: Iso-surface visualization. Right: Traced result.

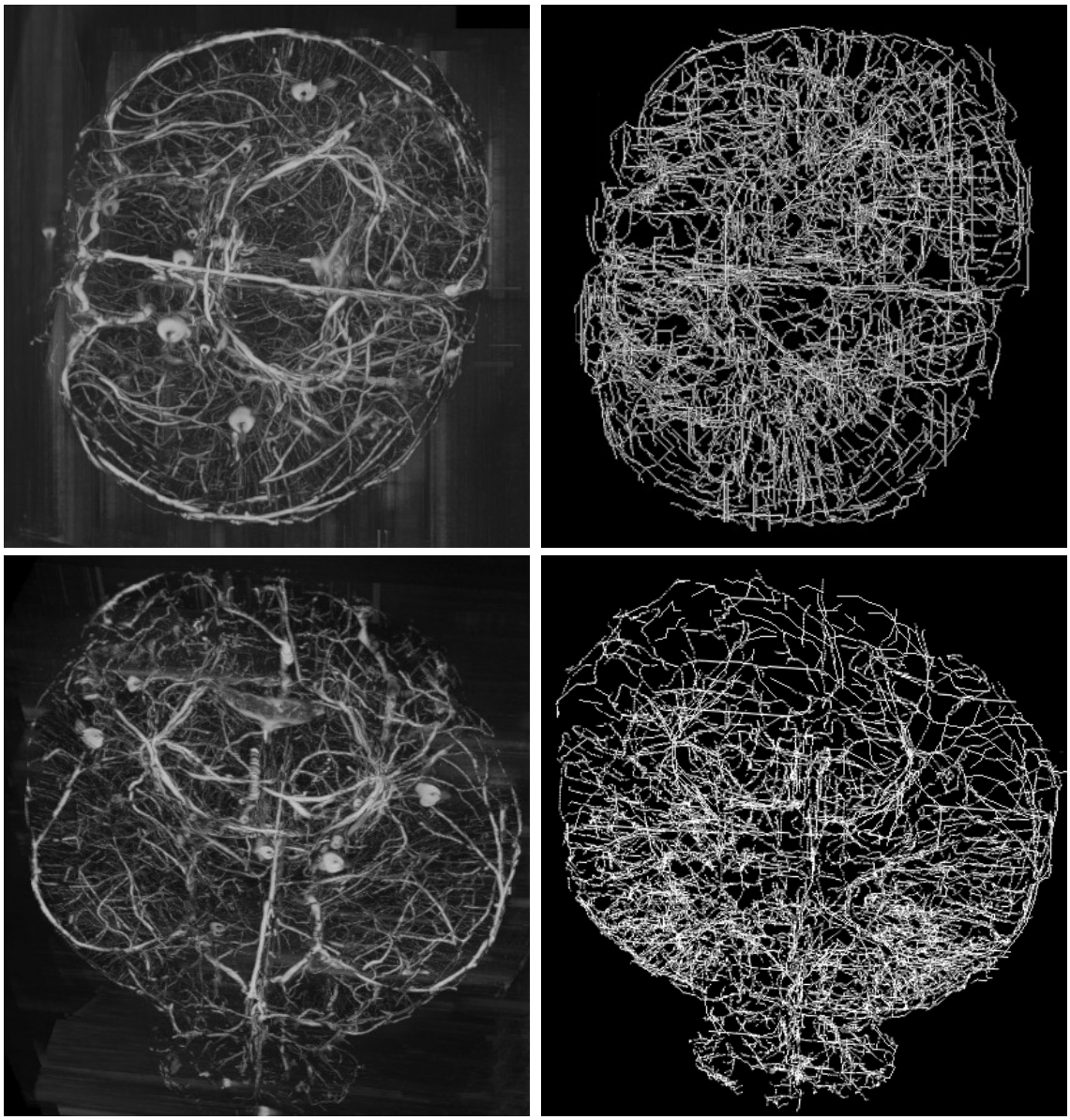


Fig. 62. Tracing result of a whole mouse brain. Left: Iso-surface visualization. Right: Traced result.

model-based generator created the synthetic vascular trees and compared the trees' statistic to the ground truth for validation. The comparison showed that there is a strong correspondence between the synthetic vascular trees and the ground truth regarding the parameter's statistics. The tracing results of the synthetic vascular trees, using my tracing method, were compared to the centerline information from the model-based generator and were found to be similar. This model-based generator can be used in neurovascular experiments to validate tracing methods.

