ACQUISITION AND MINING OF THE WHOLE MOUSE BRAIN MICROSTRUCTURE

A Dissertation

by

JAE-ROCK KWON

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 Engineering

ACQUISITION AND MINING OF THE WHOLE MOUSE BRAIN MICROSTRUCTURE

A Dissertation

by

JAE-ROCK KWON

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:

Co-Chairs of Committee,	Yoonsuck Choe
	Ricardo Gutierrez-Osuna
Committee Members,	Frank Shipman
	Jyotsna Vaid
Head of Department,	Valerie E. Taylor

August 2009

Major Subject: Computer Engineering

ABSTRACT

Acquisition and Mining of the Whole Mouse Brain Microstructure. (August 2009)

Jae-rock Kwon, B.S, Hanyang University, Seoul, Korea;

M.S., Hanyang University, Seoul, Korea

Co–Chairs of Advisory Committee: Dr. Yoonsuck Choe
Dr. Ricardo Gutierrez-Osuna

Charting out the complete brain microstructure of a mammalian species is a grand challenge. Recent advances in serial sectioning microscopy such as the Knife-Edge Scanning Microscopy (KESM), a high-throughput and high-resolution physical sectioning technique, have the potential to finally address this challenge. Nevertheless, there still are several obstacles remaining to be overcome. First, many of these serial sectioning microscopy methods are still experimental and are not fully automated. Second, even when the full raw data have been obtained, morphological reconstruction, visualization/editing, statistics gathering, connectivity inference, and network analysis remain tough problems due to the unprecedented amounts of data.

I designed a general data acquisition and analysis framework to overcome these challenges with a focus on data from the C57BL/6 mouse brain. Since there has been no such complete microstructure data from any mammalian species, the sheer amount of data can overwhelm researchers. To address the problems, I constructed a general software framework for automated data acquisition and computational analysis of the KESM data, and conducted two scientific case studies to discuss how the mouse brain microstructure from the KESM can be utilized.

I expect the data, tools, and studies resulting from this dissertation research to greatly contribute to computational neuroanatomy and computational neuroscience.

To my parents, Gijeong Kwon and Jongsik Yun, and my wife, Hyunsoo

ACKNOWLEDGMENTS

I would like to thank Dr. Yoonsuck Choe and Dr. Ricardo Gutierrez-Osuna for their support and thoughtful supervision throughout the wide range of projects that we have conducted together. Dr. Yoonsuck Choe and I have been working together since I joined the Brain Networks Laboratory (BNL) and his patient and thoughtful advice led me to the point where I am now. I particularly want to thank him for letting me join the laboratory and have this wonderful opportunity. Dr. Ricardo Gutierrez-Osuna has also given me invaluable advice regarding my research.

I would also like to thank my committee: Dr. Frank Shipman for his feedback; and Dr. Jyotsna Vaid for her advice to widen my point of view in my research topic.

I would also like to thank David Mayerich, who has been a key player in the Brain Networks Laboratory and the most knowledgeable expert of the Knife-Edge Scanning Microscopy (KESM) instrument. He is the best officemate, colleague, and a thoughtful peer mentor with whom I have had the pleasure of working.

Finally, I would like to thank the late Dr. Bruce H. McCormick whose vision and unstoppable drive for research have been greatly influential to me. All achievements in BNL would not have been possible without his dream. He will be greatly missed by all of us. This research was funded in part by NIH/National Institute for Neurological Disorders and Stroke grant #R01-NS54252 and by NSF grant CCF-0220047.

TABLE OF CONTENTS

CHAPTER Page

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

FIGURE Page

FIGURE Page

FIGURE Page

FIGURE Page

CHAPTER I

INTRODUCTION

The brain may be the final frontier of science. Scientists may have much more knowledge about the universe than their own brain as John Horgan, an American science journalist, pointed out [1][2]. As brain science made great advances during the past few decades, we have come to understand the brain much more than any other time in human history. However, in spite of numerous significant findings, there still are many mysteries of the brain that have yet to be solved.

In order to understand the function of the remarkably complex networks of neuronal cells and vasculature, their elements and inter-connections should be investigated [3]. However, the sheer structural complexity of the brain prevents us from studying brain function through examining its large-scale structure. For example, the human brain has approximately 10^{11} neurons and their possible connections are estimated to be 10^{14} [4]. Besides, the total length of the vasculature in the human brain is estimated as hundreds of miles and its surface area is more than 100 square feet [5].

It is a tremendous challenge to understand brain function because elements and interconnections of the brain networks should be fully investigated, given the sheer number of elements and their connections [3].

A. Motivation

It is important to map out the anatomical structure underlying the topological connectivity of microstructure of the brain, because the functional connectivity between

---

This thesis follows the style of *Transactions on Neural Networks*.

the units in the brain heavily depends on the topology of the underlying their connections [3]. However, previous microscopic studies have been limited by their small sample size, and this prevented a global interpretation of the relationship between functional principles and topological connectivity in the brain [6]. The data size can become extremely large when the anatomical data is acquired in sub-micron level from the whole mouse brain. The data generated can easily exceed tens of terabytes. Moreover, the objects in the data have highly complex structures and they are also densely packed. For these reasons, the investigation of the microanatomy of the brain has been limited either to a small part of the brain such as parts of the cerebral cortex, or in thick sections when the whole brain was investigated [7][8]. However, in order to find the information processing mechanism in the brain, it is inevitable to scrutinize the microstructure of the brain at the whole-brain level, and inspect the relationship between them. According to models of large-scale cortical networks, basic functions reside in scattered cortical areas, whereas complex and integrated functions are processed simultaneously in widespread areas [9]. Therefore, if it is possible, exploring the entire brain seems to be the best way, toward a more precise and complete understanding of the brain.

Even though recent advances in high resolution and high throughput imaging technologies have the potential to get microstructure of the brain of larger animals, acquiring the complete connection matrix of microstructure of a mammalian species yet remains a grand challenge.

B. Approach

The ultimate goal of brain research is not only to figure out cellular structure but also to relate function with structure. To figure out cellular level structure of the

brain, we need to have *high-resolution* microscopy technologies. At the same time, the microscopy techniques should be *high-throughput* because it is a prerequisite to large scale analysis. A major reason large-scale analysis of the structure is needed is that such structure gives us invaluable insights into the highly complex functional organization of the brain. In this context, there have been efforts to acquire, store, analyze, and visualize brain data, and much recent research has resulted in various data sets [10][11][12][7]. However only few microscopy technologies exist today that satisfy the conditions of *high-resolution* and *high-throughput* even though high resolution and high throughput imaging technologies have advanced in recent years. The main reason is that the two conditions are often hard to achieve at the same time. Most imaging techniques currently being used in image acquisition have limitations either in the speed of data acquisition or in the resulting resolution of the captured images.

Therefore, the complete anatomical connections between the network elements of the human brain are still mostly unexplored due to not just its massive volume but also for ethical reasons [13]. Given these circumstances, the mouse brain serves as a viable alternative, as a template for mammalian brains in neurobiological and behavioral research [14][15][16][6][17]. It is small in size so that researchers can possibly get a full set of brain sections; it is also relatively cheap and can be easily used without serious ethical controversies [13]; and finally it has a relatively smooth cerebral cortex [18], so that it can be easily imaged and explored. Despite many differences between the mouse brain and the human brain, the genes of both organs are 90% identical [19]. So, the mouse brain can be an invaluable tool to explore and investigate the complexity of the human brain.

Knife Edge Scanning Microscopy (KESM), a high-throughput and high-resolution three-dimensional imaging instrument, allows us to acquire microvascular and neu-

ronal microstructure data from whole mouse brains. However, many problems had existed in exploring the data sets.

To address these problems, I have developed new methods that can acquire anatomical microstructure data sets in large scale using KESM, reconstruct morphological structures from the data sets, and visualize raw and reconstructed data sets in efficient ways. Contributions of this dissertation are as follows.

- Automation of our data acquisition system using modified KESM techniques for larger data sets
- Image pre-processing techniques for reconstructing and visualizing large biological volumetric data sets
- Data representation for efficient storage, retrieval, and annotation
- Automation of structural feature extraction using a fast tracing algorithm adjusted for parallel processing
- Integrated tools to manage raw data sets, process data sets, reconstruct morphological structures from the raw data sets, and visualize both raw and reconstructed structures from data sets
- Two neuroscience case studies to discuss possibilities of utilization of the data sets and the software framework in testing neuroscience hypotheses

C. Outline of the Dissertation

The methods described in this dissertation are for imaging, processing, managing, and analyzing the whole mouse brain (on the order of several cm^3 at the sub-micron scale

(0.3 μm - 1.0 μm). Here, I present an integrated software framework to effectively acquire, manage, visualize, and analyze the mouse brain microstructure.

In this chapter, I described the motivation of my work and approaches that I used to address research challenges. In **Chapter II**, I discuss background information of my research including the brain microstructure, microscopy technologies, and the Knife-Edge Scanning Microscopy (KESM).

Chapter III, **IV**, and **V** describe the data processing pipeline of KESM. In particular, I focus on automation of the process with algorithms for data acquisition in **Chapter III**. In **Chapter IV**, data processing frameworks and relevant algorithms for the data sets are described. This includes an algorithm for fixing misaligned image columns and a method for removing image noises. **Chapter V** explains an efficient hierarchical data management method.

In **Chapter VI**, I briefly describe visualization methods. Also I will discuss a data representation method for the traced fibrous data of densely packed and interconnected networks.

Chapter VII is devoted to the statistical data analysis. Here, I focus on morphometry analysis for large amount of data sets.

In **Chapter VIII**, I explain an integrated software framework that helps us acquire, manipulate, store, manage, visualize, and analyze the data sets.

Two theoretical studies in neuroscience is discussed in **Chapter IX** to examine the possibility of utilizing of the data sets.

Chapter X summarizes this dissertation and discusses future directions of my research. This dissertation concludes with **Chapter XI**.

The main topic of my work is to provide an efficient software framework to acquire, process, manage, analyze, and visualize the data sets for KESM, a high-throughput and high-resolution three-dimensional imaging instrument. Understand-

ing microstructure in the brain is an important step in understanding its function. In order to get precise and high-resolution (sub-cellular level) three-dimensional structure of the brain, we need to have a high-throughput image acquisition system. Automation is a key issue in the high-throughput microscopy techniques. By providing an efficient software framework for organizing and analyzing the microstructures in the brain, such as neurons, their supporting cells, and microvessels, I expect to contribute to the furthering of our knowledge of the brain structure and function.

In the following, I will start by introducing general microscopy technologies. Next, I will describe high-throughput and high-resolution data acquisition technologies, with a focus on Knife Edge Scanning Microscopy (KESM).

CHAPTER II

BACKGROUND

Mapping out the precise interconnections of microstructure in the brain is crucial to understand brain function because the functional connectivity between units in the brain heavily depends on the morphology of the underlying neuronal connections [3]. The microstructure includes not only neurons and their supporting cells but also vasculature. In the brain, neurons are linked with endothelial cells of blood vessel walls through glia cells. The close functional interaction led to a concept of neurovascular unit [20][21].

The Brain Networks Lab (BNL), where I have been working, is conducting a survey of the three-dimensional microstructures across whole mammalian brains, specifically that of the C57BL/6J mouse. Three tissue stains are used: (1) Nissl stain allows us to plot the spatial distribution and morphology of all cell bodies; (2) Golgi stain is used to see the morphology of randomly selected cells (approximately 1% of neurons); and (3) India Ink stains vasculature.

A. Brain Microstructure

Blood flow in vasculature is typically well correlated with neural activity. Functional hyperemia, a localized blood influx correlated with neural activity levels, is a good example and it is the basis of functional magnetic resonance imaging (fMRI) where Blood Oxygen Level-Dependent(BOLD) signals are measured [22]. The blood flow in the brain can be considered as a proxy for neural activity and we can safely consider that the blood flow changes as activity of neurons changes [23]. Therefore, the brain microstructure, composed of neurons, microvessels, and their supporting cells [24], collectively termed the neurovascular unit (Fig. 1) should be investigated to figure

out brain function because their components dynamically interact and communicate with each other to regulate cerebral blood flow or synaptic activity [24][25][23][22][26].

1. Connectome and Connectomics

Connectomics, a new field in neuroscience, has emerged to find out anatomical structure of neurons and their inter-connections [27][28][29][30]. The goal of connectomics is to get the complete physical map of neural circuits (*connectome* [31][32]). As of now, *C. elegans* is the only species of which we have the connectome. The nematode is one of the simplest organisms with a nervous system. It has 302 neurons and 7,000 synapses and their connectivity patterns have been completely mapped out in the mid-80s [33].

Connectomics will be driven by new technologies. The Blue Brain Project [34], an ambitious attempt to reverse-engineer the mammalian brain, is an example of the large-scale modeling and simulation. As completion of the first phase, the project built a model of the basic functional unit of the brain, the neocortical column [35]. A main reason of large-scale analysis and modeling is that cognitive brain function should be understood by figuring out inter-regional pathways. Furthermore, in order to build a biologically accurate model of the brain, the first step is to inspect three dimensional morphology [34]. Thus, high-resolution and high-throughput microscopes are key prerequisites of revealing connectomes and recent microscopy technologies offer significant advances in mapping out the connectome at cellular resolution.

2. Blood Vessels

Brain microvasculature has been shown to play an important role in neurological disorders and neurodegenerative diseases including Alzheimer's Disease, Multiple Sclerosis, and Parkinson's Disease [20]. Capillaries, the basic component of the microvascular

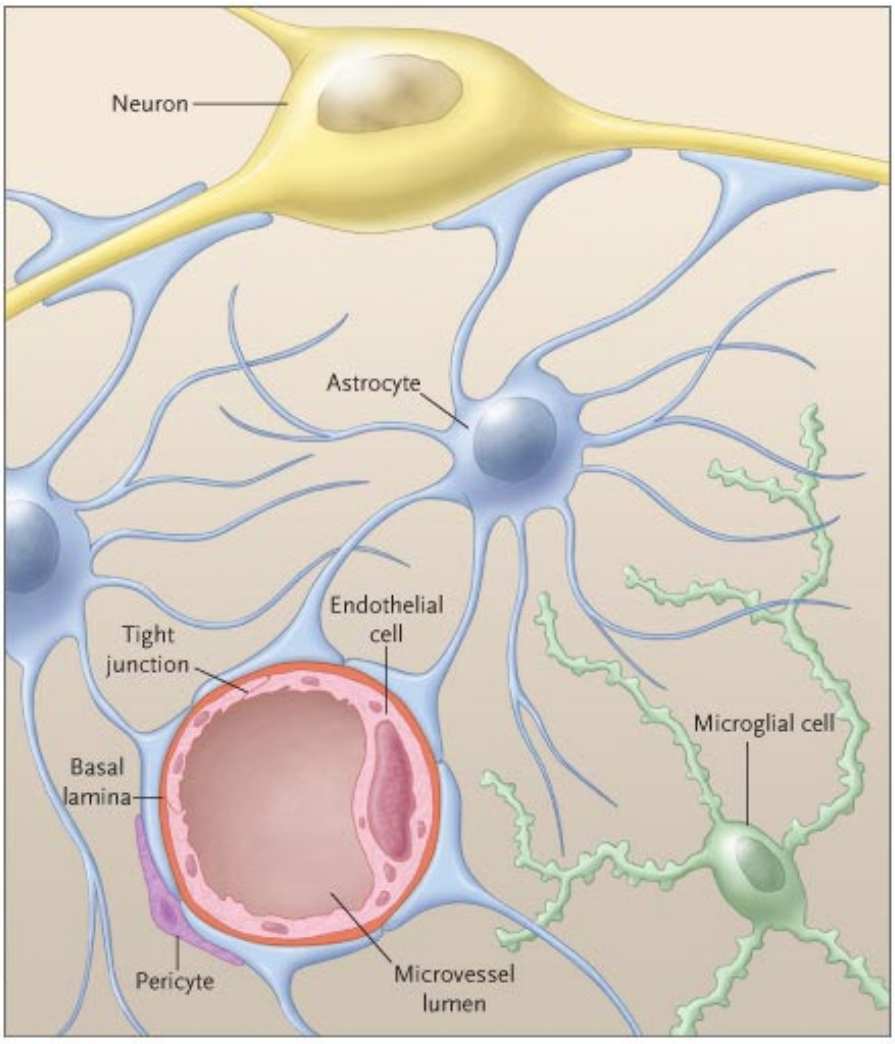


Fig. 1. The Neurovascular Unit. This unit consists of neurons, microvessels, and their supporting cells[24]. Astrocytes play an important role in communication between neurons and microvessels [25].

system, perform important nutritional functions and may also affect the neural response [22][36]. Thus, complete models of the vascular structure are important for understanding brain function and dysfunction. However, very little is known about the structure of microvascular networks. This is due both to their small size and to their extraordinary complexity. Microvascular networks have several properties that make them difficult to image and model. The capillaries are about $5\mu m$ in diameter, requiring high-resolution imaging for reconstruction. Their connected components are often stretched over several millimeters of tissue. Creating complete microvascular network data sets requires imaging entire organs at a sub-cellular resolution. Advanced imaging methods and segmentation techniques are required to cope with these large and complex data sets.

In order to examine and analyze the microstructure of the whole mouse brain tissue, first of all, a high-throughput and high-resolution data acquisition technique is needed. In this chapter, I provide a brief overview of various imaging methods to provide a proper context for the work I discuss in the remainder of the dissertation.

B. Microscopy Technologies

Confocal light microscopy [37] and multi-photon microscopy [38] are commonly used to acquire high-resolution images. The basic idea of the confocal light microscopy is to change the depth of focus (focal plane) and use a pinhole aperture to detect photons emitting from the specified depth. This is called *optical sectioning* in which actual sectioning does not take place. Each image can be considered as a thin slice of a tissue block. By stacking up these images, we can build a three-dimensional data set. However, there are limitations in such an optical sectioning method. First, the images can be acquired only on or near the tissue surface because back-scattered

light deteriorates the image contrast as scanning depth increases. Second, its scanning resolution is limited by diffraction. Thus, the main limitation factor is the resolution of the z direction (about 700 nm) [**?**]. Multi-photo microscopy improves the resolution and the imaging depth limitation but this is a optical sectioning technique that suffers from the z-axis resolution limitation. Slow imaging speed is also an issue for optical sectioning techniques such as confocal or multi-photon microscopy. The data rate is less than 8MB/s in [39] and 1 frames/s in [38].

These limits in optical microscopy can be solved by using electron microscopy (EM). EM uses a beam of electrons instead of light and has a much greater resolution than light microscopy. However, there are some limitations to EM as well. It requires the use of heavy metal stains, but these stains cannot penetrate the tissue deeply. This limitation prevents EM from being used for large tissue samples. Additionally EM imaging takes much longer than light microscopy. The long imaging time could be a practical limitation when it comes to dealing with massive volumes.

In order to overcome the scanning depth, serial physical sectioning techniques can be considered as an alternative since z-axis resolution depends on the tissue thickness (it can be up to about $30nm$ using a vibrating microtome). Also virtually no depth limit exists because it physically sections a tissue block.

Here are recent advances in physical sectioning microscopy.

- All-optical Histology [40]
- Array Tomography [41]
- Serial Block Face Scanning Electron Microscopy [42]
- Automatic Tape-collecting Lathe UltraMicrotome [43]
- Knife-Edge Scanning Microscopy (KESM) [44][45][46][36]

1. All-optical Histology

All-optical Histology (AOH) is an enhanced technique of multi-photon microscopy for removing the depth limitation (Fig. 2). Standard optical sectioning is used for imaging but, for deeper imaging, AOH ablates tissue sections using femtosecond laser pulses. The main advantage of this hybrid optical microscopy is that it overcomes the z axis limitation. However, the imaging speed is still slow due to the property of multi-photon microscopy.

2. Array Tomography

Array Tomography (AT) uses an ultramicrotome to serially section tissue embedded in acrylic resin (Fig. 3). A tissue specimen is cut into ribbons of ultrathin $(50 - 200nm)$ sections which are placed on a glass slide. The main advantage of this technique is that the resulting tissue array can be repeatedly washed, stained, and imaged which enables to use multiple fluorescent markers at a time. As an imaging method, array tomography uses electron microscopy. It again, suffers from slow imaging speed.

3. Serial Block-face Scanning Electron Microscopy

Serial Block-face Scanning Electron Microscopy (SBF-SEM) images the block surface after cutting off a tissue ribbon [42] at a resolution on the order of $10nm \times 10nm \times 50nm$ (Fig. 4). The main advantage of SBF-SEM is its high-resolution so that cellular and subcellular structure can be imaged. However, the imaging time is almost prohibitive when we need the high signal-to-noise ratio. For example, it can take up to about one year to scan a $200\mu m$ cube at the above resolution. The size of the volume can be problematic when we need to have large amount of data sets because the volume size is limited to several hundred μm in SBF-SEM. Given these conditions,

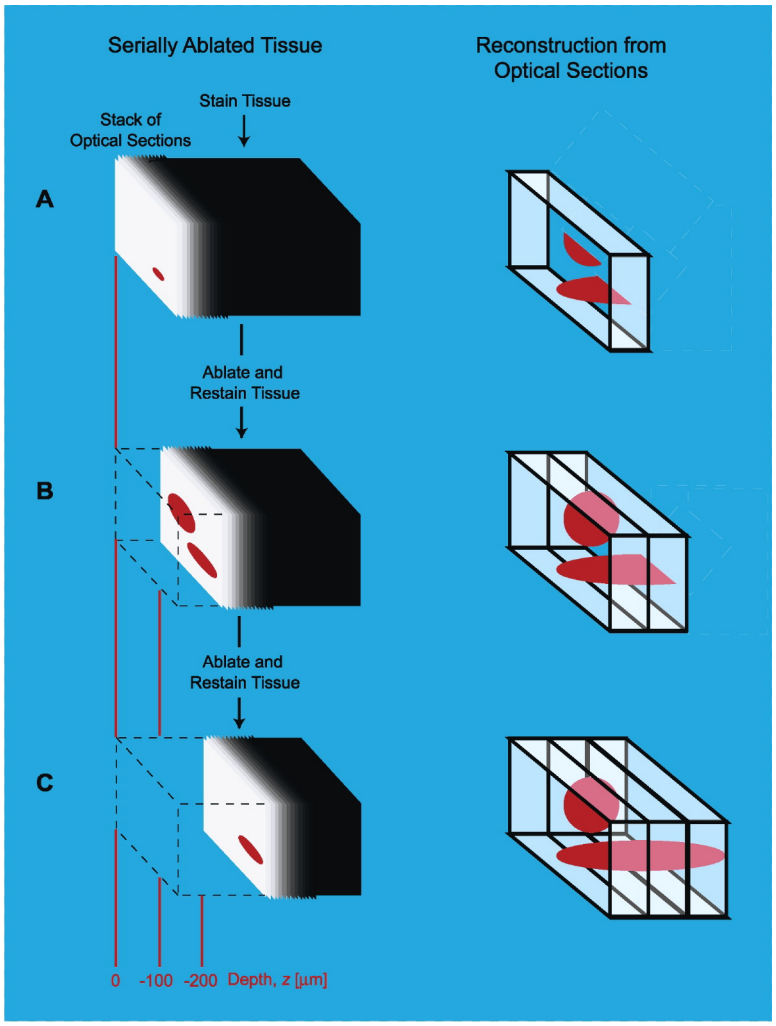


Fig. 2. All-optical Histology (AOH). It shows the iterative process by which issue is imaged and cut in AOH. (A) A tissue sample (left column) containing two fluorescently labeled structures is imaged by conventional two-photon laser scanning microscopy to collect optical sections through the ablated surface. Sections are collected until scattering of the incident light reduces the signal–to-noise ratio below a useful value; typically this occurs at 150 m in fixed tissue. Labeled features in the resulting stack of optical sections are digitally reconstructed (right column). (B) The top of the now-imaged region of the tissue is cut away with amplified ultrashort laser pulses to expose a new surface for imaging. The sample is again imaged down to a maximal depth, and the new optical sections are added to the previously stored stack. (C) The process of ablation and imaging is again repeated so that the structures of interest can be fully sectioned and reconstructed (Adapted from [40]).