CHAPTER VII

DISCUSSION AND CONCLUSION

A. Contributions

As compared to the related work in the literature, the main contributions of my method include: (1) Real time processing: My trace has $O(k)$ processing time (i.e., it is linear), where k is the fiber length. (2) Noise robustness: The algorithm is resistant to noise common in biomedical images. (3) Branch handling: The algorithm can solve branch handling issues such as ignored branching and incorrect backing-up (Fig. 7(a)(c)). (4) Smoothness: The use of CTTS improves smoothness for medial axis tracing (Fig. 7(b)). (5) Completeness: The algorithm can trace more connected fibers than the previous methods, starting from a single seed point. (6) Fast 3D tracing: My 3D tracing is based on local maximum intensity projection (MIP) within a MIP cube. This method can significantly reduce the search space. The reduction of the search space leads to fast tracing. (7) 3D tracing framework: The local MIP method provides a framework which can use any existing 2D tracing algorithms for tracing 3D data. Such reusability is a major contribution. (8) Synthetic and human-labeled validation: The synthetic validation method can cover most types of neurovascular network. Human-labeled validation can give a ground truth for tracing real data. (9) Thorough model-based validation: A large-scale validation method was introduced.

B. Open issues

In this dissertation, I have shown that my tracing algorithm can facilitate the improvement of existing tracing methods for 2D and 3D data. I also showed that a model-based generator can validate my tracing algorithms for realistic vascular trees

in the human brain. However, the following open issues remain: (1) is it efficient for us to use the automatic selection of seed points for full tracing, (2) can we keep all the benefits of 2D tracing algorithm through MIP processing, and (3) can the tracing algorithm also work well on the lowest contrast data? In this section, I will discuss these open issues and briefly introduce some ideas to begin addressing these issues.

1. **Seed point detection**: For the full tracing of neurovasculature, we have to have seed points as many as the number of separate vessels because one seed point cannot trace two separate vasculatures. However, the volume density of vasculature in human brain is not over 4 % [18]. It implies that manual selection of seed points may be more efficient than the automatic selection due to the complex processing involved in automatic selection [50, 51, 89]. However, as the size of data increases, it is hard to manually find seed points, and more erroneous seed points will occur. Thus, computational studies on enhancing the existing automatic selections of seed point will provide us with more efficient tracing performance.

2. **Loss of features through MIP**: Due to the dimension reduction of MIP processing, some features of 2D tracing methods could be lost. For example, CTTS and branch handling do not work well. However, I reduced the tracing step size and used a momentum operator to compensate for the loss of these CTTS features. Thus, I could approximate the medial axis of 3D vasculatures even though the tracing method needs additional processing time for the decreased step size and the momentum calculation. Regarding branch handling, it is hard to manage all branch cases on the projected planes. Therefore, on branch point, only one candidate direction is selected on each plane using predefined criteria and the other candidates are ignored. The ignored candidates

can be traced from the other seed points. Thus, to improve on branch handling through MIP, more specific branch cases should be investigated.

3. **Tracing on the lowest contrast data**: From the synthetic data set with various noise levels and contrast, I showed that the lowest contrast data (whose intensity profiles are from 10 to 50) gave either early termination of tracing or higher errors than high contrast data. Even though the human eye cannot easily differentiate vasculature in the lowest contrast data, some images have this contrast due to uneven illumination, noise, and lighting variation [90–92]. Currently, my solution is to consider a local threshold to find vasculature instead of a global threshold. However, if I increase the local threshold, early termination can occur. If I decrease the local threshold, the number of false positive will increase. Therefore, more thorough investigation on the lowest contrast data is needed.

C. Future work

Based on the open issues discussed in the previous section, I plan to conduct the following experiments as future work: (1) I need to investigate more intensively the open issues using the suggested ideas described in the previous section. (2) Vasculature of animals other than humans should be traced to see how general the tracing method is. I expect my algorithm to be applicable to different forms of volume data, such as horse, person, and other articulated creatures. (3) Neuron tracing with dendrite detection will be considered after enhancing my tracing methods. My MW method can already trace some dendrites. However, I need to further investigate my tracing method for 3D dendrite tracing, which is much more complex than 2D dendrite tracing. (4) The model-based generator should be combined with various