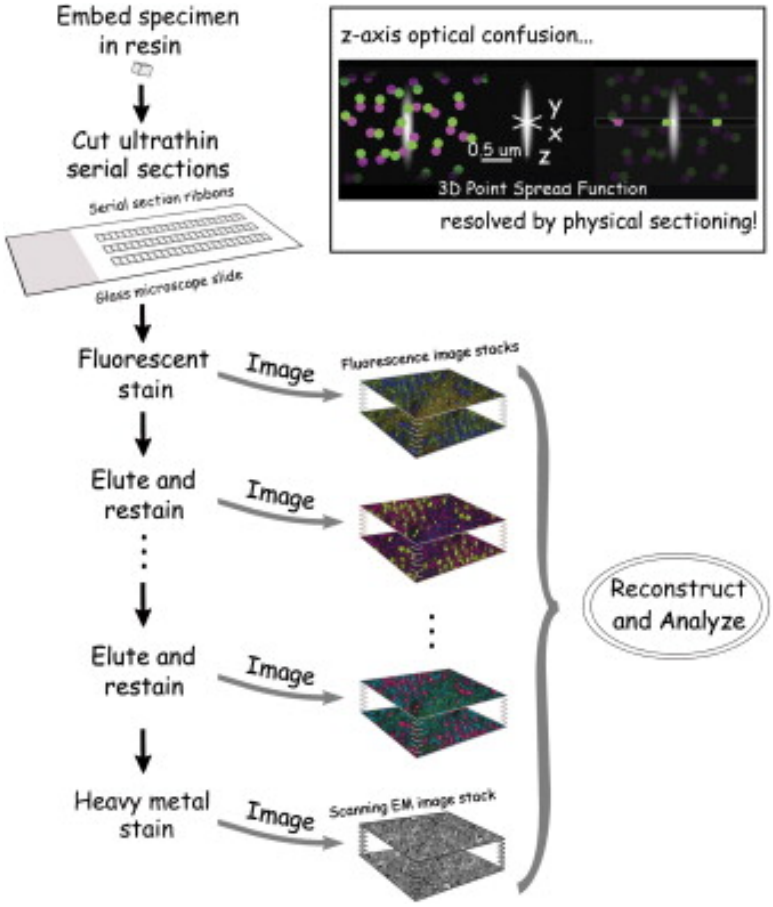


Fig. 3. Schematic Representation Of The Array Tomography Method. A tissue specimen is embedded in acrylic resin and cut into ribbons of serial ultrathin $(50 - 200nm)$ sections, which are then bonded to glass slides. The resulting array is labeled with fluorescent antibodies or other fluorescent stains and imaged to generate ultra-high-resolution volumetric images. The array can be repeatedly eluted, restained, and fluorescently imaged, and finally, it can also be stained with heavy metals and imaged under a scanning electron microscope. Insert illustrates the principle behind the axial resolution enhancement by array tomography. Optical microscopes have their poorest resolution along the optical axis, represented in the figure by z axis elongation of the 3D point spread function. Optical sectioning (left) yields an image that is severely degraded by confusion along the z axis, a problem avoided in array tomography by using ultrathin physical sectioning. (Adapted from [41]).

SBF-SEM is not a proper choice for large volumes of tissue due to its low image acquisition rate.

4. Automatic Tape-collecting Lathe Ultramicrotome

Automatic Tape-collecting Lathe Ultramicrotome (ATLUM) is one of the latest developments in serial sectioning microscopy [43] (Fig. 5). ATLUM sections a tissue block, collects a series of ribbons, and puts them on a long carbon-coated tape that is cut into a section library. As the name of the technology implies, imaging is a completely separate step. Images in the section library will be digitize only when needed.

5. Knife-Edge Scanning Microscopy (KESM)

Knife-Edge Scanning Microscopy (KESM) is a high-throughput optical technique, particularly for sectioning and imaging large tissue blocks at sub-micron level[44][45][46][36]. The operation principle of KESM is that cutting and imaging take place at the same time using a diamond knife. Thin sections are cut from the specimen and imaged using a high speed line scan camera. KESM is not constrained by the specimen thickness since physical tissue ribbons are sectioned from the specimen, like other serial sectioning techniques (Fig. 6).

Illumination is provided through the diamond knife using a high-intensity illuminator. Here, the diamond knife is used as both a microtome and an illumination device. The light is passed through the tissue ribbon as the tissue block is being sectioned (Fig. 7). KESM has lower resolution than EM since it is an optical technique, but it allows us to acquire larger image volumes of tissue in a much faster rate. It is capable of scanning a complete mouse brain (about $310mm^3$) at $300nm$ sampling resolution within 100 hours when scanning in full production mode.

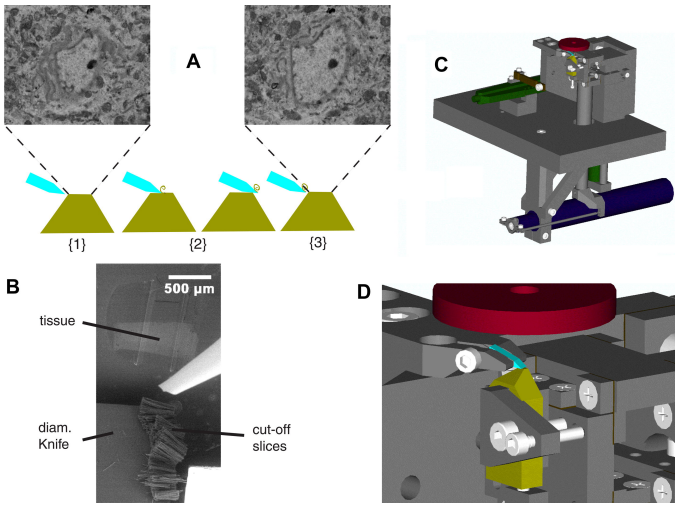
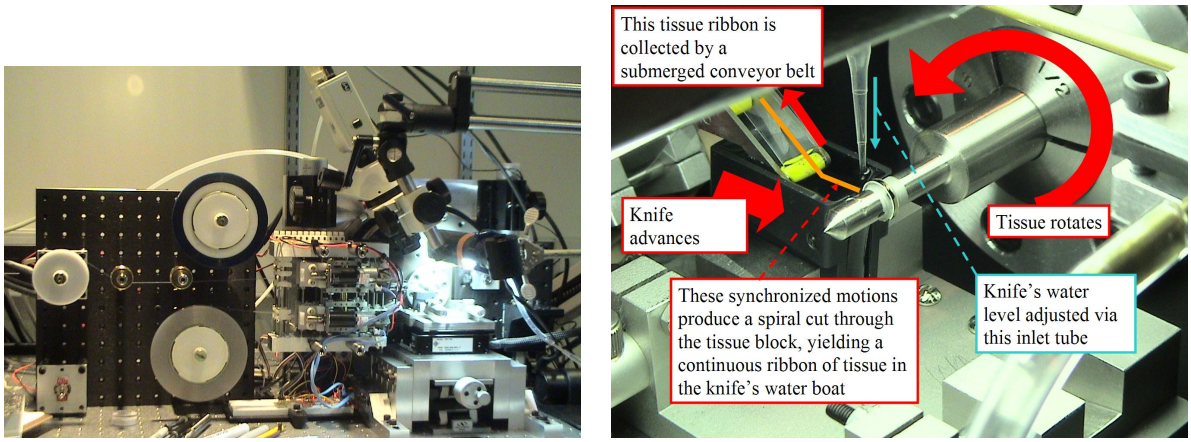


Fig. 4. Serial Block-face Scanning Electron Microscopy (A) Principle of SBFSEM operation: (1) a SEM image is taken of the surface of the plastic-embedded tissue preparation (amber trapezoid). (2) Then with a diamond knife (blue) an ultrathin slice is cut off the top of the block. (3) After retraction of the knife, the next picture is taken. The pictures shown are from an actual stack (cerebellar cortex) but are not successive slices; rather, they are spaced by five images (about $315nm$) to make the changes more apparent. (B) Usually cut-off slices pile up on the top of the knife. Protruding into the picture from the right is a puffer pipette, occasionally used to remove debris from the knife. (C and D) The mechanical design for the in-chamber microtome is shown in an overview (C) and a close-up of knife and sample (D) in renderings from the computer-aided design software. Most parts are nonmagnetic stainless steel (grey). A large-motion leveraged piezo actuator (green part on the left) drives the knife holder back and forth. The custom diamond knife (light blue) is clamped in a special holder. The sample (amber) advance is driven via a lever by a direct-current-motor-driven micrometer (dark blue). The retraction during the backwards knife motion is again piezo actuated (green cylinder in the lower right of [C]). Bearing springs are brown. The BSE detector (red) is depicted schematically above the sample. Not shown is the lateral positioning mechanism.. (Adapted from [42]).



(*a*) (*b*)

Fig. 5. Automatic Tape-collecting Lathe Ultramicrotome (ATLUM). (a) ATLUM prototype. (b) Overview of ATLUM operation. (Adapted from [47]).

I will describe more technical details of KESM in the following section.

6. Comparison of Physical Sectioning Microscopy

The resolution and the imaging speed of the methods that I described all differ. Any one of methods can play a complementary role to the other methods. In other words, each method has relative advantages and disadvantages to the other methods. However, as Table I indicates, KESM shows relatively high-resolution as well as high-throughput property.

C. Knife-Edge Scanning Microscopy (KESM)

KESM comprises four major subsystems: (1) precision positioning stage (Aerotech), (2) microscope/knife assembly (Micro Star Technology), (3) image capture system (Dalsa), and (4) cluster computer (Dell) (Fig. 6). The specimen, a whole mouse brain, is embedded in a plastic block and mounted atop a three-axis precision stage. A custom diamond knife, rigidly mounted to a massive granite bridge overhanging the

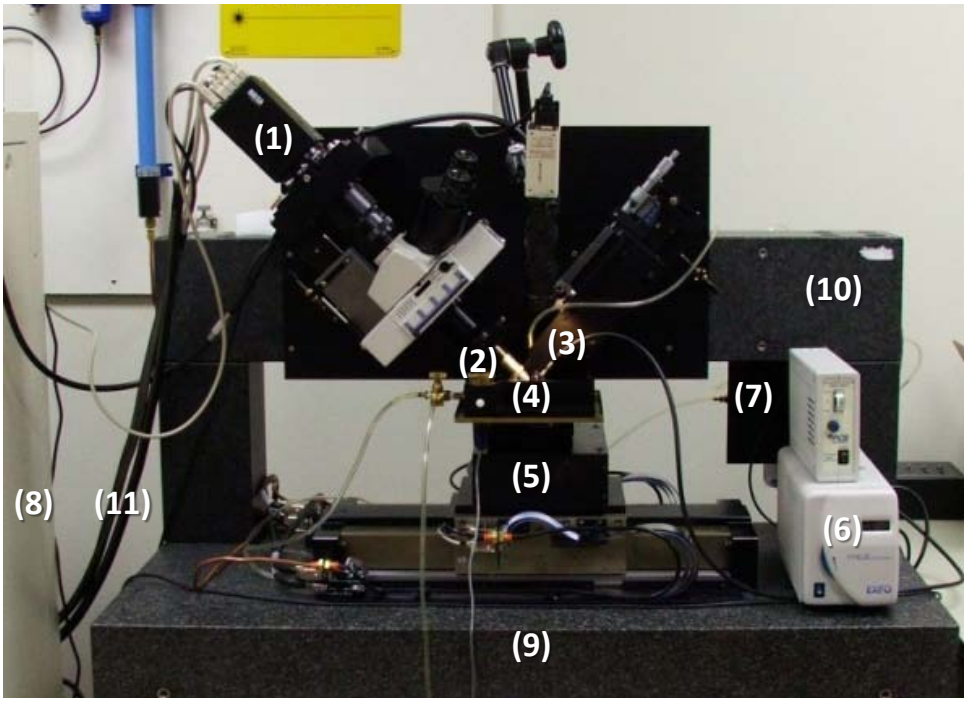
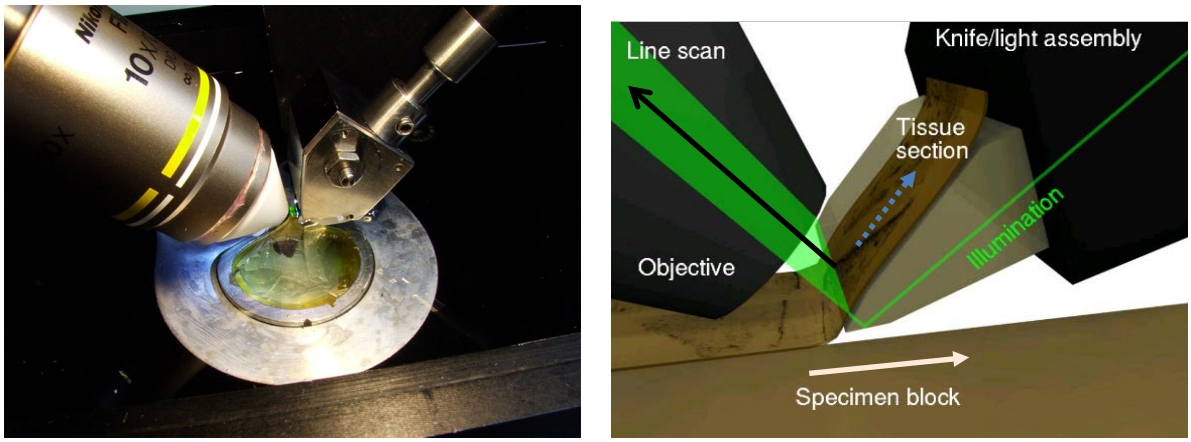


Fig. 6. The Knife-Edge Scanning Microscope (KESM). Major components of the KESM are marked: (1) high-speed line-scan camera, (2) microscope objective, (3) diamond knife assembly, (4) specimen tank (for water immersion imaging), (5) three-axis precision air-bearing stage, (6) microscope illuminator, (7) water pump for removal of sectioned tissue ribbon, (8) precision stage controller (side view), (9) granite base, (10) granite bridge, and (11) connection cable to PC server for stage control and image acquisition.



(*a*) (*b*)

Fig. 7. Tissue Sectioning And Imaging In KESM (a) A close-up of the parts 2, 3, and 4 in Fig. 6 is shown. To the left is the microscope objective, and to the right the diamond knife and light collimator. Submerged under water in the center is the plastic-embedded brain tissue held in a specimen ring. (b) The principal of operation of KESM is illustrated. The objective and the knife is held in place, while the specimen affixed on the positioning stage moves (white arrow with solid line) and gets scraped against the diamond knife, generating a thin section flowing over the knife. Line-scan imaging is done at the very tip of the knife where the distortion is minimal. Illumination if provided through the diamond knife (black arrow indicates the light path after reflecting the knife tip). Adapted from [36].

Table I. Comparison Of Physical Sectioning Microscopy

Method	Resol.(x,y)	Resol.(z)	Volume	Modality	Time
All-opt. hist.	$0.5\mu m$	$1\mu m$	$1cm^3$	FL*	$\sim$ 900hrs
Array Tomo.	$\sim 0.2\mu m$	$0.05 - 0.21\mu m$	$\sim 100^3\mu m^3$	FL*, EM**	N/A
SBF-SEM	$\sim 0.01\mu m$	$\sim 0.03\mu m$	$\sim 500^3\mu m^3$	EM**	N/A
ATLUM	$\sim 0.01\mu m$	$0.05\mu m$	$\sim 2.15^3\mu m$	EM**	N/A
KESM	$0.3 - 0.6\mu m$	$0.5 - 1\mu m$	$1cm^3$	BF***	$\sim$ 100hrs

FL: Fluorescence, EM**: Electron Microscopy, BF***: Bright field*

three-axis stage, cuts consecutive thin serial sections from the block. Unlike block face scanning, the KESM concurrently cuts and images (under water) the tissue ribbon as it advances over the leading edge of the diamond knife. A white light source illuminates the rear of the diamond knife, providing illumination at the leading edge of the diamond knife with a strip of intense illumination reflected from the beveled knife-edge, as illustrated in Fig. 7. Thus, the diamond knife performs two distinct functions: as an optical prism in the collimation system, and as the tool for physically cutting thin serial sections. The microscope objective, aligned perpendicular to the top facet of the knife, sees a tissue ribbon through the transmitted light. A high-sensitivity line-scan camera repeatedly samples the newly cut thin section at the knife-edge, prior to subsequent major deformation of the tissue ribbon after imaging. The imaged stripe is a $20\mu m$-wide band locate at the bevel at the very tip of the diamond knife, spanning the entire width of the knife. Finally, the digital video signal is passed through image acquisition boards and stored for subsequent analysis in a small dedicated computing server. The current server is a dual processor PC (3.2 GHz/2MB Cache, Xeon) with 6 GB of memory, built-in 1 TB storage, connected to an

archival RAID attachment. The process of sectioning and imaging is fully automated with minimal human intervention.

A quick calculation puts us in context, regarding the massiveness of the data that KESM can produce. Consider the acquisition of volume data representing a plastic-embedded mouse brain ($15mm$ Anterior-Posterior, $12mm$ Medial-Lateral, $6mm$ Dorsal-Ventral). A 40X objective has a field of view (knife width) of $0.625mm$. Sixteen strips (each $0.625mm$ wide by $15mm$ long) are cut for each z-axis section (like plowing a field). For a (z-axis) block height of $6mm$, 12,000 sections must be cut, each $0.5\mu m$ thick. The integrated tissue ribbon length ($15mm$/strip $\times$ 16 strips/section $\times$ 12,000 sections/mouse brain) is $2.9km$. The tissue ribbon is line-sampled at $300nm$ resolution, near the Nyquist rate for an ideal optical resolution of $(0.77)(532nm)=(0.80NA) = 512nm$. Based on this, the total data size (assuming one byte per voxel) comes out to 20 terabytes (TB) (at half the resolution in each dimension, it would be about 2.5 TB). The tissue ribbon can be sampled at 11 mm/s by line sampling at 44 kHz (180 MB/s), the camera maximum (Dalsa CT-F3-4096 pixels). Sampling the $2.9km$ tissue ribbon requires 265,000 s = 73 hours. Because mice brains are not cubical, stage return takes time, etc., we add 50% overhead, resulting in about 100 hours.

Figs. 8, 9, and 10 show typical data that can be obtained using the KESM [36]. Nissl staining dyes the RNA in the cytoplasm of all neurons and the DNA in cell bodies in all cells. However, the dendritic arbors and axons remain unstained. Thus, Nissl staining allows us to reconstruct the distribution of all cell bodies in the mouse brain, and in particular their distribution within the six layers of the cerebral cortex (see Fig. 8 and 9). Golgi staining, in contrast, reveals the entire structure of neurons, as it stains just 1% of the neurons in the tissue. Individual neurons can be seen clearly, permitting reconstruction (see Fig. 10). India ink enables high-contrast

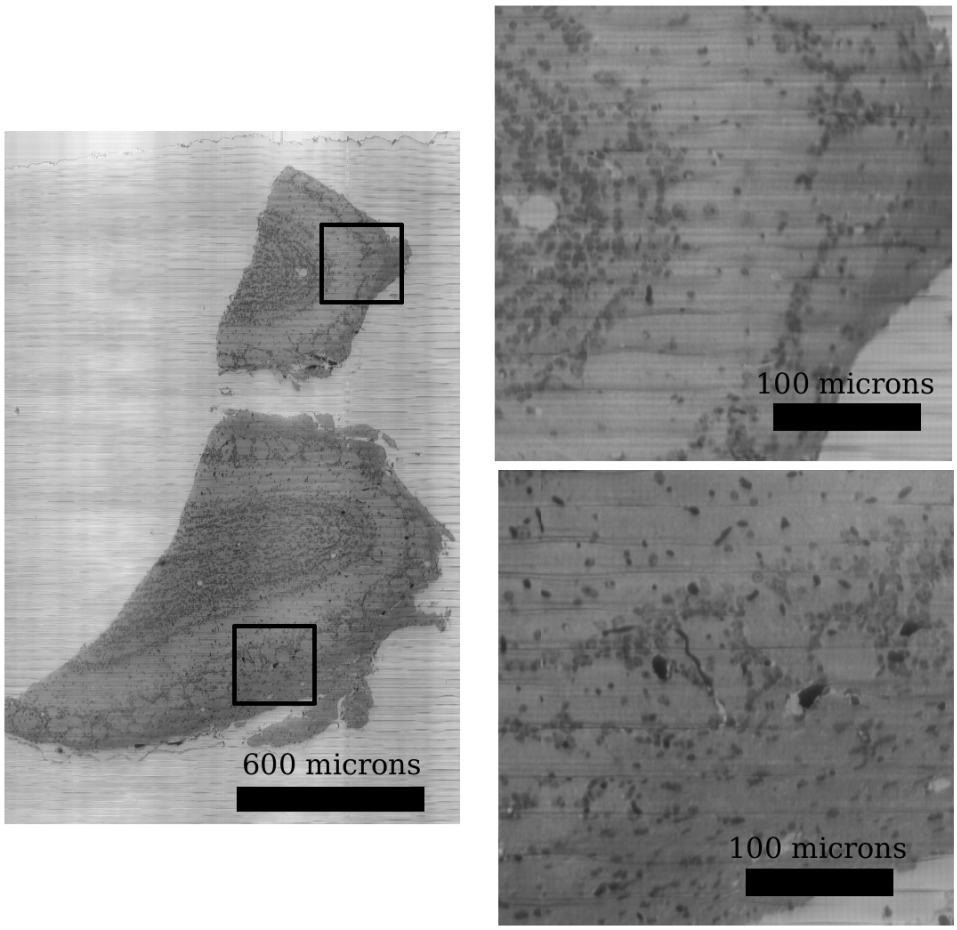


Fig. 8. Coronal Section Of The Mouse Olfactory Bulb. The complete section is shown (left) along with two close-up inserts (right). The tissue is stained with Nissl along with perfusion of India ink through the vascular system. The optical resolution of the image is $0.6\mu m$/pixel and the section is $1\mu m$ thick. (Adapted from [36].)

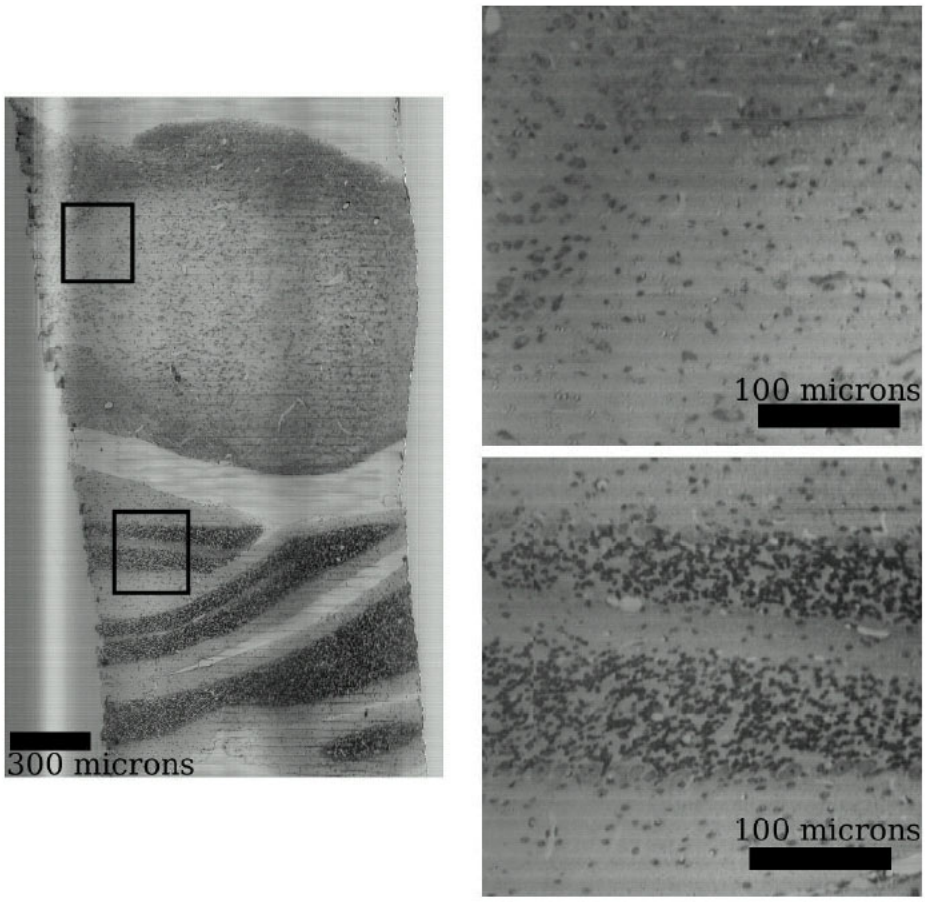


Fig. 9. Nissl Data from KESM. Coronal section of mouse brain stem is shown with part of the cerebellum visible to the bottom half. Close-up of the inserts are shown to the right. The pixel resolution of the images is 0.6 μm/pixel with a section thickness of 1 μm. (Adapted from [36].)

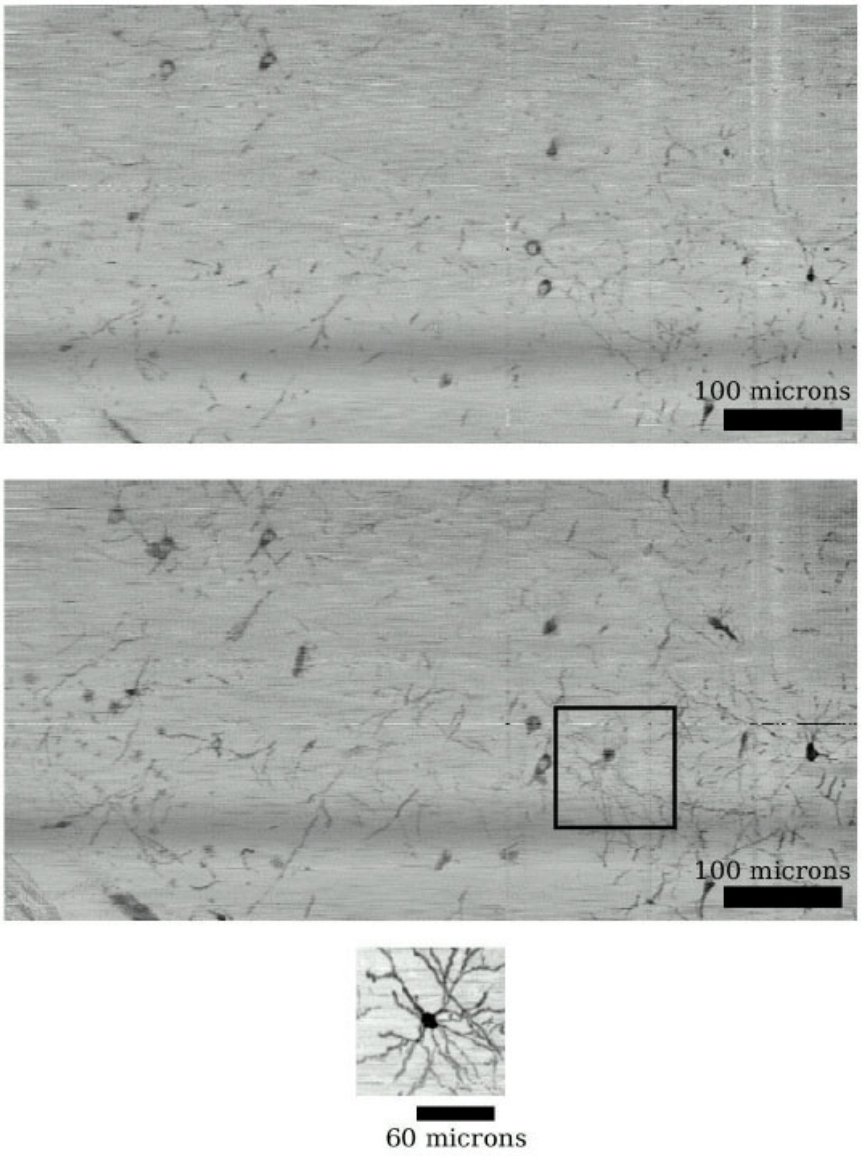


Fig. 10. Golgi Data from KESM. the stack of images generated by KESM can be viewed from the side of the stack (resectioning). A single resectioned plane is shown at the top. Golgi stain results in sparse data so often times it is easier to see familiar structures by overlaying multiple planes (middle). A single neuron can be observed when approximately 300 planes are overlayed (bottom). (Adapted from [36].)

staining of the entire vascular network (see Fig. 8).

In this chapter, I reviewed neurovascular units as a basic structural building block of the brain. We need to have high-throughput and high-resolution physical sectioning technology to map a complete connection matrix of such microstructures in the brain. This led me to introduce recent advances in microscopy technologies. Especially, I focused on the KESM, a high-throughput and high-resolution physical sectioning technology.

In the following, I will discuss a data acquisition method for the mouse brain and its automation which is a crucial factor for achieving high-throughput.

CHAPTER III

DATA ACQUISITION

Microstructure from the whole mouse brain requires very large image volumes of tissue containing great detail. Considering the massive amount of the data sets, the automation of sectioning and imaging is a crucial factor in acquiring the whole mouse brain image.

In principle, since sectioning and imaging in KESM are performed simultaneously, the captured images are perfectly aligned along with the cutting axis. This removes the need to carry out post processing which often uses complex and time-consuming algorithms to align the acquired images. Therefore, the sectioning and imaging of tissue through KESM can be done much faster than other exiting serial sectioning techniques. However, in reality there are many obstacles in achieving high-throughput imaging with KESM. One major bottleneck was the need for several manual steps involved in the acquisition processes. For years, the users of KESM had used generic commercial software to capture images with the custom built controller application for the movement of the precision positioning stage. Due to the limited inter-process communication capability of the commercial software, the user was supposed to save several images manually when the frame buffer in the image capture board was filled. Only after naming and saving the acquired images, the user was able to continue to section the tissue for a while until the frame buffer was filled again. These manual tasks were not a big problem when we sectioned and imaged only few hundred slices from the tissue, even though it was time-consuming and tedious. However, it becomes almost infeasible to maintain such manual intervention when we widen our exploration into the whole mouse brain which consists of hundreds of thousand slices.

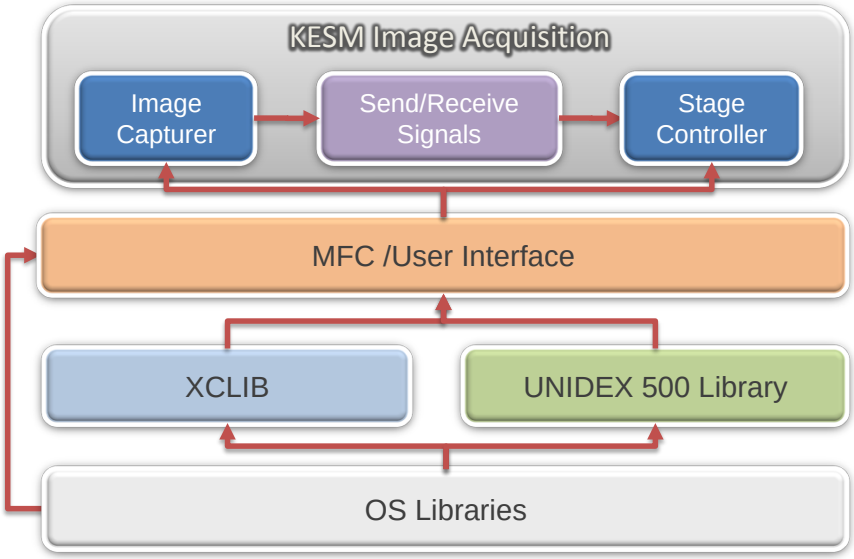


Fig. 11. Diagram of Data Acquisition System. XCLIB is a library routine to control camera actions. UNIDEX 500 library is for controlling the precision positioning system.

I have automated many of the manual tasks mentioned above to improve the actual data acquisition rate. All the naming and saving processes have also been fully automated by developing a custom image capture system rather than using a generic commercial software package (see Fig. 11 to see the overview of the system).

A. KESM Image Acquisition System

KESM comprises four major components: (1) precision positioning system, (2) microscope/knife assembly, (3) imaging system, and (4) data server. One interesting thing is that the moving part is not the knife but the specimen block mounted atop a three-axis precision positioning stage. A custom diamond knife is firmly mounted above the tissue block to a heavy granite stone in order not to vibrate when it digs into the tissue block. The precision positioning stage is located on the side of a tissue block slightly over the knife edge and moves it against the direction of the knife edge

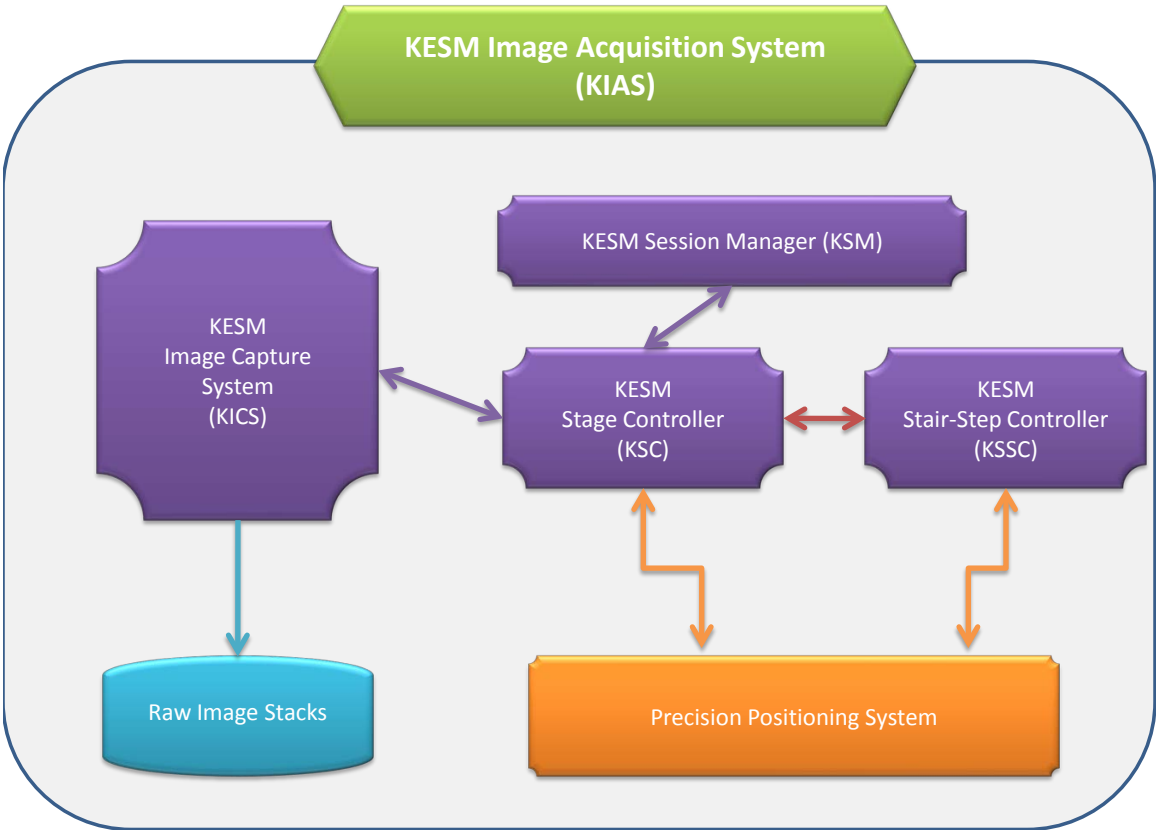


Fig. 12. KESM Image Acquisition System (KIAS) Components. KICS captures images accordingly with movements of the stage. The stage is controlled by KSC with KSSC which is an implementation of the stair-step cutting algorithm. KSM manages cutting sessions by saving and reloading all information of the previous status of the stage enabling us to resume the task where it stopped.

to section and image the tissue. The movement of the precision positioning system is controlled by a control software package, the KESM Stage Controller (KSC), which is a part of the KESM Image Acquisition System (KIAS), both of which I have developed. KIAS mainly consists of KESM Image Capturer System (KICS), KSC, KESM Stair-Step Controller (KSSC) and KESM Session Manager (KSM). Fig. 12 shows an overview of these components and their relations.

In KESM, the high speed line scan camera images a newly-cut thin tissue ribbon.

The line images are accumulated in two image capture boards until the tissue ribbon is scanned for the entire length. The accumulated line images become a single image in the frame buffers inside the two capture boards and then the image is saved to the data server [46]. The process of capturing and storing images is managed by an image capture software package, the KICS. Stage movement along the x-axis controlled by the precision positioning system triggers KICS to start or stop capturing images, and each of the captured images is automatically saved with a unique name specified by parameters such as the thickness of the section, the cutting speed of the stage, the current date and time, and the position in three-dimensional space where the tissue ribbon was sectioned.

Fig. 13 is a screen shot of KIAS. The user can change the system parameters such as the width of a slice, the thickness of a slice, the length of a slice, cutting speed, the interval and the duration of the triggering signal by using a parameter settings dialog box (Fig. 13, right). KICS is shown in Fig. 13 (left). *Load* button allows the user to load a video setting file, if the user wants to have different settings for the images. Other buttons in KIC such as *Snap* and *Save* are supposed to be used only for the purpose of testing. All capturing processes are performed automatically without any human intervention. Once sectioning starts, KSC sends control signals triggered by the stage movement to KICS to start or stop capturing the images.

The KIAS records the information for each tissue ribbon in its filename. The image filename below shows the specification.

yyyymmdd_hhmmss_x***x_cord***y***y_cord***z***z_cord***_t***thickness***_v***velocity***.tif

The first two fields separated by _ indicate the date and the time when the tissue ribbon is cut and imaged, and the next section indicates the x, y, and z coordinates

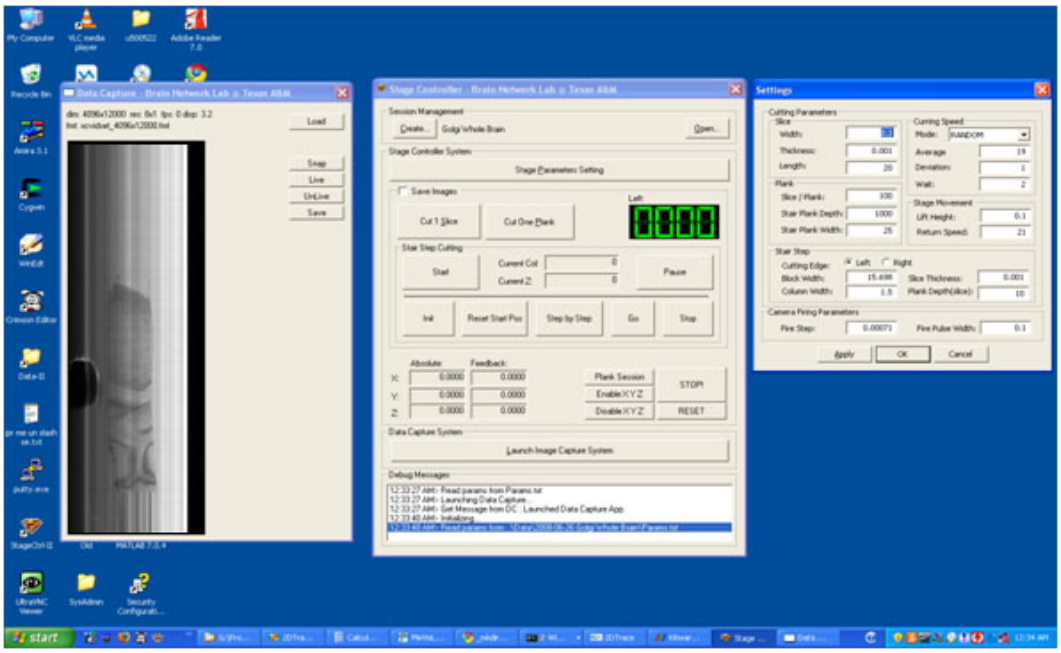


Fig. 13. KESM Image Acquisition System (KIAS) Screen-shot. (left) KICS captures images and shows the captured image to allow a user to monitor the operation. (center) KESM Stage Controller (KSC) can start, stop, pause, and resume a cutting session. The position of the stage is updated on the screen to help a user to monitor the movement of the stage. (right) Parameter Settings Dialog for KSC. A user can save and load parameters for the cutting. By setting the parameters, a user can change the cutting thickness, cutting speed, the number of columns, the number of depth for stair-step cutting, and etc.

of the upper-left corner of the image. Finally t and v indicate the thickness of the tissue ribbon and the velocity at which the tissue was cut. This scheme is simple and efficient in storing crucial information about the image. Here is an example of the image filename. This image was sectioned at 12:57:44 on April 11, 2009, and the section thickness was $1\mu m$.

20090411_125744_x***148.0967***y***26.6286***z***10.3750***_t***0.001000***_v***19.9778***.tif

B. Lateral Sectioning

Sectioning and imaging become more challenging for larger volumes of tissue. The area through the field of view (FOV) of the microscope objective is the maximum size of an image that we can acquire. However, the size of a tissue block (Fig. 14) is often larger than the FOV. For example at high resolutions, some neuronal elements such as the axon of a neuron may be stretched beyond the FOV. To address the problem, I fully analyzed and developed a lateral sectioning technique (Fig. 15) [48]. As the name indicates, the strategy is to section and image tissue in successive scan across the top of the tissue block. Lateral movement allows us to acquire images across the tissue block beyond the size of the FOV, so that this technique can be used for the acquisition of large volumetric data sets.

In the following sections, new terminologies about the KESM technology will appear, and they are rarely used in other contexts because of the uniqueness of the KESM. Here, I clarify the new terms before using them. The *tissue block* means the whole plastic block where the tissue is embedded (Fig. 14). Due to the property of lateral sectioning, multiple column-wise image stacks are generated after the sectioning. I call the column image stacks *column blocks* (Fig. 15). A thin tissue section cut

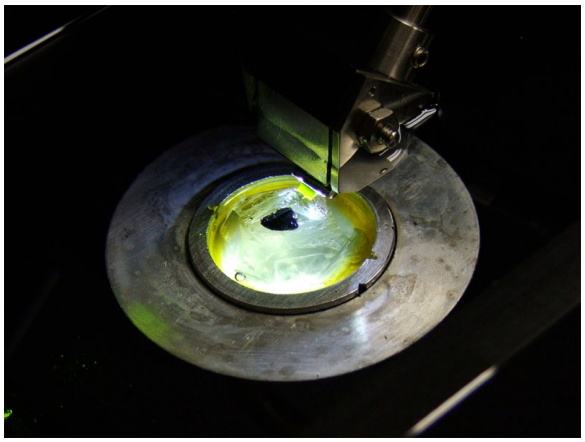


Fig. 14. Tissue Block. In the middle, the inner circular plastic block where tissue is embedded.

from a column block is a *tissue ribbon* (Fig. 15). Last, a plank can be defined as a group of tissue ribbons.

Here, I discuss lateral sectioning in more detail. Conventional serial sectioning techniques require only two axes to move the specimen block and the knife. First, one vertical lift is used to adjust the level of the specimen relative to the knife. Second, the movement of the tissue block toward the knife edge performs the actual cutting. In order to section and image the tissue across the block, the precision positioning stage moves along with y axis, perpendicular to the cutting direction (Fig. 15). The distance of the movement is decided by the specified width for a tissue ribbon. However, the lateral sectioning technique requires an additional axis that moves orthogonal to both of these axes. KESM uses a three axis precision positioning stage, so the additional axis can be used for lateral sectioning. There are two ways to perform lateral sectioning; (1) cutting an entire vertical column at a time and then moving to the next column, or (2) cutting all ribbons across the top surface of tissue blocks (across multiple columns) before moving on to the next depth. The first approach is straightforward, but it is limited by the distance between the

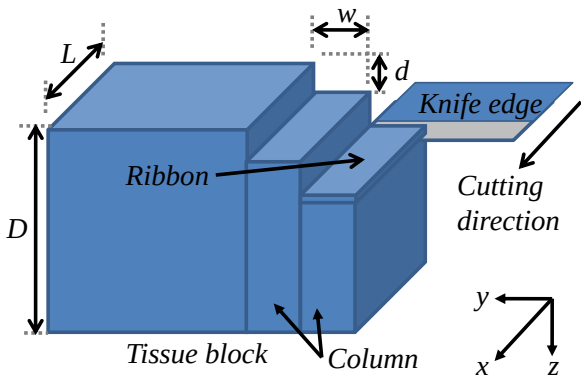
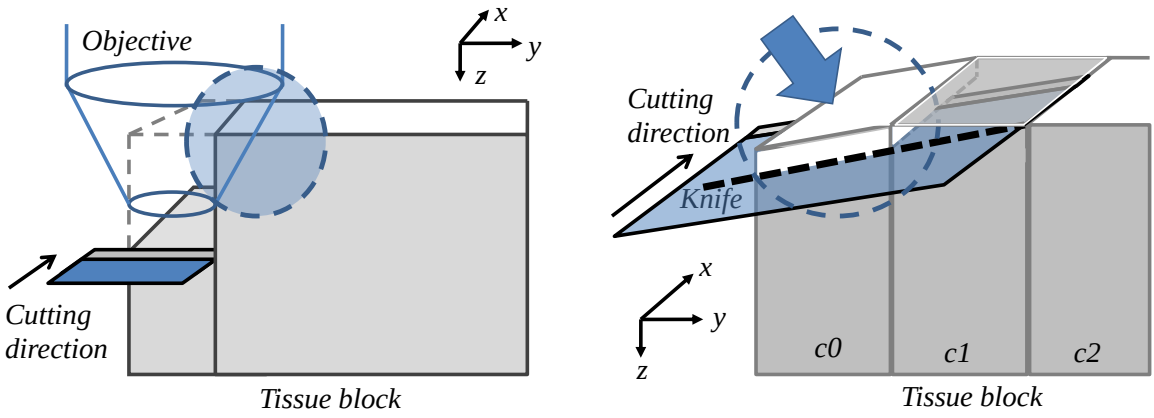


Fig. 15. Lateral Sectioning. The knife cuts the tissue block and images the tissue slices across the tissue block [48].

knife edge and the microscope objective. The side of the objective will bump into the uncut specimen block (Fig. 16 (a)). In addition, as the cutting depth increases, more contact between the edge of the knife and the neighboring tissue takes place, and this can cause tearing of the tissue and induce knife vibrations. The second approach is to cut a ribbon across the tissue surface without going down deeper into a column. Cutting a ribbon across the top surface seems a solution of the problem, but the knife is often slightly misaligned, and this can cause damage to un-imaged tissue outside the FOV of the objective (Fig. 16 (b)). Due to the problems mentioned above, these two straightforward approaches do not work with lateral sectioning. In order to successfully perform lateral sectioning with minimal tissue damage, the sectioning process should not make the objective bump into the uncut block when the knife goes down to the bottom of the tissue block and also should not cut across the tissue surface. To tackle the challenges, I derived precise quantitative requirements for stair-step lateral sectioning, along with colleagues at the Brain Networks Lab (BNL).