noise levels and contrast for more rigorous validation. Even though the synthetic data has validated my tracing algorithm with various noise levels and contrast, the realistic synthetic data extracted from the generator could give deeper insights regarding tracing algorithms. (5) Currently, my tracing algorithm does not scale anisotropic voxels. This ability is important because many current data acquisition methods do not yield isotropic data. I will test how well my tracing algorithm can work with anisotropic data and check whether there is some solution to fix bias due to this anisotropic property. (6) Even though I tested my tracing algorithm using synthetic data with Gaussian noise, noise in KESM can have different distribution from the Gaussian noise distribution. Therefore, I need more thorough tracing test using more data obtained by KESM in order to check whether my tracing algorithm can work well with KESM data without noise removal. From the whole mouse brain tracing result in chapter VI, my tracing algorithm worked well with subsampled KESM data without any preprocessing for noise elimination. However, the subsampling could remove some noise in KESM. Therefore, I need to check the original KESM data for the validation under noise specific to KESM. (7) In terms of application in medicine, my tracing algorithm will allow to measure changes in the vascular structure in conditions like embolic stroke. Morris et al. showed a reduction in cerebral microvessel diameter after 1 to 4 hours of embolic middle cerebral artery (MCA) occlusion in rat using quantitative laser scanning confocal microscopy [93]. They also showed a significant reduction (mean 10$\sim$12%) in vessel diameter in the ipsilateral cortex when compared to the homologous region in the contralateral hemisphere. Therefore, if I gather the vessel diameter information using my tracing algorithm and store the information, this statistical information can be helpful for analyzing the changes due to a stroke. I expect that the microvascular networks in ischemia related diseases can also be traced for analysis.

D. Expectation of my research

The goal of this dissertation was to develop a tracing algorithm for obtaining neurovascular network information. I introduced and validated the tracing algorithm with efficient and effective primitives constructed from geometry and image processing techniques for 2D and 3D data: (1) Moving Window (MW) for the selection of the next trace step's direction, (2) Cubic Tangential Trace Spline (CTTS) for the interpolation of tracing, (3) local Maximum Intensity Projection (MIP) for extending 2D tracing method to 3D tracing method, (4) synthetic data set combined with various noise levels and low to high image contrast, and (5) model-based generator for validating large-scale synthetic data.

I showed that in the tracing result, (1) MW can reduce the complexity of the tracing process, (2) MW also serves an enhancement for handling of branch points, (3) the Gaussian filter in MW makes the tracing method robust to noise, (4) CTTS provides a fast interpolation method for tracing medial axis, (5) MIP can use any existing 2D tracing methods for 3D tracing, and (6) MIP also provides a significant reduction of the search space.

For validating my tracing algorithm, I conducted experiments demonstrating that my method had linear computational processing time in terms of fiber length and could deal with noise types representative of medical data (MR, CT, and ultrasound). Thus, my tracing algorithm can facilitate enhanced analysis of medical data. With the model-based generator, I showed that (1) the synthetic vascular trees made by the generator had convincing similarity to the ground truth in terms of carefully selected parameters (the diameter, length, and number of vessel elements) and (2) my tracing result of the synthetic vascular trees was comparable to the centerline provided by the generator. Thus, we can think that these parameters have strong relevance to

the realistic generation of the synthetic vascular trees and my tracing method can be utilized for obtaining accurate neurovascular network information from extensive data volumes (the goal of my dissertation) with strong validation with the generator. In the future, I expect that (1) my tracing algorithm can provide neuroscientists and neuroengineers with a framework for standardizing the extension from 2D tracing methods to 3D tracing methods by using MIP and (2) the model-based generator can lay the groundwork for realistic modeling and validation of microvascular networks in human brain by providing realistic neurovascular trees.

E. Summary

In this chapter, I have summarized the contributions of my research. I have discussed several open issues (automatic seed point detection, features maintenance through MIP, and the handling of lowest contrast data for tracing) with alternative solutions outlined. I also have discussed future directions regarding the research in this dissertation. All future work can be thought of as generalization of my tracing method. I have given a conclusion describing the goal of my dissertation and the expectation of my research.

REFERENCES

[1] S. G. Diamond, T. J. Huppert, V. Kolehmainen, M. A. Franceschini, J. P. Kaipio, S. R. Arridge, and D. A. Boas, "Dynamic physiological modeling for functional difuse optical tomography," *Neuroimage*, vol. 30, pp. 88–101, 2006.

[2] B.V. Zlokovic, "Neurovascular mechanisms of Alzheimer's neurodegeneration," *Trends in Neurosciences*, vol. 28, no. 4, pp. 202–208, 2005.