The stair-step lateral sectioning algorithm allows us to avoid the problems men-



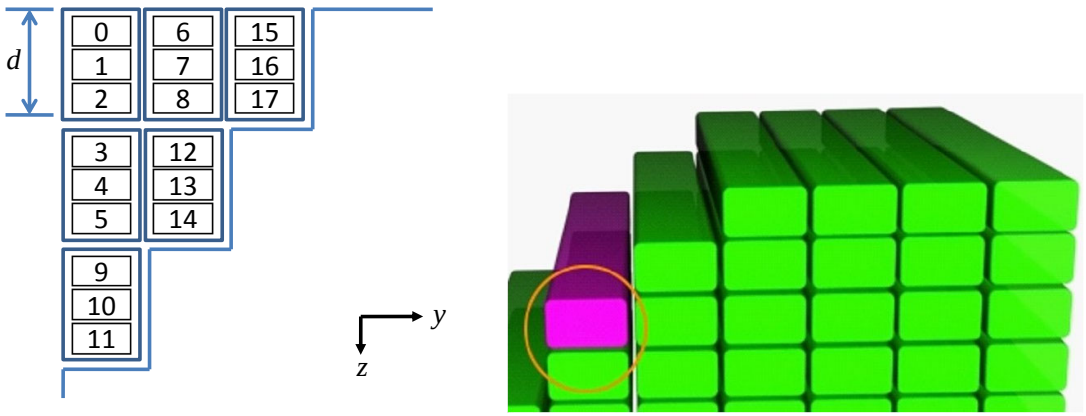
(a) (b)

Fig. 16. Problems In Lateral Sectioning. (a) The object comes into contact with the tissue. The area of dotted gray circle shows damage of the tissue by the objective. (b) Un-imaged tissue is being damaged when sectioning is taking place across the surface. The big arrow inside the dotted circle shows damage in the un-cut tissue column by the misaligned knife. [48].

tioned above by cutting small stacks of images in a stair-step fashion (Fig. 17). The order of sectioning should be as Fig. 17 (a) to avoid damage of *un-imaged* columns (Fig. 16 (b)). The key requirement here is that the stair step depth should be small enough so that the objective does not bump into the uncut neighboring tissue block.

In order to avoid damage to the tissue or the microscope objective, the stair step depth d needs to be:

- Small enough so that the microscope objective does not make contact with un-cut tissue
- Small enough to minimize knife vibrations and tissue tearing, and
- Large enough so that the knife assembly does not cause damage to neighboring columns when it moves to an adjacent column to cut a ribbon from them.



(a) (b)

Fig. 17. Stair-step Cutting. (a) The number presents the order of sectioning. In this specific case, the stair depth d is 3. (b) One bar represents a *plank* which is a group of ribbons within a stair depth [48].

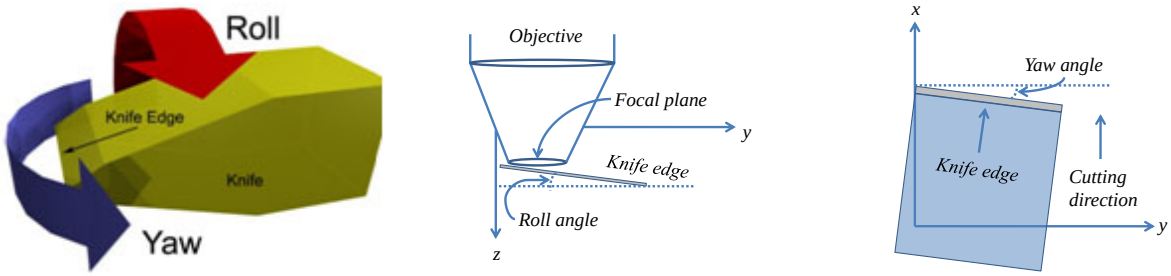
Another issue is that the knife is not often perfectly aligned to the tissue block. The scanning on the knife edge introduces error in imaging when the knife is not properly aligned. The misalignment of the knife can be due to roll and/or yaw (Fig. 18).

These misaligned angles of the knife are difficult to measure accurately. However, an upper bound for the knife misalignment angle can be determined, so that we can calculate an acceptable stair-step depth d (Fig. 19).

For the entire tissue section to be in focus, both side points of the knife edge within the FOV of the objective must be within the Focal Depth (FD). This means that the maximum misaligned angle of the knife θ can be determined by:

$$\theta \leq \arctan\left(\frac{FD}{FOV}\right) \tag{3.1}$$

where FD is the focal depth of the objective and FOV is the field-of-view. If the roll angle of the knife is greater than θ, this can be detected by the user because part of



(a) (b) (c)

Fig. 18. Knife Misalignment. (a) Error in roll and yaw. (b) The objective sees a tissue ribbon on the tip of the knife edge. If the knife is rolled, the captured images are distorted. (c) The yaw error of the knife edge introduces deformation of captured images as well [48].

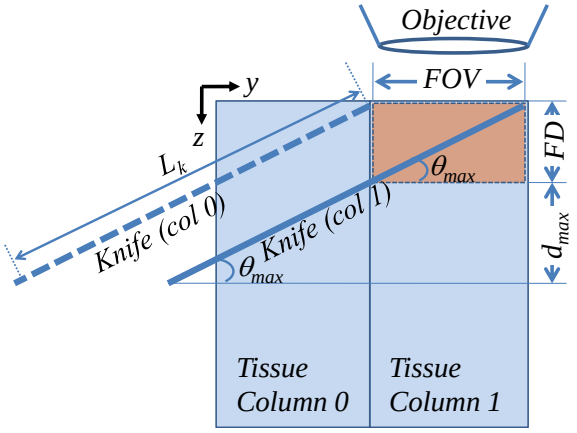


Fig. 19. Acceptable Stair-step Depth. The roll error is exaggerated just for the purpose of explanation [48].

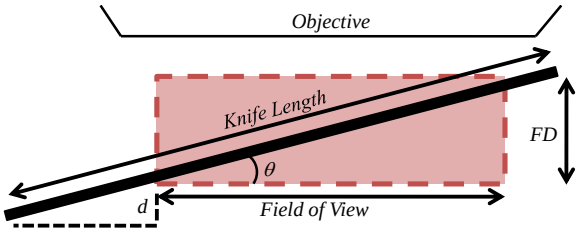


Fig. 20. Misalignment Of The Knife Relative To The Focal Plane. θ represents the maximum undetectable misalignment. In practice, θ is quite small resulting in $d < 3\mu m$ [48].

the image will be out of focus. If θ complies with the above constraint, the angle of the knife cannot be detected using the imaging hardware available in KESM. Therefore, we compute a worst-case depth d based on the maximum possible un-detectible value of θ:

$$d = L_k \sin\theta_{max} - FD \tag{3.2}$$

where L_k is the length of the knife (Fig.19) and $\theta_{max} = \arctan\left(\frac{FD}{FOV}\right)$. where FD is the focal depth of the objective and FOV is the field of view. If the roll angle of the knife is greater than θ, this can be detected by a user because part of the image will be out of focus (Fig. 20).

In short, the stair step depth d should be greater than d_{min} (Fig. 19) before lateral sectioning takes place, otherwise the left edge of the knife will damage the tissue column 0 (see the tissue column 0 in Fig. 19). In other words, the tissue column 0 should be cut by at least the depth $FD+d_{min}$ before cutting the column 1. The graphic user interface of KESM Stage Controller (KSC) allows a user to change the depth d in terms of the number of ribbons in a plank which is a group of ribbons within a stair depth, so the user can calculate the number of ribbons considering the cutting section thickness along the z-axis.

Algorithm 1 Stair-step sectioning

$nNumOfCols \Leftarrow (int)(dTotalBlockWidth/dColWidth)$

$nCurCol \Leftarrow 0$

$nCurColZ[0..(nNumOfCols-1)] \Leftarrow$ initial z positions

while $nCurColZ[nCurCol] < nMaxBlockDepth$ **do**

  for $nIndexRibbon$ to $nPlankDepth$ **do**

    Section a tissue ribbon

    $nCurColZ[nCurCol]$ += $nRibbonThickness$;

  end for

  $nPlankThickness \Leftarrow nRibbonThickness \times PlankDepth$

  if $nCurColZ > (nNextColZ + nPlankThickness \times 2)$ **then**

    $nCurCol \Leftarrow nCurCol + 1$

  else

    $nCurCol \Leftarrow nCurCol - 1$

  end if

end while

I implement stair-step sectioning by maintaining a height field of the tissue surface with colleagues at BNL. I can then constrain cutting so that the height difference between two columns never exceeds the calculated value d from Equation 3.2. Changes to the cutting parameters (e.g. column thickness) can be handled robustly by simply re-sampling the height field in order to insure that there is no loss in data. The algorithm used to constrain the sectioning process is shown in Algorithm 1.

To summarize, I discussed how the KESM Image Acquisition System (KIAS) was designed and built, KIAS being an automated high-throughput data acquisition system for KESM featuring stair-step lateral sectioning implemented in the KESM

Stage Controller (KSC). Sectioning and imaging a large specimen such as the whole mouse brain takes a long time (several days or even several weeks). Based on our experience, sectioning and imaging the whole mouse brain would take several weeks in 8 hour shift in the current KESM setup. The user should be able to stop the system whenever it is needed, and the cutting session should be able to resumed later with exactly same conditions. The management of multiple sessions is conducted by the KESM Session Manager (KSM). KESM Image Capturer System (KICS) is also developed to save the acquired images automatically.

CHAPTER IV

DATA PROCESSING

Acquired images using KESM have several different types of artifacts. Lateral sectioning often introduces column misalignment even though lateral sectioning itself allows us to image larger tissue block than the FOV of the objective. Also, KESM uses the knife edge as a collimator so that the misaligned knife may introduce different intensity levels inside a same image (Fig. 21). These artifacts should be effectively removed to reconstruct three-dimensional structure from the volumetric data sets acquired using KESM.

This chapter is divided into two main sections. First, I focus on describing the column misalignment problem and its solution. The second part is devoted to data processing methods to remove noise due to intensity irregularities.

All data processing modules share the same software structure (see Fig. 22). Image Segmentation and Registration Toolkit (ITK) [50] is used for image processing, Visualzation Toolkit (VTK) [51] for visualization, and Qt [52] for cross platform user interface.

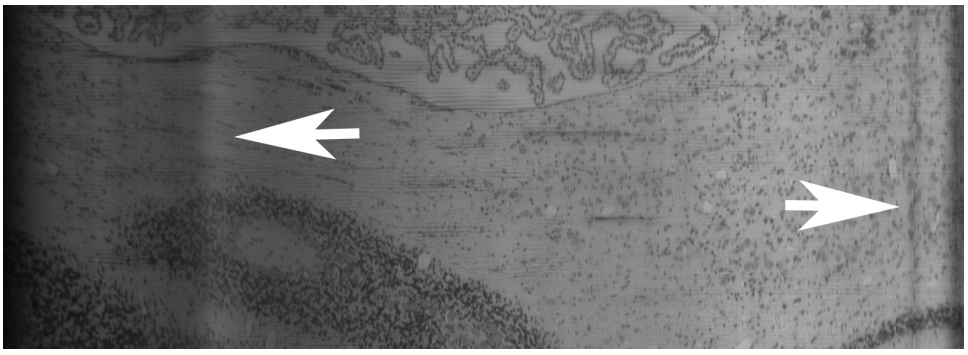


Fig. 21. Overall Intensity Shift. The overall intensity shift along the x-axis is due to knife misalignment (arrows). Adapted from [49].

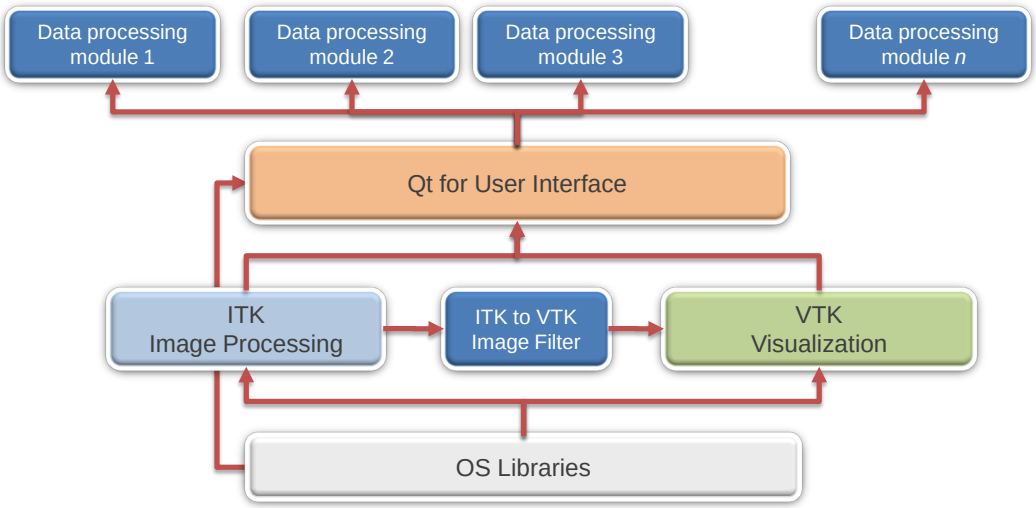
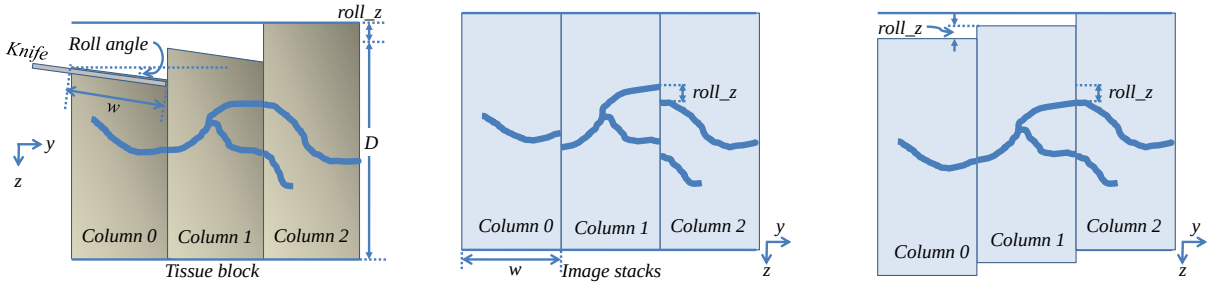


Fig. 22. Diagram For Data Processing. ITK [50] is used for image processing, VTK [51] for visualization, and Qt [52] for cross platform user interface.

A. Misalignment between Image Stacks

KESM captures line images when a ribbon being cut is at the tip of the knife edge as the knife sections the tissue ribbon from a specimen block. This technique that scans the knife edge rather than the entire tissue block surface improves the imaging speed without sacrificing image resolution. On the other hand, the scanning process can introduce imaging artifacts if the knife is not aligned properly. When a user focuses on only one column whose width is less than or equal to that of the Field Of View (FOV), the image distortions due to knife misalignment could be ignored, because the overall structural properties are intact.

Lateral sectioning in a stair step fashion removes the limit in cutting depth, compared to conventional serial sectioning techniques and extends the section width beyond that of the FOV of the objective by cutting multiple lateral columns. However, image distortions due to knife misalignment become a more complicated problem when lateral sectioning generates image stacks in multiple columns from a single



(a) (b) (c)

Fig. 23. Misaligned Columns. (a) Cutting tissue ribbons from multiple columns with a tilted knife. (b) The acquired image stacks. (c) Finding the amount of the misalignment.

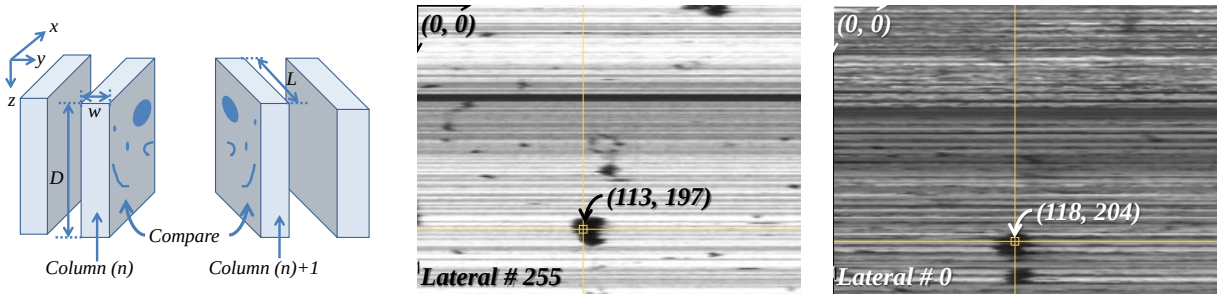
tissue block.

Next, I discuss two types of image distortions due to knife misalignment. As a possible solution, block alignment will be described and its problems will be discussed. Finally I will describe potential solutions for the image distortion problem as proposed work.

1. Column Misalignment Detection and Correction

Suppose the knife is rolled in α degrees and yawed in β degrees (For the roll and th yaw error, see Fig. 18). The roll angle α introduces deformation in the captured images. The image volume captured from a column will be tilted due to the roll. In such a case, the knife sections the tissue block and captures an image not from an exactly horizontal tissue surface but from a tilted surface with the *roll angle* (Fig. 23 (a)). When the captured images are stacked, and the columns are aligned (Fig. 23 (b)), we can see that the fibrous structures are not perfectly aligned as they were in the original column due to the knife roll.

Microstructure within each column is subject to be distorted a little bit when the knife is rolled. However, it is not a severe problem as far as one column is concerned,

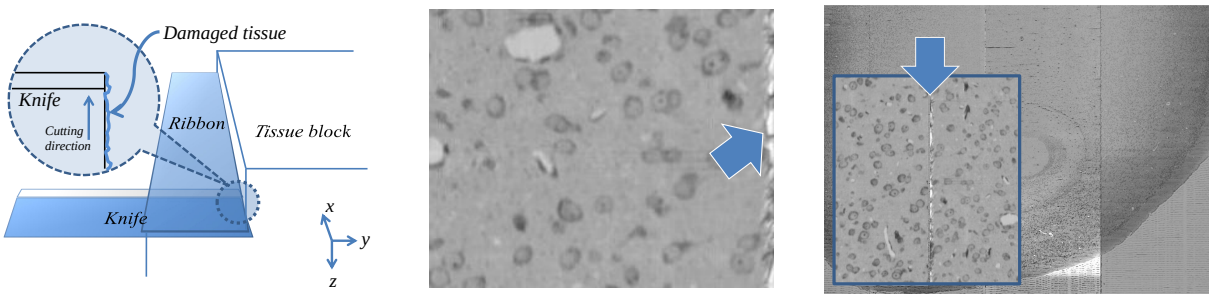


(a) (b) (c)

Fig. 24. Column Alignment. (a) Comparing two lateral image stacks. (b) A real image from a lateral part of a column. (c) A lateral image with tissue damage.

because basic structural properties in the column block will be intact. When the adjacent columns are put together, on the other hand, the misalignment between columns can be prominent and can be identified from the error in the orientation of the knife across columns (Fig. 23). We have to find out $roll\_z$ (Fig. 23 (b) and (c)) to adjust the misaligned columns. The knife orientation introducing misalignment of the tissue columns is fixed from the beginning unless mechanical error occurs. So, the $roll\_z$ derived from any two neighboring columns can be applied to all other columns. It is straightforward to get $roll\_z$ and the roll angle α. First, get the right lateral surface image from the image column n and get the left lateral image from the image column $n + 1$ (Fig. 24). Then, compare the two lateral surface images which are bound to be slightly different, and find out how much displaced the two images are. However, it is not easy to match two adjacent lateral surface images due to tissue damage from lateral sectioning.

During the sectioning process, some tearing occurs at the interface of two neighboring columns (Fig. 25) because the knife only cuts beneath the tissue ribbon and not on the side. One side of a tissue ribbon is actually torn from the neighboring columns. The amount of actual data loss is quite small and the damage is usually



(a) (b) (c)

Fig. 25. Tissue Damage Due To Lateral Sectioning. (a) Damage occurs due to tearing at the interface of two columns. (b) This results in some data loss at the edge of the image (arrow). (c) Two neighboring blocks show tissue damage between them.

less than $5\mu m$ in width in most cases. If a 10X objective is used, the horizontal pixel resolution is $0.6\mu m$ and the FOV is $2.5mm$: 0.2% ($= 5\mu m/2.5mm$) of data is lost in the torn part. Even though the tissue damage is small enough to be ignored compared to the total data width, the damage looks quite visible when we look at the surface made up to the torn side (Fig. 24 (c)).

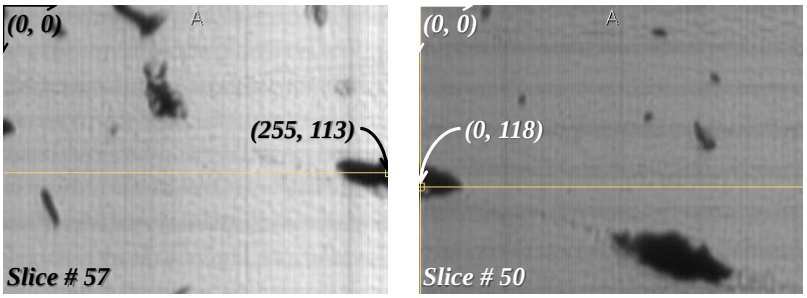
One way to find out the misplacement of the images is to look for prominent structural features throughout the neighboring pair of columns. In most cases, the whole data set has multiple columns, so there is a high probability of finding prominent features across the neighboring columns. Fig. 24 (b) and Fig. 24 (c) are two such cases. Even though the surface images are noisy due to tissue damage, the center point can be identified from the circular features from each lateral surface image. In this example, we can say two surfaces are misplaced by $5(= 118 - 113)$ pixels along the x axis and $7(= 204 - 197)$ pixels along the z axis. Since the offsets are based on the knife angle which is unchanged during a cutting session, this misplacement values can be applied to all other columns.

The knife can also often have yaw even though it could be a very small amount.

This angle introduces another distortion of the captured images (see 2 for detail). Each of the ribbons captured is slightly skewed (yaw_x) if the knife is yawed. The ribbons should be aligned to exactly restore the original morphological properties. As in the case of distortion from the knife roll angle, the structural information from the acquired images could be a little bit different from the original one. The structural properties, however, can be restored by moving a ribbon backward or forward along the x axis. Once yaw_x is determined, the same amount of yaw_x can be applied to all the other ribbons, because the yaw error is introduced by the knife misalignment just like the roll error. The aligned ribbons recover the original morphological properties across the ribbons but the final result is slightly skewed. In most cases, the yaw_x is so small that the skew distortion can be ignored. We can skew the final images again in the opposite direction of the original distortion to restore the original spatial properties, but skewing back the images might introduce further distortions.

When examining the yaw error, we have to consider the vertical distortion. According to the previous result where the image is misplaced by 7 pixels along with y axis, it is safe to say that the ribbon m from the column n and the ribbon $m+7$ from column $n+1$ should be compared when we measure the yaw misalignment. Fig. 26 shows that the 7 pixels are quite a reasonable outcome, because the same structural information is found in ribbon number 57 and ribbon number 50. When it comes to the vertical misplacement in Fig. 26, the center points of the circular objects from the two neighboring ribbons are misaligned by 5 pixels, and this is a consistent outcome with the column misplacement result.

In this section, I introduced a manual method in which the user can align two neighboring image columns by comparing their lateral surface images facing each other. There exists a potential solution that can be performed automatically. As Fig. 24 (b) and (c) show, the outermost lateral surface is severely torn. Therefore, it is not



(*a*) (*b*)

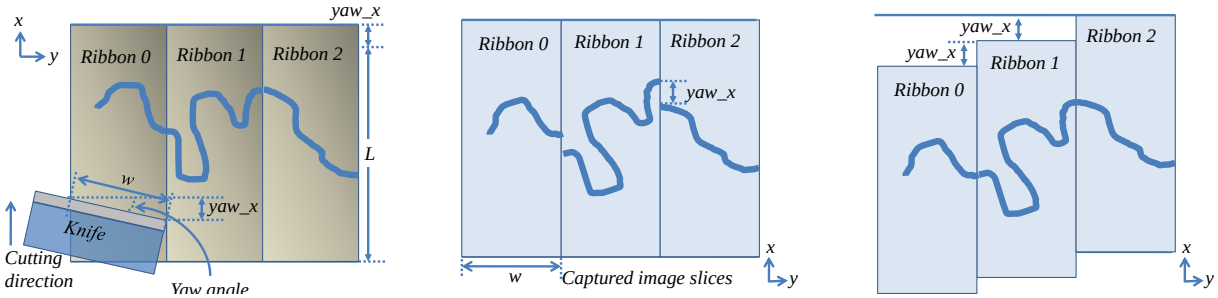
Fig. 26. Measuring Yaw Misalignment. (a) Part of a ribbon from a slice. (b) The neighboring part of ribbon from the adjacent slice.

easy to find matching features from two neighboring surfaces facing each other. Even though the lateral surface is too much rugged, if we go slightly deeper into the surface by several pixels, we can see a clearer image. In this case although the lateral surface to be compared would be farther apart, subsequently more accurate comparison of the two neighboring lateral images can be possible. More importantly, using a clearer image in the comparison can shed light on finding the amount of knife misalignment in an automatic way.

2. Distortion of Tissue Ribbon

As mentioned in the previous sections, small misalignments in the knife orientation are difficult to detect and result in the knife contacting the specimen surface at a slight angle θ. Although the roll angle of the knife is the only misalignment that can cause unwanted tissue damage, note that it is also possible for there to be a slight yaw misalignment (Fig. 27 (a)). Both of these knife angles cause each column to be imaged at a slight skew (Fig. 27).

When the image stacks are placed next to each other, this results in misalignment at the interface between columns (Fig. 23 (b)). We fix this misalignment by



(a) (b) (c)

Fig. 27. Distortion From The Yaw Error. (a) The knife with a way angle cuts the tissue block. (b) Acquired images. (c) Aligned tissue ribbons.

applying a translation to each column to compensate (Fig. 23 (c)). The direction of translation parallel to the plane created by the column interface. Each component of the translation (x and z) is proportional to the angle of misalignment. We note two important properties of these offsets:

- The offsets are based on knife misalignment along two axes. These angles are too small to be measured with the KESM optics but can produce noticeable distortion in the data set.
- Since the offsets are based on the knife angle, they are constant throughout the entire data set.
- Practical constraints on the knife angle (discussed above) limit these offsets to very small values (1–4 pixels).

In my initial experiments, I attempted to determine these offsets automatically by aligning images representing the interface between the columns. The major difficulty with this approach is that tissue tearing, and therefore data loss, occur at this interface. Although it is possible to acquire image data slightly *inward* of the

interface, this sampling was too coarse to allow effective alignment based on image data alone.

Since the knife misalignments are constant throughput a data set, we found that the offsets need to be determined manually only once. This was done by selecting a small volume of tissue at the interface of two neighboring columns. These volumes were aligned by using filament structures, such as vasculature and neuronal processes, as fiducials. Many of the larger filaments have trajectories that can be interpolated through missing data at the interface. After an initial estimated alignment, we can then explore several other samples along the interface to evaluate and refine the offsets. Since imaging can occur uninterrupted for several hours at a high data rate (approximately 30 GB/hour), one-time manual alignment (requiring only a few minutes) was an efficient method for determining offsets between neighboring columns.

B. KESM Volume Noise

KESM exhibits several types of noise. Irregularities in lighting is a main cause of noise in KESM because it uses the knife edge as a collimator as well as a cutting device. Knife vibration is also a cause of the lighting irregularities. Knife chatter is well known in machining, but is not a problem in conventional imaging devices [49]. However, due to the fact that KESM images data as it is being cut, knife vibration becomes a issue in KESM.

1. Noise from Lighting Defects

Since the problem of knife chatter marks were successfully addressed in [49], I focus algorithms to remove noise due to irregularities in lighting. A fluctuation in illumina-

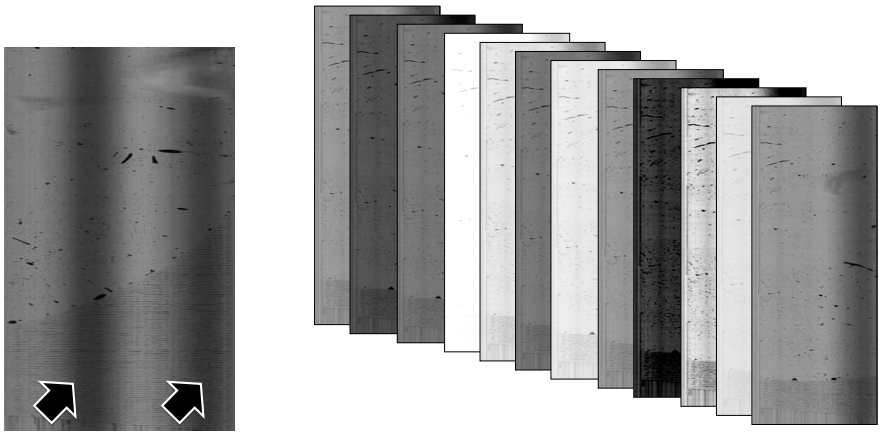
tion across the knife edge is one of the major lighting defects. The sources of defects are as follows.

- Misalignment between the objective connected to the camera and the edge surface of the knife. This results in intensity irregularities across the *width* of the image.
- Fluctuation in illuminator power.
- Different exposure time to illumination. Varying cutting speed to reduce knife vibration causes different exposure time to the line scan camera.
- Defects on the surface of the knife edge. Since the knife edge is used as a collimator, the defects on the edge cause refraction of illumination. This results in dark vertical stripes in the images.

Images from the KESM often suffer from not just intra-image intensity difference but also inter-image difference. Slight misalignment between the knife edge and the objective (eventually the line scan camera) causes uneven illumination on an image across the horizontal direction (Fig. 28 (a)). The overall intensity level in a part of an image is higher or lower than the rest of the image.

The cutting speed randomly changes in each sweep to reduce knife chatter marks in the specimen being cut. The difference in the speed of knife movement (it appears to be knife movement but actually specimen mounted on top of the stage moves) causes different light exposure time to the line scan camera. In other words, when the speed is slower, more photons come to the camera which results in a brighter image. Fig. 28 (b) shows the intra-image intensity difference.

Dark vertical stripes are also caused by non-uniform illumination but the main reason is not from changes of illumination itself. It is because of defects on the knife



(a) (b)

Fig. 28. Intensity Difference in Captured Images. (a) Intensity irregularities across the image width. The black arrows indicate darker vertical areas in the image. (b) Different intensity levels between images.

edge. The unique way of KESM imaging, i.e. the knife also being used as a collimator, causes dark vertical lines if the knife edge has defects.

Fig. 29 (a) represents intensity levels from a normal area. In contrast, Fig. 29 (b) shows intensity levels having two different layers of background intensity ranges. The brighter one (the white downward arrow) indicates the background of normal intensity areas neighboring the vertical stripes. The gray arrow points to the area of the background of the vertical stripes. There are still clear distinctions between foreground and background objects even inside the vertical stripes.

2. Image Processing Techniques

The unique noise in KESM images should be removed effectively to build a uniform volumetric data set for segmentation. I specifically focus on image processing techniques that use only local information inside an image so that the processing can be run on heterogenous systems in parallel. Parallelism in data processing is important,

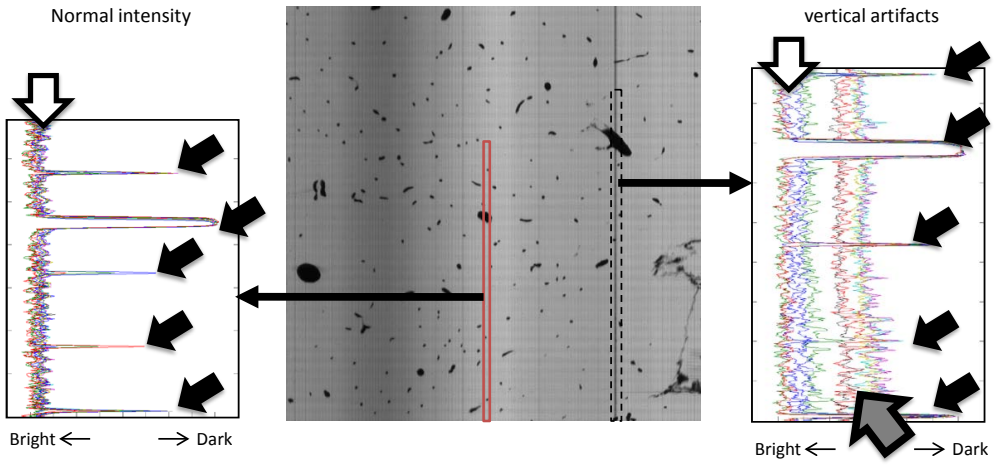


Fig. 29. Vertical Stripes. Intensity levels of the foreground objects (slanted black arrows) are different from those of the background (hollow downward arrows). (left) An area with normal intensity. (center) An image example with vertical stripes. (right) The gray arrow indicates the intensity level of the background of the vertical stripe.

to deal with large amounts of data resulting from the KESM.

The first step of noise removal is to normalize overall intensity levels within an image. Each pixel in a row is normalized based on the median value of the row. The same process is applied for the columns in the image. The normalization process is iterated for all rows in an image, and then all columns in the image, according to Eq. 4.1, 4.2, and 4.3. The desired median intensity level, L, is defined as a base background intensity level to normalize the inter-image intensity difference in the following (see Fig. 30).

$$T_x = \frac{p[x, Y]}{M(Row(I, Y))} \times L \tag{4.1}$$

$$T_y = \frac{p[X, y]}{M(Col(I, X))} \times L \tag{4.2}$$

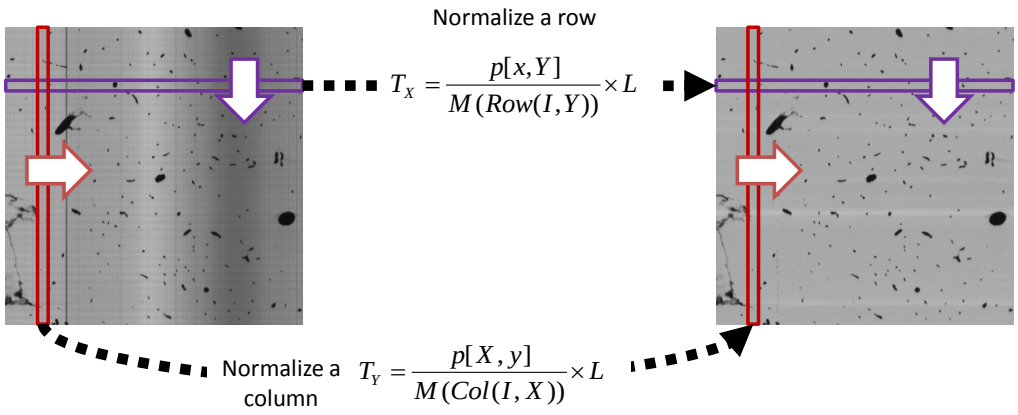


Fig. 30. Intensity Normalization. First, all rows are normalized based on the median value of each row. After the normalization of the rows, all columns are processed in the same way.

$$\bar{I} \quad = \quad \forall_x T_X \left(\forall_y T_Y (I, y), x \right) \tag{4.3}$$

where $M(\cdot)$ is the median value of a series of pixels, L is a constant representing a desired median intensity level in inter-image variation, $Row(I, Y) = \forall_x p[x, Y]$, $Col(I, X) = \forall_y p[X, y]$, $I = \forall_x \forall_y p[x, y]$, and $\bar{I}$ is the normalized image.

Noise from non-uniform illumination and knife defects can be simply solved by the intensity normalization method based on the median value. However, it becomes more difficult if the stripes are excessively dark. The excessiveness can be determined by the clear distinction between intensity levels of background and foreground objects. In other words, we can say that the dark stripes are *excessive* if the intensity levels of the background are not clearly distinctive from those of the foreground objects. When vertical stripes are excessively dark, the intensity normalization algorithm cannot recover from the noise. The intensity levels of foreground objects (dark arrows in Fig. 31) after normalization based on median values are even brighter than those of the background (gray arrows in Fig. 31) in the original image which means that normalized foreground objects become brighter than other foreground objects (see

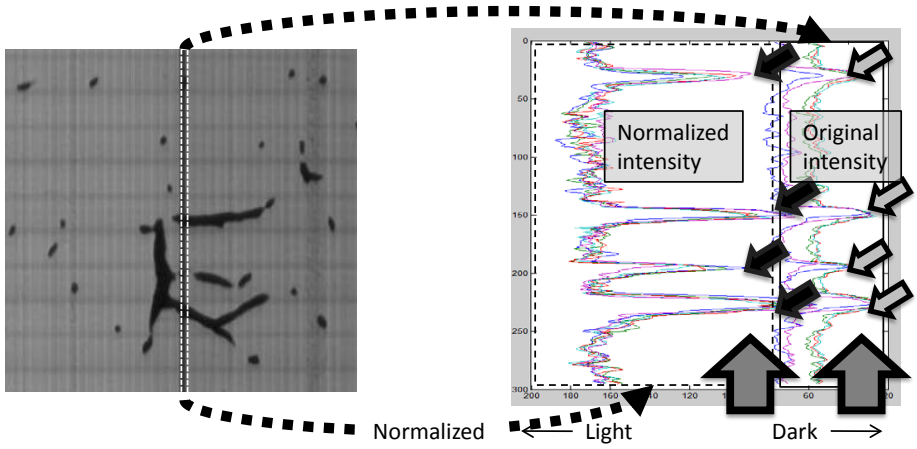
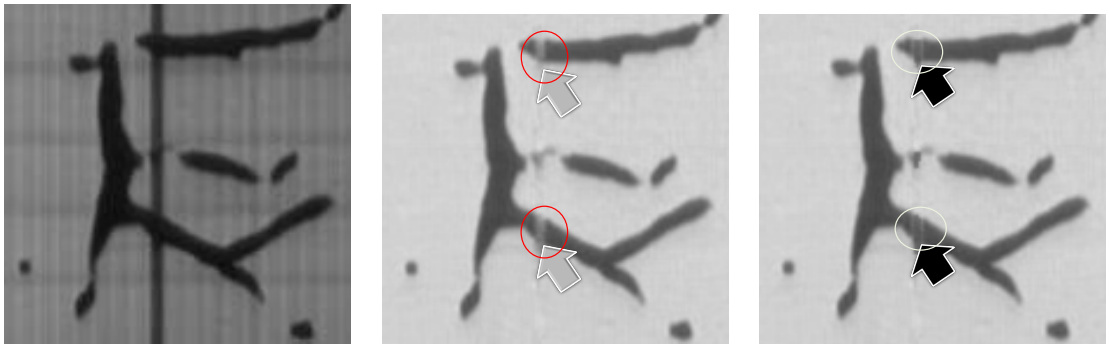


Fig. 31. Excessively Dark Vertical Stripe. The intensity level of foreground objects in the dark vertical stripes becomes too bright (over-normalization) compared to foreground objects in other areas (compare areas pointed by two larger gray arrows).

Fig. 32 (b)).

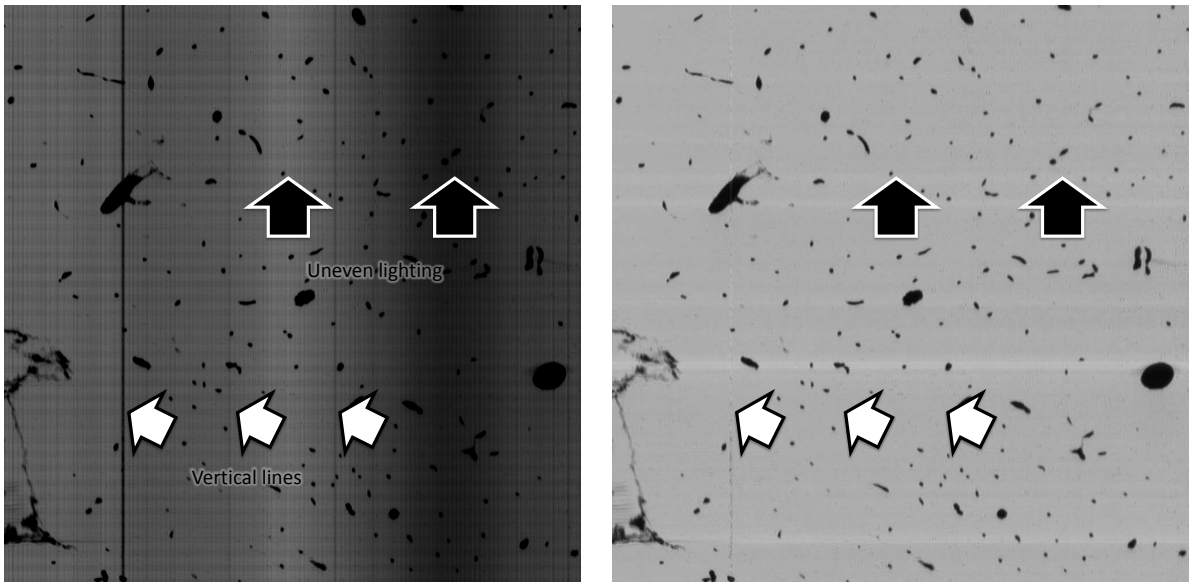
To address this excessive normalization problem, I introduce the *Selective Normalization* method. This new method *selectively* normalizes columns in an image only when the median value of a column is not too small which means that the whole column is generally too dark to be recovered. In this specific case, the new algorithm maintains intensity levels of foreground objects while the background is being normalized. Fig. 32 illustrates the processes. Without using the *Selective Normalization* method, the vertical stripes would not have been properly recovered as shown in Fig. 32 (b). The new method can recover the vertical stripes (see Fig. 32 (c)). A result of the noise removal process is shown in Fig. 33.

To summarize, KESM images have inter-image and intra-image intensity unevenness. In order to build clear volumetric data sets for segmentation, the uneven intensities should be properly recovered. An intensity normalization method based on median values of rows and columns is used to remove the unevenness both in an



(a) (b) (c)

Fig. 32. Selective Normalization. (a) An example of excessively dark vertical lines. (b) Foreground objects (arrows) become too much bright. (c) Fixing over-normalization. Foreground objects (arrows) are properly recovered.



(a) (b)

Fig. 33. Normalization Result. (a) The original image before normalization. The image has dark vertical lines (white arrows) and uneven intensities across the horizontal direction (black arrows). (b) Vertical lines are recovered and uneven intensity levels are successfully removed.

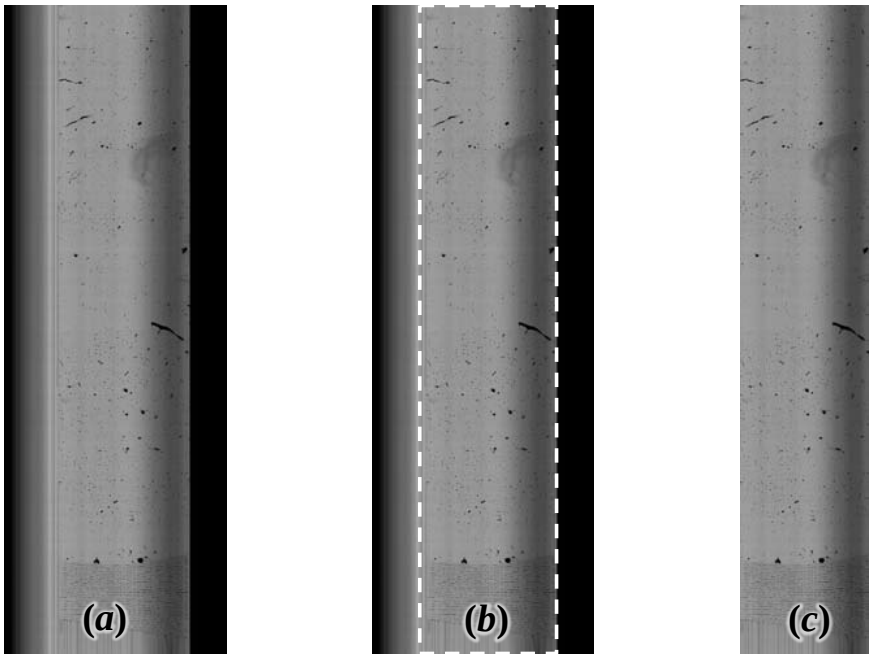


Fig. 34. Extra Regions In A Captured Image. (a) A captured image has extra regions in its left and right side containing no data. (b) The image should be cropped to remove those areas. (c) An example of the cropped image.

image and between images. The excessive normalization problem can be solved by the selective normalization algorithm.

C. Image Chunk Alignment

Captured images from KESM have extra regions containing no data because the Field of View (FOV) of the objective should be larger than the width of the tissue ribbon. Fig. 34 (a) shows a tissue ribbon captured. The white dotted box in Fig. 34 (b) indicates the actual tissue area.

In principle, KESM does not need to do image registration because a tissue ribbon is imaged as it is being cut. The high speed line scan camera images the same location of the edge of the knife all time. However, in practice, several image block chucks are generated (see Fig. 35) in image stacks since the knife position may be

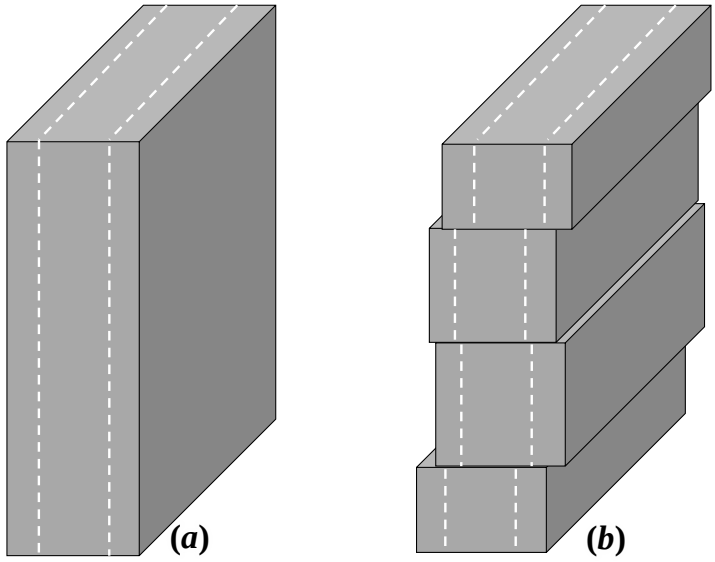


Fig. 35. Misaligned Image Chunks. (a) Images are supposed to be aligned an image stack, in principle. (b) The knife needs to be re-installed in the middle of cutting sessions, in practice, so that the angle of the knife often changes a little bit from the previous session. It causes a misaligned image chunk.

slightly changed after knife reinstallment. The knife often needs to be reinstalled in the middle of cutting sessions for maintenance. When the knife is reattached to the knife assembly, the angle of the knife edge tends to be slightly different from that of the previous session. This causes misaligned image chunks in a column of image stack.