[3] M. R. Metea and E. A. Newman, "Glial cells both dilate and constrict blood vessels: a mechanism of neurovascular coupling," *Journal of Neuroscience*, vol. 26, pp. 2862–2870, 2006.

[4] R. B. Panerai, "System identification of human cerebral blood flow regulatory mechanisms," *Cardiovascular Engineering International Journal*, vol. 4, pp. 59–71, 2004.

[5] B. Rosengarten, O. Huwendiek, and M. Kaps, "Neurovascular coupling and cerebral autoregulation can be described in terms of a control system," *Ultrasound Medical Biology*, vol. 27, pp. 189–193, 2001.

[6] D. Itoh, S. Aoki, K. Maruyama, Y. Masutani, H. Mori, T. Masumoto, O. Abe, N. Hayashi, T. Okubo, and K. Ohtomo, "Corticospinal tracts by diffusion tensor tractography in patients with arteriovenous malformations," *Journal of Computer Assisted Tomography*, vol. 30, pp. 618–623, 2006.

[7] R. Cassius, Z. Joseph, S. Sam, H. Ricardo, D. Pushpa, and P. Mark, "The accessory middle cerebral artery," *Operative Neurosurgery Supplement*, vol. 63, pp. 10–14, 2008.

[8] D. Mayerich, J. Kwon, Y. Choe, L. Abbott, and J. Keyser, "Constructing high-resolution microvascular models," *3rd Microscopic Image Analysis with Applications in Biology Workshop*, 2008.

[9] D. Mayerich, L. Abbott, and B. H. McCormick, "Knife-edge scanning microscopy for imaging and reconstruction of three-dimensional anatomical structures of the mouse brain," *Journal of Microscopy*, pp. 134–143, Jul. 2008.

[10] B. H. McCormick, B. Busse, Z. Melek, and J. Keyser, "Polymerization strategy for the compression, segmentation, and modeling of volumetric data.," Tech. Rep. 2002-12-1, Department of Computer Science, TAMU, College Station, TX, 2002.

[11] B. H. McCormick, B. Busse, P. Doddapaneni, Z. Melek, and J. Keyser, "Compression, segmentation, and modeling of filamentary volumetric data," *ACM Symposium on Solid Modeling*, vol. 4, pp. 9–11, 2004.

[12] W. Denk and H. Horstmann, "Serial block-face scanning electron microscopy to reconstruct three-dimensional tissue nanostructure," *PLoS Biology*, vol. 2, pp. e329, 2004.

[13] B. H. McCormick, "Development of the brain tissue scanner.," Tech. Rep. 2002-03-18, Brain Networks Laboratory, Department of Computer Science, TAMU, College Station, TX, 2002.

[14] Allen Institute for Brain Science, "http://www.brain-map.org/," Allen Brain Atlas, Accessed Aug. 2007.

[15] A. MacKenzie-Graham, E. S. Jones, D. W. Shattuck, I. D. Dinov, M. Bota, and

A. W. Toga, "The informatics of a c57bl/6j mouse brain atlas," *Neuroinformatics*, vol. 1, pp. 397–410, 2003.

[16] S. Mikula, I. Trotts, J. M. Stone, and E. G. Jones, "Internet-enabled high-resolution brain mapping and virtual microscopy," *Neuroimage*, vol. 35, pp. 9–15, 2007.

[17] Z. Melek, D. Mayerich, C. Yuksel, and J. Keyser, "Visualization of fibrous and thread-like data," *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, pp. 1165–1172, 2006.

[18] F. Cassot, F. Lauwers, C. Fouard, S. Prohaska, and V. Lauwers-Cances, "A novel three-dimensional computer-assisted method for a quantitative study of mcirovascular networks of the human cerebral cortex," *Microcirculation*, vol. 13, pp. 15–32, 2006.

[19] A. Ott, M. Breteler, F. Harskamp, J. J. Claus, T. Cammen, D. Grobbee, and A. Hofman, "Prevalence of Alzheimer's disease and vascular dementia: association with education. the Rotterdam study," *British Medical Journal*, vol. 310, pp. 970–973, 1995.

[20] V. Hachinski, C. Iadecola, R. C. Petersen, M. M. Breteler, D. L. Nyenhuis, S. E. Black, W. J. Powers, C. DeCarli, J. G. Merino, R. N. Kalaria, H. V. Vinters, D. M. Holtzman, G. A. Rosenberg, A. Wallin, M. Dichgans, J. R. Marler, and G. G. Leblanc, "National institute of neurological disorders and stroke. Canadian stroke network vascular cognitive impairment harmonization standards," *Stroke*, vol. 37, pp. 2220–2241, 2006.