Due to the misalignment of the image chunks, cropping the tissue region becomes a painful and time consuming manual job. I have developed an automation algorithm to crop the tissue area using template matching based on Sum of Absolute Differences (SAD) (see Eq. 4.4 and Fig. 36).

$$d^x_{SAD}(x,y) = \sum_{c=0}^{(w_t-1)} \sum_{r=0}^{(h_t-1)} |p_s(x+c, y+r) - p_t(c,r)| \tag{4.4}$$

where w_t and h_t is the width and the height of the template image respectively, p_s is

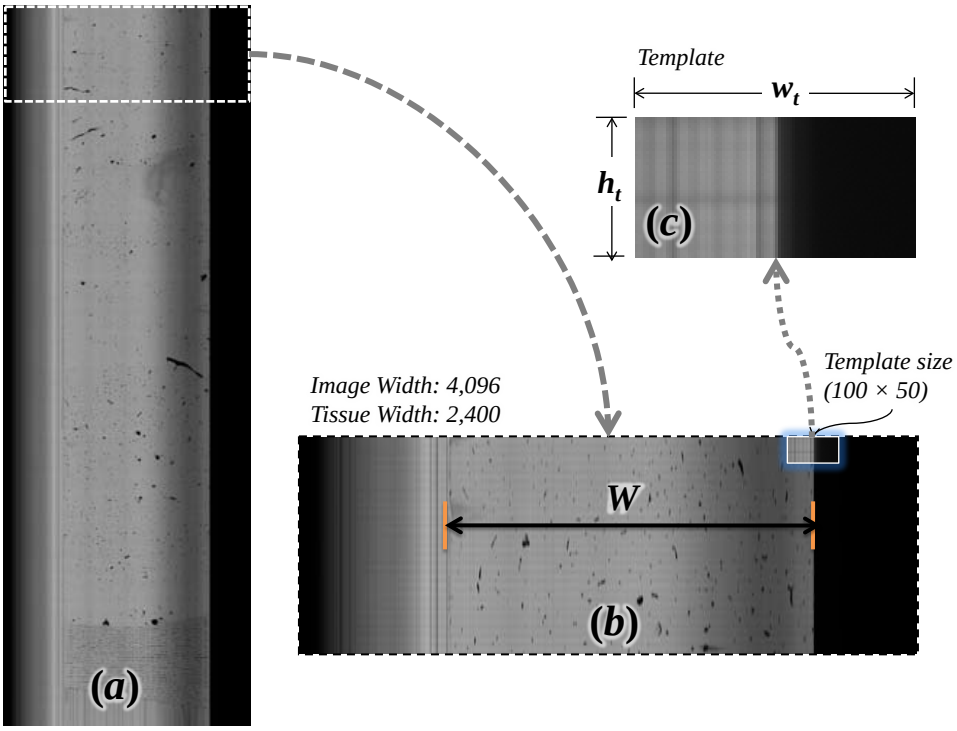


Fig. 36. Cropping By Template Matching. (a) An example of a captured image. (b) Part of an image shows the tissue width. (c) Template (100 × 50).

a point in the raw image, p_t is a point in the template image.

The left edge of the tissue image is not clear since the area was on the knife edge when the image was being cut while the right side has clear cut between the foreground and background information. The x start position of the template in the raw image is the minimum value among d_{SAD} (see Eq. 4.5) meaning that the smallest difference exists between the template and the area of the image.

$$d^x_{SAD} = min_x(d_{SAD}(x, y)) \tag{4.5}$$

Thus, the x start and end positions of the tissue area (x_{right}) can be calculated by Eq. 4.6 and 4.7

$$x_{right} = d^x_{SAD} + \frac{w_t}{2} \tag{4.6}$$

$$x_{start} = \left(d^x_{SAD} + \frac{w_t}{2}\right) - W \tag{4.7}$$

In this chapter, I discussed data processing methods for the image stacks acquired by KESM. Lateral sectioning was introduced to overcome the limitation of the FOV. Due to the fact that lateral sectioning generates multiple images stacks (column volumes), column misalignment and distortion in acquired ribbon images were introduced to the image columns. I described these problems and their solutions in this chapter. In addition, image noise owing to the irregularities in illumination was described and I discussed possible solutions to recover from the noise. Besides these image process techniques, I introduced an automatic cropping algorithm to address the problem of misaligned image chucks in an image stack.

In the following, I will describe an efficient data management method to deal with large data sets to be able to be used in visualization and analysis of such data.

CHAPTER V

DATA MANAGEMENT

The information seeking principle of *overview first, zoom and filter, and then details-on-demand* [53] is useful when we develop a system to explore large volumes of data. To provide the user an overview of the data, the whole data should be visualized at a lower resolution at the first stage. However, it is impractical to generate the data at runtime in different scales from the huge amounts of data. Therefore, to provide the features such as overview and zoom in/out, the data should be presented in different scales before the visualization. The primary examples of data representations for three-dimensional volumetric data are octree [54] and treemaps [55]. The main focus of these multi-resolution and hierarchical data representations is to support fast access and efficient rendering of data. Instead, in this dissertation, I focus on an efficient data storage and retrieval scheme for interactive exploration of large data sets.

The management of such large and complex data collections is also a challenging task, so researchers often use general purpose data models to manage interactive exploration of remotely stored large data [56]. HDF5 (Hierarchical Data Format) [57] is such an example and can present complex data objects. HDF5 is also a software library that provides high-level APIs. For the case of KESM data, however, the main requirement of the data representation is to provide an efficient data storage and retrieval method in a local system. The types of the data for KESM are restricted to images, three-dimensional volumes, reconstructed geometric data, and some of metadata. To fulfill the requirements and consider the properties of the KESM data, I have developed a customized hierarchical volume representation that is easier to extend and has no overhead in using high-level APIs of general purpose data models.

A. Hierarchical Data Representation

Once lateral sectioning is implemented, the amount of the data volumes that can be acquired becomes extremely large. In order to figure out how big it can get, we can calculate an approximate size of the captured images from a tissue block. The typical size of an image is $4,096 \times 9,000$ pixels for the 10X objective, although the length depends on the size of the tissue. The typical length is from $9,000$ to $13,000$ pixels. The pixel depth of a captured image is eight bits in gray scale. Among the $4,096$ pixels, only $3,319$ pixels contain meaningful data, because some of the imaged area is beyond the knife edge. Assuming the number of the columns is eight, the height of the tissue block is 1 cm (10 mm $= 10 \times 1,000\mu$ m), and the cutting thickness is 1μ m, the total size of the block is going to be $3,319$ (image width) $\times 9,000$ (image length) $\times 10,000$ (the number of slices) $\times 8$ (the number of columns) $= 2,389,680,000,000$ which is more than 2 terabytes. The exact formula is as follows:

$$V = W_i \times H_i \times \left(\frac{H_t}{T}\right) \times C \tag{5.1}$$

where V is the tissue volume size, W_i is the image width, H_i is the image height, H_t is the tissue block height, T is the cutting thickness or the tissue ribbon thickness, and C is the number of columns.

It is hard to handle such a large data set with pre-existing tools or techniques. Currently, a 256 or a 512 cubic pixel volumes are being used as data sets to be analyzed, because such amounts of data can fit in the RAM of modern desktop PCs. Figure 15Error! Reference source not found. shows an example from the rat cortex stained with Nissl. This example has only four and 1/5 columns, but it can still show that how much small a 256 or a 512 cube is compared to the whole data set resulting from lateral sectioning. For example, the smaller rectangle to the left represents

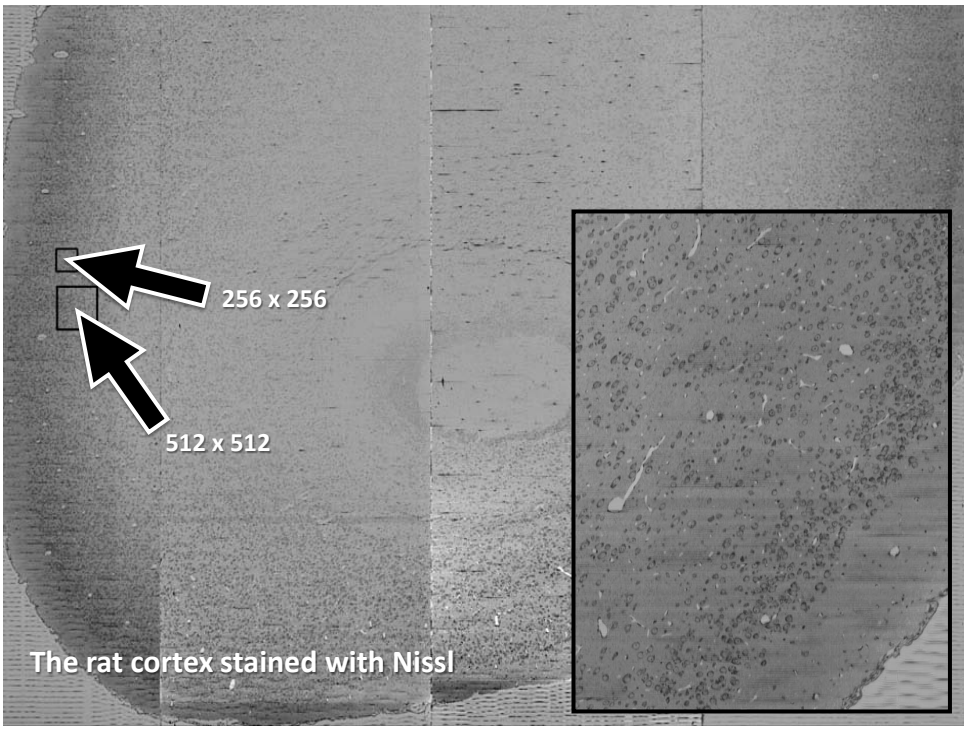


Fig. 37. The Size Comparison Of Data Volumes. The smallest rectangle to the left represents 256 x 256, and the slightly larger rectangle below it represents 512 x 512. The underlying image is the rat cortex stained with Nissl (saggital section).

256×256 and the larger rectangle below it shows 512×512 (see Fig. 37).

In summary, since lateral sectioning technique has been introduced in KESM, it has been generating large amounts of biological data. However, conventional methods cannot be used to handle such data sets because the sheer amount of data easily exceeds the main memory in a typical desktop PC, so a data representation scheme for the KESM data sets has to be developed.

1. Multi-resolution and Hierarchical Representation

To manage such a large data set, I have developed a multiresolution and hierarchical data representation scheme. All images captured by the automated data acquisition

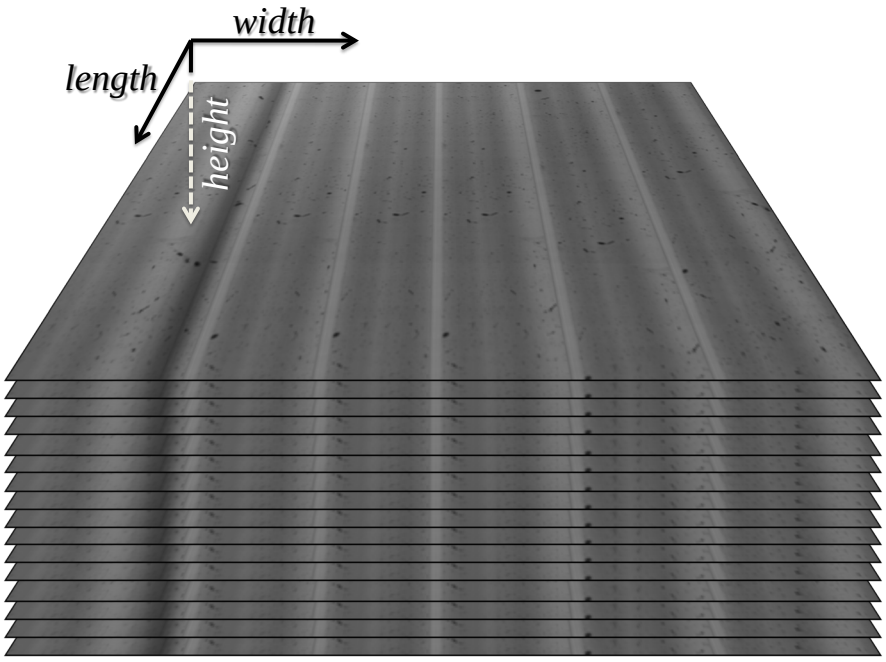


Fig. 38. Whole Image Stack. The width, the height, and the depth of the image stack are defined as shown in the picture.

system are saved in the lossless Tag Image File Format (TIFF) [58]. The raw images are stacked up column-wise after removing unnecessary regions from each image in the column blocks. If misalignment between adjacent column blocks is detected then, column blocks should be rearranged. After correction of the misalignment, the column blocks can be attached side by side. The attachment takes place on an image-by-image basis. Open source libraries, LibTIFF [59] and Image Magick [60] are used to stitch image patches into a large virtual slice. The continuous connections of neighboring image columns result in a *whole image stack* (Fig. 38).

For proper coordination, the *width*, *length*, and *height* in the image stack need to be defined. First, the horizontal direction is the *width* axis. Second, the vertical to the width axis and the direction from the rear to the front is the *length* axis. Last,

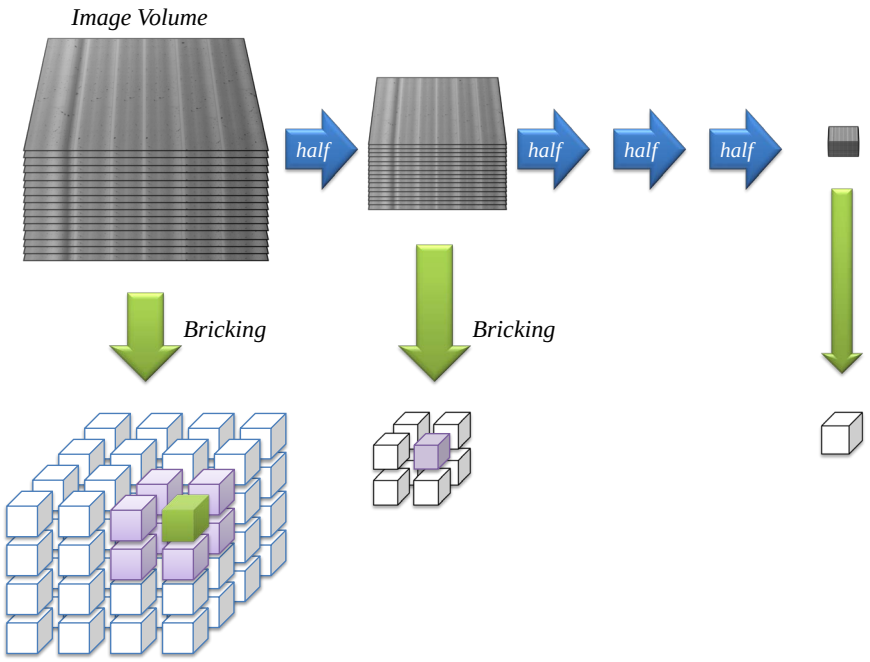


Fig. 39. Multiresolution Volumes And Bricking. The three-dimensional volumetric data set is divided into a unit volume. Also the volume is re-scaled at the half of the original resolution. This process is iterated until the volume size is as same as that of the unit volume.

the orthogonal direction to the width-length plane and toward the depth of the image stack is the *height* axis.

The size of the whole image stack is too large to load into the main memory of a typical desktop PC, so the data should be divided into small *bricks*. This partitioning of data is called *bricking* [61] (see Fig. 39). And a group of bricks is defined as a *brick volume*. On the other hand, the whole image stack is also degraded into a lower resolution volume to give the user a better overview of the data. The bricking and the generation of lower resolution volumes continue until the brick becomes a *unit volume*. The size of a unit volume is automatically calculated based on the tissue block size and the ribbon size to be in between 256 and 512.

The lower resolution image stacks and their brick volumes are saved in a hierarchical directory structure. The specifications of the directory structure are as follows. Each specimen has a unique root data directory, and the images and the bricks will be saved into the separated directories. The name of an image file starts with the resolution field that is followed by the serial number of the image. The name of a brick also starts with the resolution field, and indexes for the width, the length, and the height follow the resolution field.

```
1 / <-- Data Root
2 /Specimen
3 /Specimen/Resolution/
4 /Specimen/Resolution/Images
5 /Specimen/Resolution/Images/ResolutionSerialNumber.tif
6 ...
7 /Specimen/Resolution/Bricks
8 /Specimen/Resolution/Bricks/
    ResolutionIndexWidthIndexLengthIndexHeight.brk
9 ...
```

```
1 Resolution: (
2     1 = actual size |
3     2 = half of 1 |
4     4 = half of 2 |
5     8 = half of 4 |
6     ...
7 )
```

Here is an example of the data directory.

```
1 / <-- Data Root
2 /MouseBrain001
3 /MouseBrain001/1/ <-- actual size
4 /MouseBrain001/1/Images
5 /MouseBrain001/1/Images/1_00000000.tif
6 /MouseBrain001/1/Images/1_00000001.tif
7 /MouseBrain001/1/Images/1_00000002.tif
8 ...
9 /MouseBrain001/1/Bricks
10 /MouseBrain001/1/Bricks/1_0_0_0.brk
11 /MouseBrain001/1/Bricks/1_0_0_1.brk
12 /MouseBrain001/1/Bricks/1_0_0_2.brk
```

```
...
/MouseBrain001/1/Bricks/1_0_1_0.brk
/MouseBrain001/1/Bricks/1_0_1_1.brk
/MouseBrain001/1/Bricks/1_0_1_2.brk
...
/MouseBrain001/2/ <-- half size
/MouseBrain001/2/Images
/MouseBrain001/2/Images/2_00000000.tif
/MouseBrain001/2/Images/2_00000001.tif
/MouseBrain001/2/Images/2_00000002.tif
...
/MouseBrain001/2/Bricks
/MouseBrain001/2/Bricks/2_0_0_0.brk
/MouseBrain001/2/Bricks/2_0_0_1.brk
/MouseBrain001/2/Bricks/2_0_0_2.brk
...
/MouseBrain001/2/Bricks/2_0_1_0.brk
/MouseBrain001/2/Bricks/2_0_1_1.brk
/MouseBrain001/2/Bricks/2_0_1_2.brk
...
/MouseBrain001/4/ <-- quarter size
/MouseBrain001/4/Images
/MouseBrain001/4/Images/4_00000000.tif
/MouseBrain001/4/Images/4_00000001.tif
/MouseBrain001/4/Images/4_00000002.tif
...
/MouseBrain001/4/Bricks
/MouseBrain001/4/Bricks/4_0_0_0.brk
/MouseBrain001/4/Bricks/4_0_0_1.brk
/MouseBrain001/4/Bricks/4_0_0_2.brk
...
/MouseBrain001/4/Bricks/4_0_1_0.brk
/MouseBrain001/4/Bricks/4_0_1_1.brk
/MouseBrain001/4/Bricks/4_0_1_2.brk
...
```

Let us review the data file format for the brick. The brk data file begins with a 20-byte header containing the following information.

```
nIndexBrick:   4 bytes
nSizeWidth:    4 bytes
nSizeLength:   4 bytes
nSizeHeight:   4 bytes
Resolution:    4 bytes
```

The 4-byte *nIndexBrick* indicates the unique id of a brick in the brick volume.

Also it can be used to calculate the position of a brick in the brick volume. The following formula shows how to get the indexes for the width, the length, and the height.

$$nIndexWidth = (nIndexBrick\%nSizeWidthLength)\%nSizeWidth \quad (5.2)$$

$$nIndexLength = (nIndexBrick\%nSizeWidthLength)/nSizeWidth \quad (5.3)$$

$$nIndexHeight = nIndexBrick/nSizeWidthLength \quad (5.4)$$

where $nSizeWidthLength = nSizeWidth \times nSizeLength$.

The 20-byte header is followed by a 32-byte reserved area for the future use. This reserved area will be used when the captured images need additional information such as the color depth (as of now, the KESM produces only 8-bit, 256 gray scale images). The *brk* filename consists of four elements. The first element represents the resolution. 1 means the actual size, 2 means half the actual size, 4 means a quarter of the actual size, and so on. The three numbers after the resolution field represent the position indexes along the width, height, and depth in the resolution volume. For example, the *brk* filename 2_2_3_1.*brk* means that (1) the volume is half the size, (2) the width index is 2, the length index is 3, and the height index is 1. Fig. 40 shows the result of the indexing example.

To summarize, in this section, I described a hierarchical data representation to support an efficient data storage and retrieval scheme for huge data sets. By using the scheme we are able to manage large KESM data. The following section will describe a parallel reconstruction system for extracting three-dimensional objects from the data.

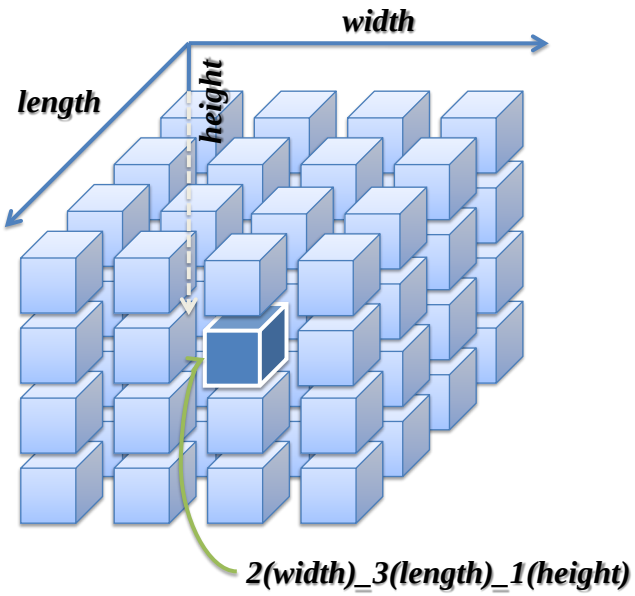


Fig. 40. An Example Of Indexing A Brick.

B. Parallel Processing Framework

In microscopy studies, it is often needed to reconstruct three-dimensional objects from a set of the serial sections. However, it is not easy to find three-dimensional structural information from serial sections. Reconstructing three-dimensional geometric structure from image stacks is another important step in examining and analyzing the morphological properties of neurons and vascular networks from the whole mouse brain. The images acquired through serial sectioning are stacked as figures shown in Fig. 41 (right-top) to form a volume (Fig. 41, right-bottom). Inside the three-dimensional volume, there often are many objects. If the objects can be described by a set of features, then we could get essential information about the geometric structure of the objects from the features. The reconstruction task is to extract the set of features such as geometric structures and shape from the volumetric data set (Fig. 41, left-bottom). The example shows traced blood vessels in India ink-stained tissue. From the reconstructed data, we can measure much essential information about the volumetric data, e.g., the number of branches, the length of a certain fibrous structure, the average radius of the fibrous structure and the number of cell bodies.

There has been much research on reconstructing three-dimensional objects from image stacks. Reconstruct [62] is a free tool for montaging, aligning, and analyzing serial sections. NeuronMorpho [63], a plugin for ImageJ [64], allows the user to measure the neuron morphology. These tools are, however, based on manual work by the user, so it is not useful for large amounts of data from the KESM.

There are also many tracing methods to extract the objects if interest from a tissue volume [65][49], but I will not delve into reconstruction algorithms of three-dimensional objects. Rather, I discuss a parallel reconstruction method where any reconstruction algorithm can be used as long as their output follows the specifications

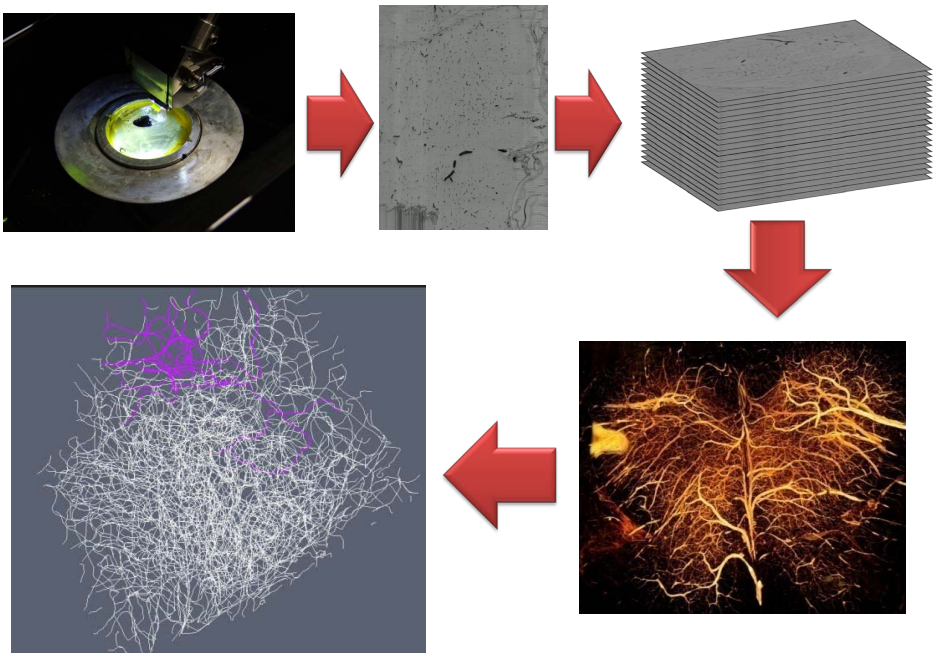


Fig. 41. 3D Volume Reconstruction From Image Stacks. (left-top) Data acquisition. (middle-top) An acquired image. (right-top) Stacked images can be considered as a volumetric data set. (bottom-left) An example of blood vessel from a mouse spinal cord stained with India ink.

in the following chapter.

As I mentioned in earlier sections, the amount of data is unprecedented, so it is not easy to use conventional methods to deal with the data sets. First, let us review a volume data analysis technique, vector tracing, that can trace structures within an image stack such as cell bodies, dendrites, axon fibers, and vasculature. Many existing tracing techniques were developed for two-dimensional images. Some of the methods were extended to work in three-dimensions, but they had limitations in terms of the speed of tracing and the accuracy of the result, because most of the extended solutions from two-dimension used basically the same approach while expanding the search space to three-dimensions. The output of the vector tracing algorithm is a

set of points, the diameters of each point, and connection information of the points. Therefore the output has the following properties.

- Most structures in the data can be reconstructed by using only local structural information, and
- The local results of reconstruction can be combined at a later stage.

These properties allow parallelism to be exploited. The volumetric data will be divided into small blocks and the reconstruction task of each small volume can be performed by a separate processor. Due to the properties of the vector tracing algorithm, the geometric structures from neighboring small blocks can be combined later to build larger blocks. I expect the overhead for combining the small blocks would be negligible compared to the benefit of parallelism, because the type of computations required is either addition or subtraction. In other words, when we combine the outputs from the neighboring blocks, all the new positions of the vertices can be calculated by simply adding or subtracting a value to the original positions.

For the parallel processing model, four major aspects are considered. First, the input is a brick that is a sub-volume from the whole image stack in full resolution. Second, the processing unit is an execution module that runs any object tracing algorithm for three-dimensional objects inside the brick. The format of the output file of the processing unit will be described in the specification of the fiber data file in the following chapter. The server maintains the original image files and the reconstructed results from client nodes. Also the information of the reconstruction status for all the sub-volumes is maintained by the server. Below are the possible states of the server and the clients. The run-time information of each client is managed by the CReconInfo data structure in the server.

```
1  StateClient = { IDLE_OK | IDLE_FAIL | CONNECTED | RECEIVING |
       RUN | SENDING }
2
3  StateServer[numClient] =
4              { IDLE_OK | IDLE_FAIL | CONNECTED | SENDING |
                 WAITING | RECEIVING }
5
6  CReconInfo {
7       bool m_bFinished;
8       LONG m_lStartTime;
9       LONG m_lEndTime;
10      bool m_bSuccess;
11      LONG m_lReconAlgorithmID
12      string m_strResultFileName;
13      LONG m_lResultFileID;
14      string m_strSourceFileName;
15      LONG m_lSourceFileID;
16 }
```

Considering the four aspects above, I designed a parallel reconstruction protocol for a network of PCs (Fig. 42). Two different configurations are possible for the system. The first one is a single server system which consists of one server and multiple client PCs. For the single server configuration a star network was chosen as the network topology. It consists of a central server and several client nodes. There are several advantages of the star network. First of all, the network can be easily scaled up or down by adding or removing nodes. The central server is a repository that saves all the outputs from the clients and combines all the building blocks. The server manages information about the building blocks while client PC nodes reconstruct geometric data from small image volumes. The client PCs do not need to manage information about the whole volume; rather, they simply perform a task that the server has allocated. In other words, the client nodes only need to run reconstruction algorithms to trace geometric structures from the small data volumes they have to take care of. Secondly, the client PCs in star networks do not need to directly communicate with each other. This allows us to use any kind of computer

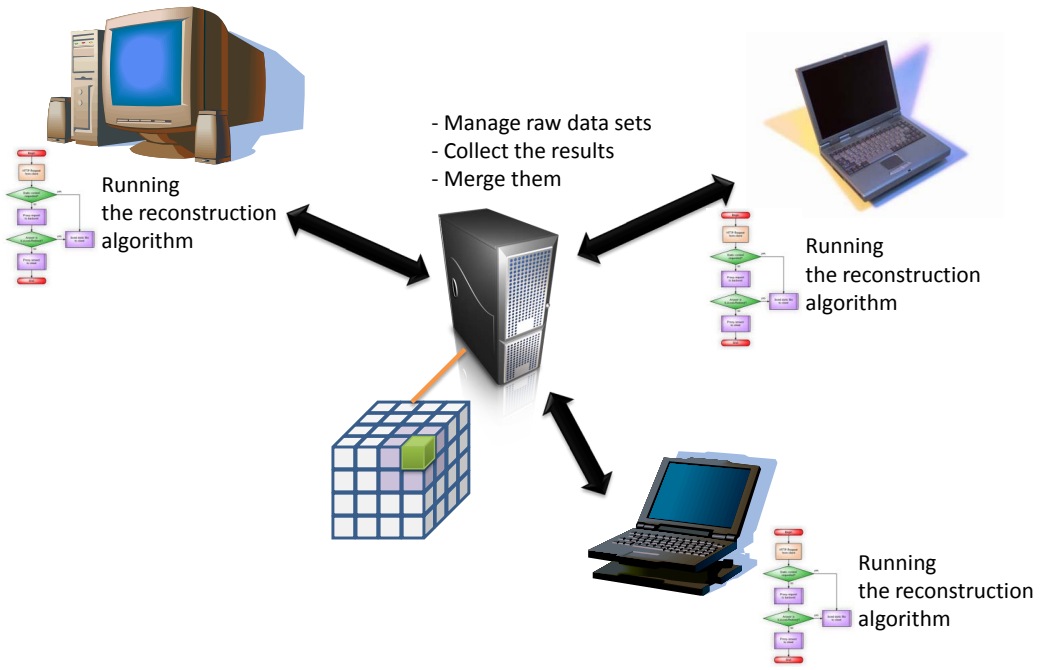


Fig. 42. Single Server Configuration. Star network topology is used to increase robustness of the system. The server manages raw data sets and collects traced results from clients.

system that can run the reconstruction algorithm for the client PCs.

The single server system can be easily extended to a multi-server system. If the total amount of data is too large to be handled by a single server or the reconstruction task needs to be finished earlier, then a multi-server configuration can be considered. A meta server should be introduced in the multi-server configuration which can be considered as a double layer star network topology. A meta server is a central server of the other server nodes, and each server node is also a central server for other clients. The main difference between the single server and the multi-server configuration is that the meta server does not have the raw image data. Instead, it collects all the reconstructed geometric data set from the sub-servers. In most cases, the raw data

Fig. 43. Mutl-server Configuration. The main server has multiple sub-server and each sub-server manages client connections.

sets are extremely large (tens of terabytes), so it is not practical for a meta-server to hold a copy of all the raw image data (Fig. 43).

In order to guarantee data transfer reliability between the server and the client PCs, I decided to use a secure File Transfer Protocol (sFTP) rather than using a custom data transfer protocol. The file size transmitted through the network is between 16 MB (8-bit gray scale image in $256 \times 256 \times 256$ cube) to 128 MB ($512 \times 512 \times 512$). Besides the raw transmission, the server shares information with the client PCs via KESM Parallel Reconstruction Protocol (KPRP, Fig. 44).

The server needs to know what the clients are doing at a given moment and what they are going to do next. To do this, the server manages the task control blocks of the clients, but does not directly control the clients. The server acts passively, and

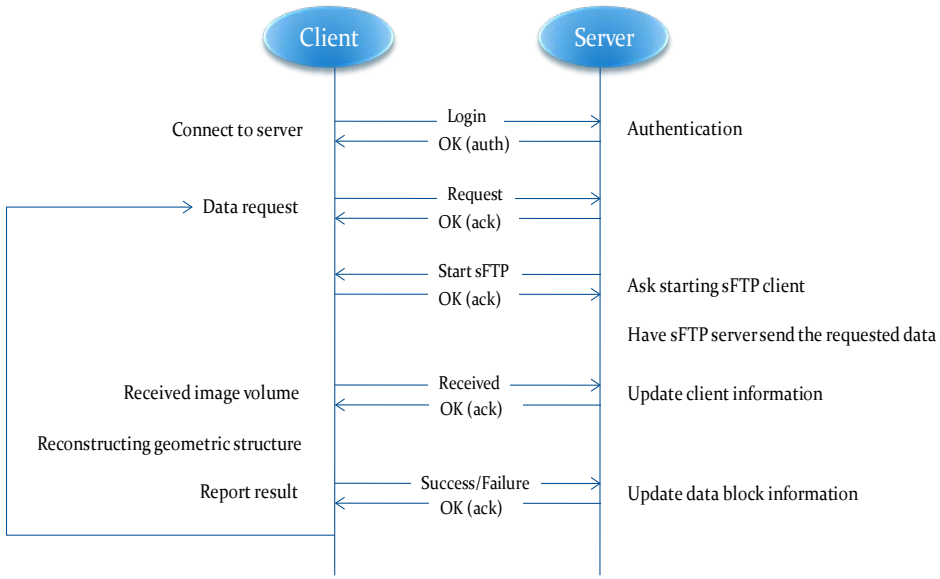


Fig. 44. KESM Parallel Reconstruction Protocol.

the clients have initiatives in the reconstruction process. In some sense, the server simply waits until the clients send requests or report their status, and does not direct the clients run the reconstruction algorithm. The clients are supposed to ask for a data block to the server, run the reconstruction algorithm, and transmit the result to the server. The server collects the reconstructed geometric data, and maintains the overall process. This makes the system simple to maintain. The advantages of the present system design are as follows.

- Minimum communication: the server does not directly control the clients. The clients do not rely on the server when they perform their own tasks; rather, they just notify their status and task information to the server.
- Robustness: when some of the client PCs crash, it simply means to the server that some of the tasks allocated to a client node cannot be performed within a certain amount of time. The server will reallocate the task to another client as another client asks for a new task.
- Simplicity and scalability: if more client PCs are connected to the server, it simply improves the performance without increasing a significant management overhead for the server.

Fig. 45 shows a screen-shot of the parallel reconstruction framework.

I discussed a parallel processing framework in this section. Due to large amount of data sets from KESM, it is a big challenge to reconstruct whole three-dimensional geometric structure from image stacks. A parallel processing framework was introduced to reduce the time. Following sections are about

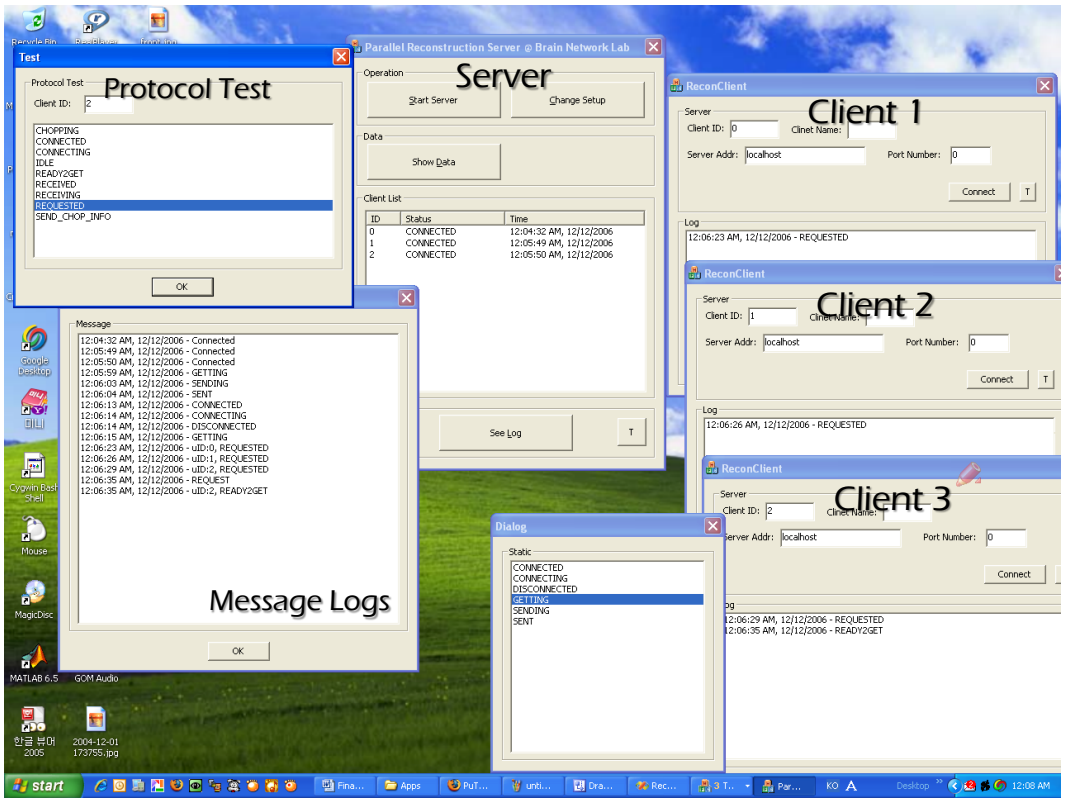


Fig. 45. KESM Parallel Reconstruction.

C. Unit Volume

A *Unit Volume* is defined as a basic processing block for visualization and analysis. In order to visualize volumetric data sets, we need to extract a Region of Interest (ROI) since the entire volume is too large to load into the system memory of a modern computer.

The size of the unit volume is calculated automatically based on the whole tissue block size. In most current computer system, the cubic size should be less than $512 \times 512 \times 512$. The width, the length, and the height are decided to be in between 256 and 512. The width is defined as the size of the horizontal direction (Eq. 5.5), the length means the size of the vertical dimension inside an image (Eq. 5.6), and the height represents the *height* of the volume (Eq. 5.7).

$$UnitWidth = \frac{(W_R \times C)}{2^w} \tag{5.5}$$

$$UnitLength = \frac{L_R}{2^l} \tag{5.6}$$

$$UnitHeight = \frac{\max(H_R(c))}{2^h} \tag{5.7}$$

where W_R is the tissue ribbon width, C is the number of columes, L_R is the ribbon length, $H_R(c)$ is a height of the image column stack indexed by c, and w, l, h are the smallest numbers that make the $UnitWidth$, $UnitLength$, and $UnitHeight$ are between 256 and 512.

$$Col = \frac{\text{floor}(UnitWidth \times W_i)}{R} \tag{5.8}$$

$$StartXinCol = R - (R \times (Col + 1) - W_i \times UnitWidth) \tag{5.9}$$

where Col is a column index, R is the width of a ribbon, W_i is the index for the *width*

direction, and $StartXinCol$ is the start position in the column.

Once we have the Col and the $StarXinCol$ values (Eq. 5.8 and 5.9, then we are ready to extract image regions from each image in the image stacks. Col tells us which folder should be explored (note that images are located in separate folders based on the column number). The image file names in each folder are based on the *height* index. So it is trivial to get the right image file in the folder. $StartXinCol$, as the name implies, indicates the start position for extracting a region from the image. If the extracted region is smaller than that of the unit volume, then the algorithm continues to extract a region in the image in the $Col + 1$ folder. This process is continued until the size of the extracted region is the same as that of the unit volume.

Here is an example for better illustration:

```
1 The number of columns: 8
2 The tissue ribbon width: 2,400
3 The total width of the tissue block: 19,200
4 The maximum height of image stack columns: 5,573
```

Suppose the scale is 1/4 and the index of the width, $iWidth$ is 5. First, the tissue ribbon width, 600 ($= 2,400/2^2$) is calculated. The automatically calculated width of the unit volume is 304 ($= 38 \times 8$) in this example. 38 is the width of a ribbon in the smallest scale (in this case, 1/32) and 8 means the number of columns. Using Eq. 5.8, we can get $Col =$ is 2 ($=$ floor$((304 \times 5)/600)$. $StartXinCol$ is 320 ($= 600 - (600 \times 3 - 5 \times 304)$ (Eq. 5.9). In column number 2, we can get a region whose width is 280 which is less than the unit width, so we need to move on to the next column. After extracting a region whose width is 24 in the next column, these regions which are extracted from separated images should be *stitched* into an image. Fig. 46 illustrates these steps. This process is iterated until to collect all image pieces across columns. A unit volume is made by stacking up unit images.

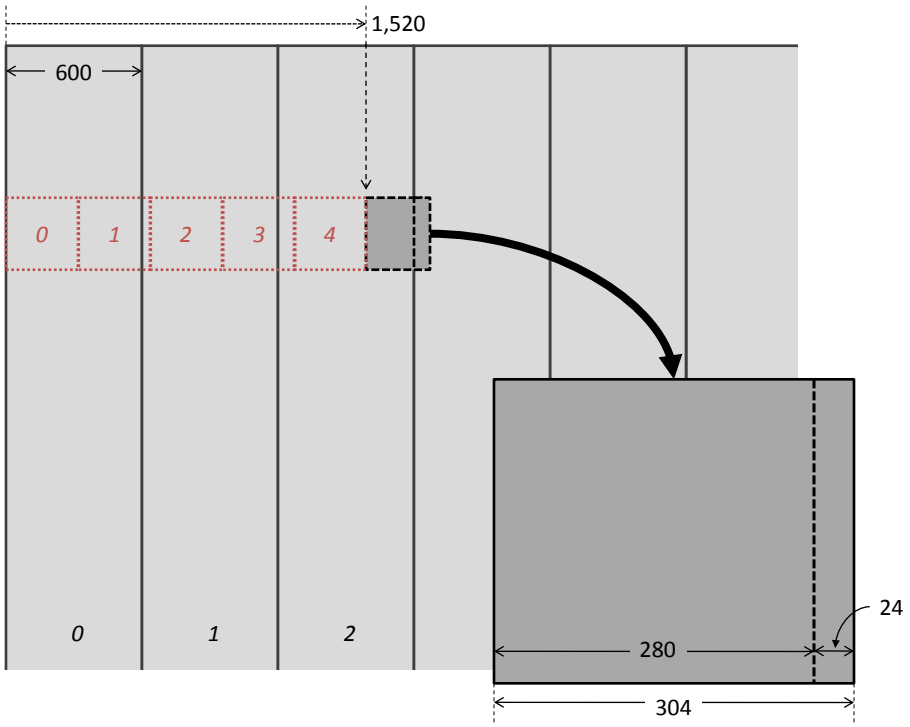


Fig. 46. An Example Of Extracting A Unit Image For A Unit Volume.

To summarize, I discussed data representation methods to manage large amount of data sets in the first section of this chapter. To address the challenge, I described a multi-resolution and hierarchical data representation method, and parallel processing framework to speed up the processing speed. The last section of this chapter was devoted to an algorithm for extracting a unit volume from the multi-resolution and hierarchical data representation.

In the following chapter, I describe visualization, image segmentation, and data representation for the fibrous three-dimensional data structure.

CHAPTER VI

INTERACTIVE VISUALIZATION

Scientific visualization [66][67] provides methods that allow us not only to explore data, but also to test hypotheses based on measurements or simulations. Visualization of KESM data is problematic in many aspects [68]. First, the volume of the data is too large for a typical desktop computer to be handled. Second, the anatomical entities may be densely packed and branches of dendrites often intertwined in complex ways. In this context, simply isolating a single structure from such a data set could be difficult. Interaction is also an important element in scientific visualization, because interactivity helps the user interpret and understand the data sets. Interaction methods should allow the user navigate the data by selecting a Region of Interest (ROI), or object of interest.

Through previous chapters, I showed how large volumetric data sets can be successfully acquired, and processed, ready to be analyzed. To handle and examine the large biological data sets, interactivity is an important factor. For example, to grasp an idea about overall structure, seeing the whole block is an inevitable function. However, it is almost infeasible to load all the image data, process them, and explore inner structure, because the large amounts of the data often exceed the system resource of the desktop PCs. To investigate the data sets, the user should be able to select a portion of the data to examine it in more detail. Annotation is also an important function for researchers to understand the data sets. Experts in a certain area can help other researchers or share their interpretation of the data sets through shareable annotations.

From the reconstruction method proposed in the previous chapters, I am able to provide volumetric data sets as well as geometrical structure describing the mor-

phology of the object in the data sets. The reconstructed data sets may shed light on understanding the data, because we can get information about connectivity of fibrous structure and length or thickness of the fibers through the reconstructed topological and morphological data sets. This information is important especially to researchers who are interested in neuronal processes, because it is commonly known that the morphological properties of biological entities lead to difference in functions.

A. Data Visualization

A three-dimensional data viewer for volumetric data gives us a better overview, but distortion due to perspective in three-dimensional objects may be inevitable. On the other hand, a two-dimensional image viewer can provide precise and detailed information about an image, but it is hard to figure out the overall structure in the volumetric data sets by viewing the images sequentially. In this context, a combination of two-dimensional and three-dimensional visualization is needed to build an effective scientific visualization system.

Let us consider about a two-dimensional image viewer first. The user should be able to examine a portion of a data set more accurately, and for this two-dimensional image viewer with zoom in and zoom out function needs to be used. The KESM Two-dimensional Image Viewer (K2IV) (Fig. 47)can be considered as a *virtual microscopy* [69] which shows image slides and helps researchers analyze neuroanatomical data sets. The two-dimensional image viewer is a sub-module for the KESM Data Visualization System (KDVS).