[21] S. A. Mayer, N. C. Brun, K. Begtrup, J. Broderick, S. Davis, M. N. Diringer, B. E. Skolnick, and T. Steiner, "Recombinant activated factor vii for acute

intracerebral hemorrhage," *The New England Journal of Medicine*, vol. 352, pp. 777–785, 2005.

[22] Jr R . Freitas, "Nanotechnology, nanomedicine and nanosurgery," *International Journal of Surgery*, vol. 3, pp. 243–246, 2005.

[23] E. J. Birks, P. D. Tansley, J. Hardy, R. S. George, C. T. Bowles, M. Burke, N. R. Banner, A. Khaghani, and M. H. Yacoub, "Left ventricular assist device and drug therapy for the reversal of heart failure," *The New England Journal of Medicine*, vol. 355, pp. 1873–1884, 2006.

[24] E. J. Bekkers and C. A. Taylor, "Multiscale vascular surface model generation from medical imaging data using hierarchical features," *IEEE Transactions on Medical Imaging*, vol. 27, pp. 331–341, 2008.

[25] S. Chaudhuri, S. Chatterjee, N. Katz, M. Nelson, and M. Goldbaum, "Detection of blood vessels in retinal images using two-dimensional matched filters," *IEEE Trans. Med. Imag.*, vol. 8, pp. 263–269, 1989.

[26] A. R. Cohen, B. Roysam, and J. N. Turner, "Automated tracing and volume measurements of neurons from 3-D confocal fluorescence microscopy data," *J. Microscopy*, vol. 173, pp. 103–114, 1994.

[27] M. Sussman, P. Smereka, and S. Osher, "A level set approach for computing solutions to incompressible two-phase flow," *J. Comp. Physics*, vol. 114, pp. 146–159, September 1994.

[28] M. Descoteaux, L. Collins, and K. Siddiqi, "Geometric flows for segmenting vasculature in MRI: Theory and validation," *Medical Image Computing and Computer Assisted Intervention*, pp. 500–507, 2004.

[29] C. L. Tsai, C. V. Stewart, H. L. Tanenbaum, and B. Roysam, "Model-based method for improving the accuracy and repeatability of estimating vascular bifurcations and crossovers from retinal fundus images," *IEEE Trans. Inform. Technol. Biomed.*, vol. 8, 2004.

[30] O. Friman, M. Hindennach, and H. O. Peitgen, "Template-based multiple hypotheses tracking of small vessels," *IEEE International Symposium on Biomedical Imaging*, pp. 1047–1050, may 2008.

[31] S. Worz and K. Rohr, "Segmentation and quantification of human vessels using a 3-D cylindrical intensity model," *IEEE Trans. Imag. Proc.*, vol. 16, pp. 1994–2004, 2007.

[32] Y. Sato, S. Nakajima, N. Shiraga, H. Atsumi, S. Yoshida, T. Koller, G. Gerig, and R. Kikinis, "Three-dimensional multi-scale line filter for segmentation and visualization of curvilinear structures in medical images," *Med. Image Anal.*, vol. 2, pp. 143–168, 1998.

[33] O. Wink, W. J. Niessen, and M. A. Viergever, "Multiscale vessel tracking," *IEEE Trans. Med. Imag.*, vol. 23, pp. 130–133, 2004.

[34] R. Manniesing, B. K. Velthuis, M. S. van Leeuwen, I. C. van der Schaaf, P. J. van Laar, and W. J. Niessen, "Level set based cerebral vasculature segmentation and diameter quantification in ct angiography," *Med. Image Anal.*, vol. 10, pp. 200–214, 2006.

[35] W. Niessen, "Model-based image segmentation for image-guided interventions," in *Image-Guided Interventions*. New York: Springer, 2008.

[36] H. Li and A. Yezzi, "Vessels as 4d curves: Global minimal 4d paths to extract

3d tubular surfaces," *IEEE Conference on CVPR Proc.*, vol. 26, pp. 1213–1223, Sept. 2006.

[37] M. Holtzman-Gazit, R. Kimmel, N. Peled, and D. Goldsher, "Segmentation of thin structures in volumetric medical images," *IEEE Trans. Imag. Proc.*, vol. 15, pp. 354–363, Feb. 2006.

[38] A. Huang, G. M. Nielson, A. Razdan, G. E. Farin, D. P.Baluch, and D. G. Capco, "Thin structure segmentation and visualization in three-dimensional biomedical images: A shape-based approach," *IEEE Transactions On Visualization and Computer Graphics*, vol. 12, pp. 93–102, 2006.

[39] B. W. Hong, E. Prados, S. Soatto, and L. Vese, "Shape representation based on integral kernels: Application to image matching and segmentation," *The 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 833–840, 2006.

[40] S. Pizer, D. Fritsch, P. Yushkevich, V. Johnson, and E. Chaney, "Segmentation, registration, and measurement of shape variation via image object shape," *IEEE Trans. Med. Imag.*, vol. 18, pp. 851–865, 1999.

[41] R. Polli and G. Valli, "An algorithm for real-time vessel enhancement and detection," *Comput. Math. Programs Biomed.*, vol. 52, pp. 1–22, 1997.

[42] W. K. Pratt, "Image improvement," in *Digital Image Processing*. Hoboken, New Jersey: Wiley, 1978.

[43] M. E. Leventon, W. E. L. Grimson, and O. Faugeras, "Statistical shape influence in geodesic active contours," *IEEE International Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 1316–1323, 2000.

[44] S. Jehan-Besson, A. Herbulot, M. Barlaud, and G. Aubert, "Shape gradient for image and video segmentation," in *Mathematical Models in Computer Vision: The Handbook*. Heidelberg, Germany: Springer, 2005.

[45] J. Yang, L. H. Staib, and J. S. Duncan, "Neighbor-constrained segmentation with level set based 3D deformable models," *IEEE Trans. Med. Imag.*, vol. 23, pp. 940–948, 2004.

[46] A. Tsai, W. Wells, C. Tempany, E. Grimson, and A. Willsky, "Mutual information in coupled multi-shape model for medical image segmentation," *Medical Image Analysis*, vol. 8, pp. 429–445, Dec. 2004.

[47] S. Osher and N. Paragios, "Geometric level set methods in imaging, vision and graphics," *Heidelberg, Germany: Springer*, 2003.

[48] D. Adalsteinsson and J. A. Sethian, "A fast level set method for propagating interfaces," *J. Comp. Physics*, vol. 118, no. 2, pp. 269–277, 1994.

[49] N. Paragios and R. Deriche, "Geodesic active contours and level sets for the detection and tracking of moving objects," *IEEE Trans. Pattern Aanalysis and Machine Intelligence*, vol. 22, pp. 266–280, 2000.

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

[51] A. Can, H. Shen, J. N. Turner, H. L. Tanenbaum, and B. Roysam, "Rapid automated tracing and feature extraction from retinal fundus images using direct

exploratory algorithms," *IEEE Trans. Inform. Technol. Biomed.*, vol. 3, pp. 125–138, June 1999.

[52] J. L. Coatrieux, M. Garreau, R. Collorec, and C. Roux, "Computer vision approaches for the three-dimensional reconstruction: Review and prospects," *Critical Rev. Biomed. Eng.*, vol. 22, pp. 1–38, 1994.

[53] K. Haris, S.N. Efstratiadis, N. Maglaveras, C. Pappas, J. Gourassas, and G. Louridas, "Model-based morphological segmentation and labeling of coronary angiograms," *IEEE Trans. Med. Imag.*, vol. 18, pp. 1003–1015, October 1999.

[54] H. Shen, B. Roysam, C. Stewart, J. Turner, and H. Tanenbaum, "Optimal scheduling of tracing computations for real-time vascular landmark extraction from retinal fundus images," *IEEE Trans. Inform. Technol. Biomed.*, vol. 5, pp. 77–91, March 2001.

[55] F. Mourgues, F. Devernay, G. Malandain, and E. Coste, "3D+t modeling of coronary artery tree from standard non simultaneous angiograms," in *Medical Image Computing and Computer-Assisted Intervention. Proceedings 4th International Conference*, vol. 2208. Utrecht, Netherlands, Lecture Notes in Computer Science Series, Oct. 2001.

[56] D. Conte, P. Foggia, C. Sansone, and M. Vento, "Graph matching applications in pattern recognition and image processing," *Proc. of International Conference on Image Processing 2003*, pp. 14–17, 2003.

[57] D. Conte, P. Foggia, C. Sansone, and M. Vento, "How and why pattern recognition and computer vision applications use graphs," in *Applied Graph Theory*

in Computer Vision and Pattern Recognition, vol. 52, pp. 85–135. Heidelberg, Germany: Springer, 2007.

[58] K. Sun, N. Sang, and T. Zhang, "Marked point process for vascular tree extraction on angiogram," in *Energy Minimization Methods in Computer Vision and Pattern Recognition*, vol. 4679, pp. 467–478. Heidelberg, Germany: Springer, 2007.

[59] A. Frangi, W. Niessen, K. L. Vincken, and M. A. Viergever, "Multiscale vessel enhancement filtering," *Medical Image Computing and Computer Assisted Intervention*, vol. 1496, pp. 130–137, 1998.

[60] J. F. Carrillo, M. H. Hoyos, E. E. Davila, and M. Orkisz, "Recursive tracking of vascular tree axes in 3D medical images," *International Journal of Computer Assisted Radiology and Surgery*, vol. 1, pp. 331–339, 2007.

[61] S. R. Aylward, E. Bullitt, S. M. Pizer, and D. Eberly, "Intensity ridge and widths for tubular object segmentation and registration," *IEEE Workshop on Mathematical Methods in Biomedical Image Analysis*, vol. 56, pp. 131–138, 1996.

[62] S. R. Aylward and E. Bullitt, "Initialization, noise, singularities, and scale in height ridge traversal for tubular object centerline extraction," *IEEE Trans. Med. Imaging*, vol. 21, pp. 61–75, 2002.

[63] C. Kirbas and F. Quek, "A review of vessel extraction techniques and algorithms," *ACM Computing Surveys*, vol. 36, pp. 81–121, 2004.

[64] A. F. Frangi, W. J. Niessen, R. M. Hoogeveen, T. V. Walsum, and M. A. Viergever, "Model-based quantification of 3-d magnetic resonance angiographic images," *IEEE Trans. Med. Imag.*, vol. 18, pp. 946–956, 1999.

[65] M. Mandegar, Y. C. Fung, W. Huang, C. V. Remillard, L. J. Rubin, and J. X. Yuan, "Cellular and molecular mechanisms of pulmonary vascular remodeling: role in the development of pulmonary hypertension," *Microvasc Res.*, vol. 68, pp. 75–103, 2004.

[66] E. R. Weibel, "Morphometry of the human lung," *Anesthesiology*, vol. 26, pp. 367, May 1963.

[67] S. Singhal, R. Henderson, K. Horsfield, K. Harding, and G. Cumming, "Morphometry of the human pulmonary arterial tree," *Circ. Res.*, vol. 33, pp. 190–197, 1973.

[68] W. Huang, R. T. Yen, M. McLaurine, and G. Bledsoe, "Morphometry of the human pulmonary vasculature," *J. Appl. Physiol.*, vol. 81, pp. 2123–2133, 1996.

[69] G. A. Ascoli and J. L. Krichmar, "L-neuron: A modeling tool for the efficient generation and parsimonious description of dendritic morphology," *Neurocomputing*, vol. 32, pp. 1003–1011, 2000.

[70] G. A. Ascoli, "Variability of dendritic structure among and within morphological classes and experimental protocols," *The Society for Neuroscience 33rd Annual Meeting*, pp. 203–210, 2003.

[71] J. Pelt, V. Ooyen, and H. Uylings, "Modeling dendritic geometry and the development of nerve connections," *Computational Neuroscience: Realistic modeling for experimentalist*, vol. 10, pp. 179–208, 2001.

[72] R. S. Nowakowski, N. L. Hayes, and M. D. Egger, "Competitive interactions during dendritic growth: a simple stochastic growth algorithm," *Brain Res*, vol. 576, pp. 152–156, 1992.

[73] T. A. Hely, B. Graham, and A. Ooyen, "A computational model of dendrite elongation and branching based on map2 phosphorylation," *J. Theor. Bio.*, vol. 210, pp. 375–384, 2000.

[74] A. Ooyen, B. P. Graham, and G. J. A. Ramakers, "Competition for tubulin between growing neurites during development," *Neurocomputing*, vol. 38, pp. 73–78, 2001.

[75] K. L. Karau, R. C. Molthen, A. Dhyani, S. T. Haworth, C. C. Hanger, D. L. Roerig, R. H. Johnson, and C. A. Dawson, "Pulmonary arterial morphometry from microfocal x-ray computed tomography," *Am. J. Physiol.*, vol. 281, pp. 2747–2756, 2001.

[76] B. Minnich, H. Bartel, and A. Lametschwandtner, "Quantitative microvascular corrosion casting by 2d and 3d-morphometry," *Ital. J. Anat. Embryol.*, vol. 106, pp. 213–220, 2001.

[77] J. P. Eberhard, A. Wanner, and G. Wittum, "Neugen: a tool for the generation of realistic morphology of cortical neurons and neural networks in 3d," *Neurocomputing*, vol. 70, pp. 327–342, 2006.

[78] R. C. Cannon, D. A. Turner, G. K. Pyapali, and H. V. Wheal, "An on-line archive of reconstructed hippocampal neurons," *Journal of Neuroscience*, vol. 84, pp. 49–54, 1998.

[79] Y. Tamori, "Theory of dendritic morphology," *Phys. Rev.*, vol. 48, pp. 3124–3129, 1993.

[80] F. Lauwers, F. Cassot, V. Lauwers-Cances, P. Puwanarajah, and H. Duvernoy,

"Morphometry of the human cerebral cortex microcirculation: general characteristics and space-related profiles," *Neuroimage*, vol. 39, pp. 936–948, 2007.

[81] J. E. Bresenham, "Algorithm for computer control of a digital plotter," *IBM Systems Journal*, vol. 4, pp. 25–30, 1965.

[82] Y. Sun, "Automated identification of vessel contours in coronary arteriograms by an adaptive tracking algorithm," *IEEE Trans. Med. Imag.*, vol. 8, pp. 78–88, March 1989.

[83] C. Boldak, Y. Rolland, and C. Toumoulin, "An improved model-based vessel tracking algorithm with application to computed tomography angiography," *JBBE*, pp. 41–64, 2003.

[84] Y. Zhang, X. Zhou, J. Lu, J. Lichtman, D. Adjeroh, and S. TC Wong, "3-d axon structure extraction and analysis in confocal fluorescence microscopy images," *Neural computation*, vol. 20, pp. 1899–1927, 2008.

[85] D. A. Tata and B. J. Anderson, "A new method for the investigation of capillary structure," *J. Neurosci. Methods*, vol. 113, pp. 199–206, 2002.

[86] S. R. Kayar, P. G. Archer, A. J. Lechner, and N. Banchero, "The closest-individual method in the analysis of the distribution of capillaries," *Microvasc. Res.*, vol. 24, pp. 326–341, 1982.

[87] G. S. Kassab, D. H. Lin, and Y. C. Fung, "Morphometry of pig coronary venous system," *Am. J. Physiol.*, vol. 267, pp. 2100–2113, 1994.

[88] T. Mitsa and Q. Jiang, "Modeling 3-d lung morphogenesis using fractal geometries and a boundary constraint," *Engineering in Medicine and Biology Society*, vol. 11, pp. 568–569, 1993.

[89] J. Shan, H. D. Cheng, and Y. Wang, "A novel automatic seed point selection algorithm for breast ultrasound images," *International Conference on Pattern Recognition*, vol. 7, pp. 1–4, 2008.

[90] L.P. Dzung, X. Chenyang, and L.P. Jerry, "A survey of current methods in medical image segmentation," *Annual Review of Biomedical Engineering*, vol. 2, pp. 315–338, 2000.

[91] I. Bollmann, P. Quinn, and M. Vela, "Automated particle analysis: Calcareous microfossils," in *Image Analysis, Sediments and Poleoenvironments*. Dordrecht, The Netherlands: Springer, 2003.

[92] F.J. W-M Leong, M. Brady, and J.OD. McGee, "Correction of uneven illumination (vignetting) in digital microscopy images," *J. Clin. Pathol*, vol. 56, pp. 619–621, 2003.

[93] D. C. Morris, K. Davies, Z. Zhang, and M. Chopp, "Measurement of cerebral microvessel diameters after embolic stroke in rat using quantitative laser scanning confocal microscopy," *Brain Research*, vol. 876, pp. 31–36, September 2000.

VITA

Dong Hyeop Han was born in Busan, South Korea. In 1992, he entered Yonsei University in Seoul, South Korea. He received the degree of Bachelor of Science and Master of Science in 1996 and 1998. During the following six years he was employed as an associate engineer at Samsung Electronics Company. He entered the graduate school at The University of Wisconsin at Madison in September, 2004 and transferred to the graduate school at Texas A&M University in September, 2006, and graduated in August 2009 with a Ph.D.

Permanent Address: Samsung Electronics Printing Division, Maetan 3Dong, Paldal Gu, Suwon City, Kyungki Do, South Korea. Zip:442-742

This dissertation was typed by Dong Hyeop Han.