KESM data sets often contain densely packed fibrous structures, and because of this, it is hard to explore the data sets by only a two-dimensional image viewer. Volume rendering is a better approach to get an overview of the data sets and it gives

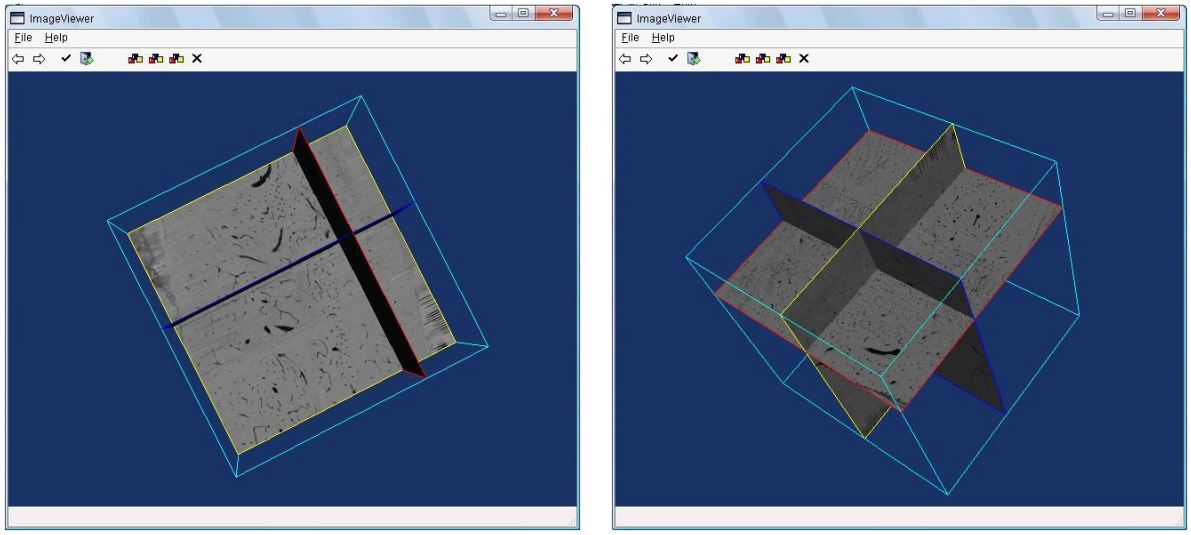


Fig. 47. KESM Two-Dimensional Image Viewer (K2IV). Users can rotate the volume to see other surfaces.

us a more comprehensible picture. Through the volume visualization tool or three-dimensional data viewer, a user can more easily identify neurons, their connectivity, and microcirculation. A three-dimensional data viewer is a rendering tool, so the image stacks are displayed in a volumetric form that is semi-transparent to show the internal structures in the volume. The three-dimensional volumetric data viewer can be used as a building block for integrated tools like the two-dimensional image viewer, K2IV. Here, I have implemented a three-dimensional image volume visualization tool and it will also be extended to KESM Three-dimensional Volume Renderer (K3VR) which will be integrated into the KDVS (Fig. 48).

Efficient navigation will require multiple resolutions of the data. In order to increase responsiveness, as I described in the previous sections, data sets will be pre-generated at different zoom levels, and stored to a database as scaled-down images. When a user is looking at a large portion of the data, a lower resolution data set will be presented. If the user wants to see more specific features from a small portion of the data, higher resolution data would be presented.

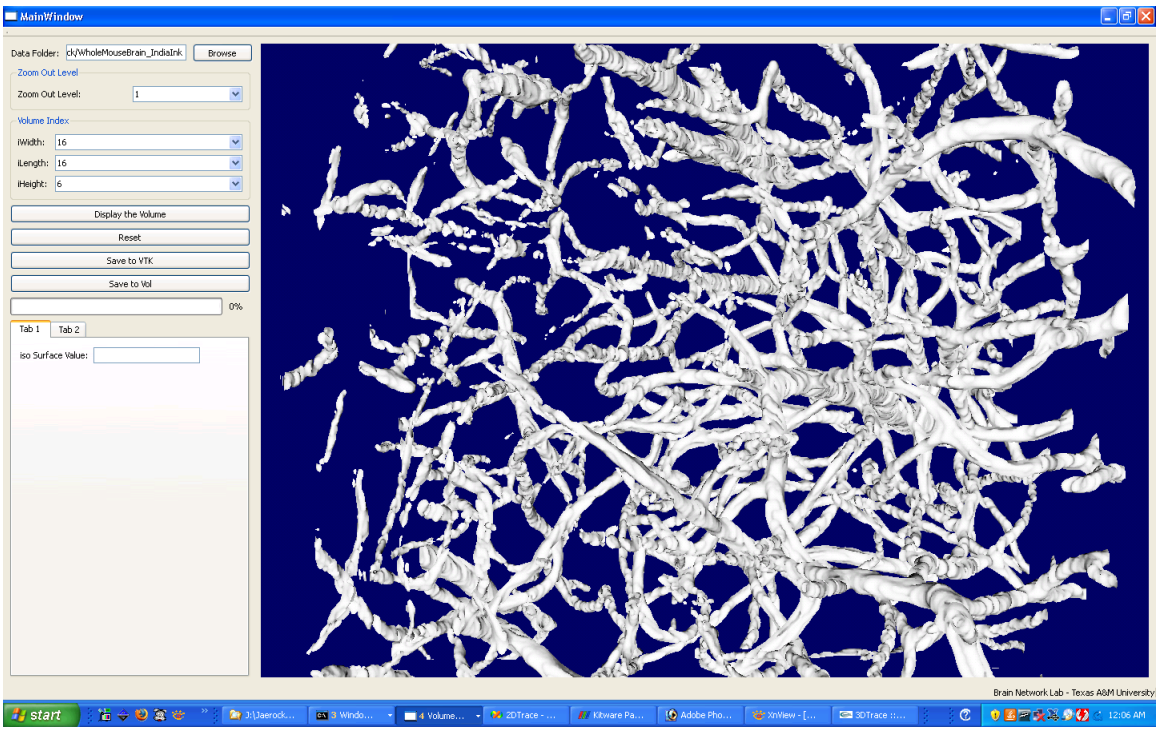


Fig. 48. KESM Three-dimensional Volume Renderer (K3VR). The volume renderer is integrated to the data representation method. The user can choose scale level and indexes of the unit volume to display.

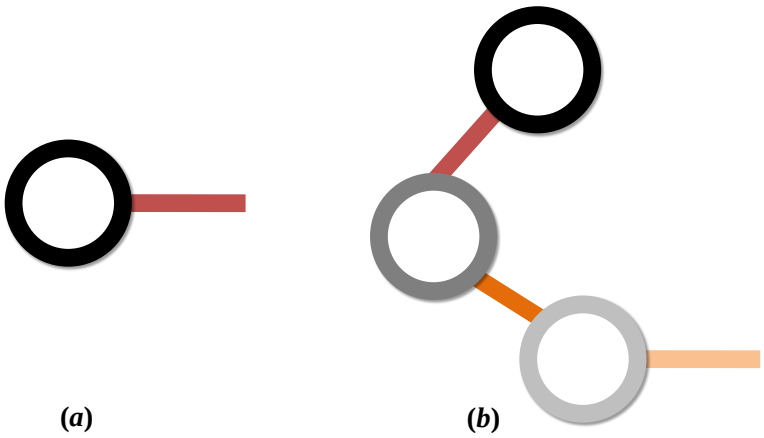


Fig. 49. Basic Elements of Fibrous Data. (a) Node segment. It consists of a node pointer and a set of points led to the next node. (b) Line. A line can be represented by a group of node segments.

B. Fibrous Data Representation

As part of the visualization system, an output file format has been developed. Eventually the geometric data reconstructed from the parallel reconstruction system will be visualized with either the two-dimensional image viewer or the three-dimensional volume renderer to help the user understand the data. Therefore, the visualization system needs to understand the data file format for the reconstructed geometric data.

1. Data Representation

After three-dimensional segmentation, the traced structure needs to be stored in a format that can be used in analysis of the data sets. Fibrous data can be represented by a group of line segments and their connections.

A *node segment* is defined as the basic elements of the fibrous structure as shown in Fig. 49 (a). A line can be described by a group of connected node points (see Fig. 49 (b)). By using the basic elements, traced fibrous structure can be stored in a

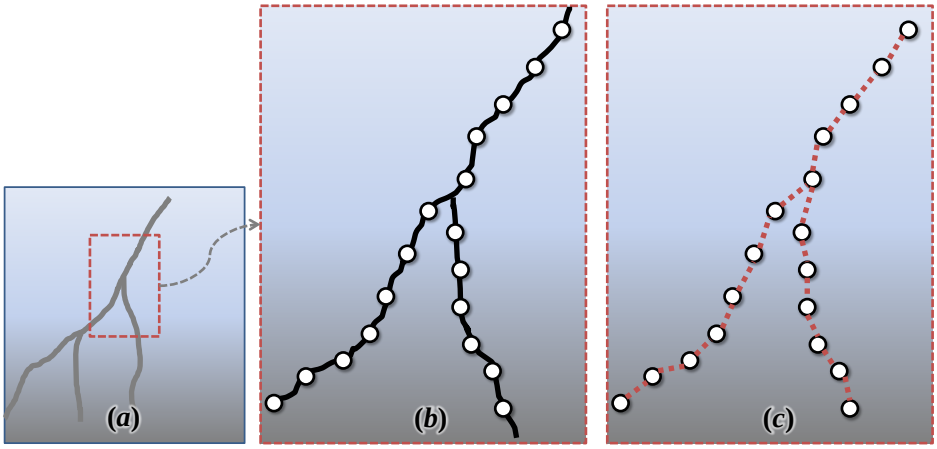


Fig. 50. Traced Data Representation. (a) An example of traced structure. (b) Traced points are represented by groups of *node points*. (c) All points in the traced structure are represented in a form of connected node segments.

certain format. Fig. 50 shows a conceptual illustration for the traced data. The traced three-dimensional structure can be seen as groups of connected line segments (see Fig. 50 (b)). The segment in the traced structure is simplified to the *node segment*. Thus, as Fig. 50 (c) shows, all points in the traced structure are represented in a form of connected node segments.

2. Fibrous Data File Format

The data format consists of two major sections. First, all the data points are described. Second, they are followed by the details of connections between the points.

```
1 NumPoints
2 index0    x0    y0    z0    r0
3 index1    x1    y1    z1    r1
4 index2    x2    y2    z2    r2
5 ...
```

The data file starts with an integer number that represents the total number of points (vertices). In the format, index$n$ is an integer label that identifies each point.

xn, yn, and zn are the coordinates of the point and rn is the radius of the point. The radius of the point is measured from the cross section when two neighboring points are connected by an edge, where the first point is pointing toward the second point. The second part of the data file has information about the conductivities of the points.

```
NumLines
NumEdges
index_n     index_m
index_m     index_l
index_l     index_o
...
NumEdges
index_p     index_q
index_q     index_r
index_r     index_s
...
```

Below is a simple example to illustrate the data file format. Fig.51 shows the connectivity based on the example below.

```
12  <-- the number of points
1     x1     y1     z1     r1
2     x2     y2     z2     r2
3     x3     y3     z3     r3
4     x4     y4     z4     r4
5     x5     y5     z5     r5
6     x6     y6     z6     r6
7     x7     y7     z7     r7
8     x8     y8     z8     r8
9     x9     y9     z9     r9
10    x10    y10    z10    r10
11    x11    y11    z11    r11
12    x12    y1     z12    r12
2   <-- the number of lines
8   <-- the number of edges consisting the line
1     2
2     3
3     4
4     5
5     6
6     7
7     8
4   <-- the number of edges consisting the line
```

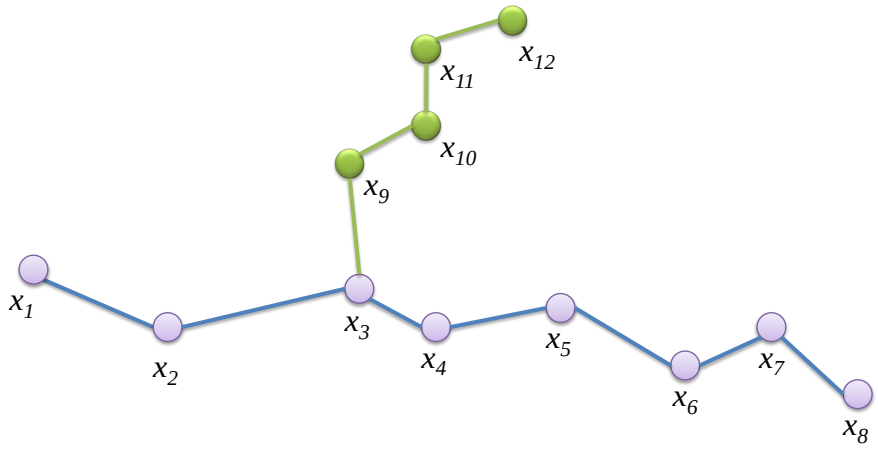


Fig. 51. An Example Of Fibrous Data.

24	12	11
25	11	10
26	10	9
27	9	2

Once the image volumes are described in geometric information, we can now get a statistics of the geometrical structures. In the example above, $x3$ is a branch point (Fig. 51).

I followed the information seeking principle *overview first, zoom and filter, and then details-on-demand* [53] to visualize the data sets. Large-scale high-resolution neuroanatomical data sets often cannot be displayed because of their immense size, but to grasp an overview of the data, rendering the data at a low resolution at first may be a good approach. When a user navigates the data and wants to see further details, the visualization system should retrieve relevant portions of the data and display them at a higher resolution.

Two-dimensional image visualization allows a user to see the image more precisely, whereas volume rendering provides us a better overview. When we navigate

the data sets while changing the view angle, it gives us various points of view so that we can more easily identify structures such as branch points in closely packed fibrous structures. I have developed a two-dimensional image viewer and a three-dimensional volume renderer to provide a better way for a user to understand the image data sets.

As I described in the previous sections, the amount of the data sets is extremely large, so I introduced the KESM Hierarchical Data Clustering (KHDC) scheme in a previous chapter. The data sets are supposed to be divided into many small sub-volumes. Also, MIP map (multum in parvo, meaning much in a small space) volumes are generated for visualization in KHDC. The geometric structure inside a sub-volume will also be traced. Therefore, we will have numerous volumetric blocks of the raw images, and the traced data sets from those blocks. These viewers that I developed here can deal with only an image volume without any context of the whole data set. A geometric viewer is also needed to show a user the traced data. The hybrid of a raw image viewer and a geometric viewer is a good approach to provide a user with an enhanced ability to explore the data sets.

For the implementation of the visualization system, I used the Visualization Toolkit (VTK) [51] and Insight Segmentation and Registration Toolkit (ITK) [50], an open source library for three-dimensional visualization. The VTK has been used for the visualization part and ITK for the image processing part. The user interface has been implemented in Qt [52] which is a cross platform GUI framework. The following chapter is devoted to data analysis and results of it.

CHAPTER VII

DATA ANALYSIS

Efficient data analysis methods for large volumes of data sets play an important role in neuroscience study. Standard procedure in experimental neuroscience study is as follows. Possible situations such as aging are applied to a tissue, and the tissue is compared with a control tissue. In order to compare the test tissue with the control tissue, methods for measuring three-dimensional structural information are necessary. Eventually the findings from the comparison can refine computational models (see Fig. 52).

A. Data Analysis

Efforts of my colleagues at BNL and I allowed us to scan two whole mouse brains stained with Golgi and with India Ink, respectively. In this chapter, I describe some results of data analysis from one of our data sets. For this analysis, I used a processed three-dimensional volumetric data sets of a mouse brain specimen stained by India ink.

Dotted square boxes in Fig. 53 indicate volume areas analyzed here. One box stands for four adjacent $225\mu m \times 262.5\mu m \times 290\mu m$ cubic volume. I intentionally chose three different regions, specifically the olfactory bulb, the cerebral cortex, and the cerebellum, to see statistical difference between those areas.

As iso-surface visualizations in Fig. 54 (a), 55 (a), and 56 (a) show, the structural differences cannot be easily observed by just looking at the three-dimensional visualization. Before starting data analysis, the tracing method that has been used needs to be discussed. To trace fibrous microstructure in three-dimensional volumetric data sets, I used a model-based method [72] implemented for [73]. Fiber segments are

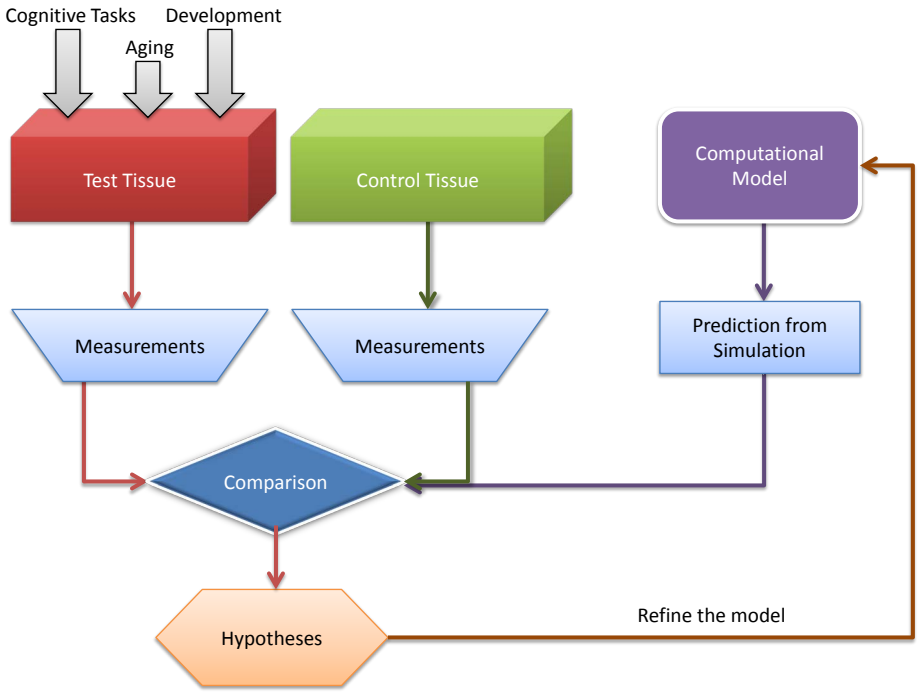


Fig. 52. Possible Scenario For Data Analysis. Possible experimental conditions can be applied to a test tissue. Efficient ways of Data analysis allows us to compare the test tissue and the control tissue.

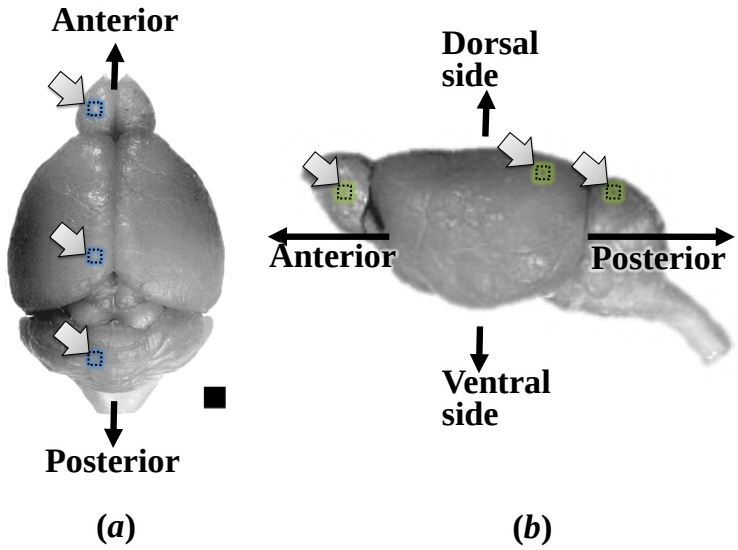
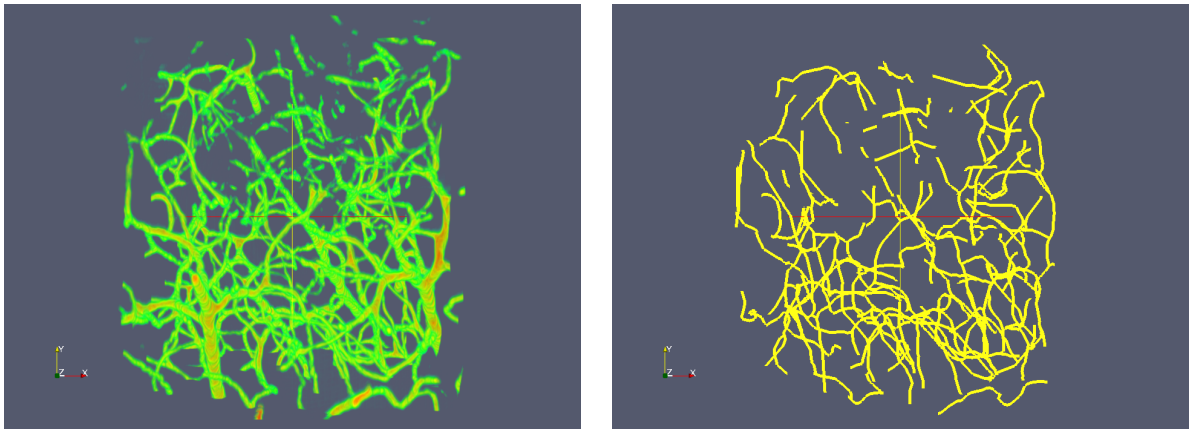
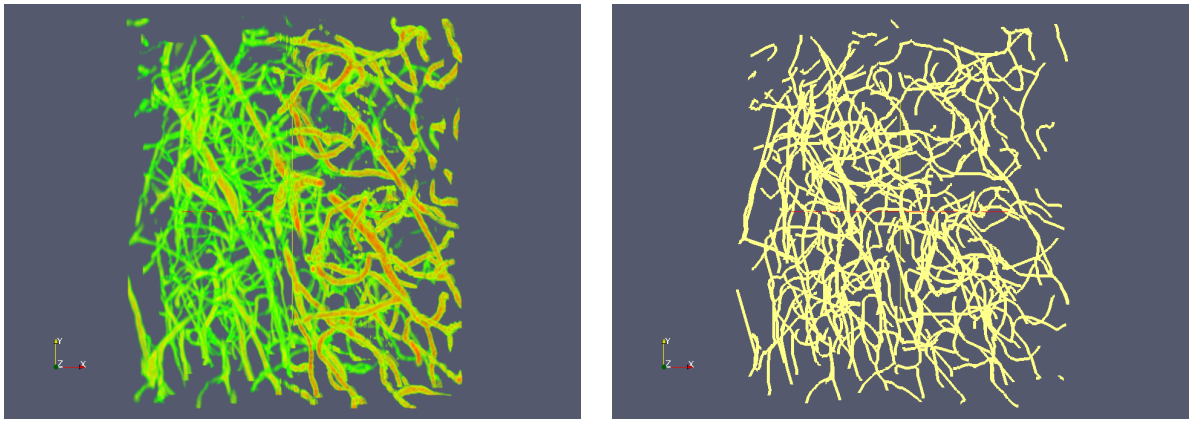


Fig. 53. Mouse Brain. Dotted square boxes (gray arrows) indicate the analyzed areas. (a) Dorsal view of the mouse brain. The black rectangle is a scale bar, 1 mm. Mouse brain image adapted from [70]. (b) Sagittal view of a mouse brain. Mouse brain image adapted from [71].



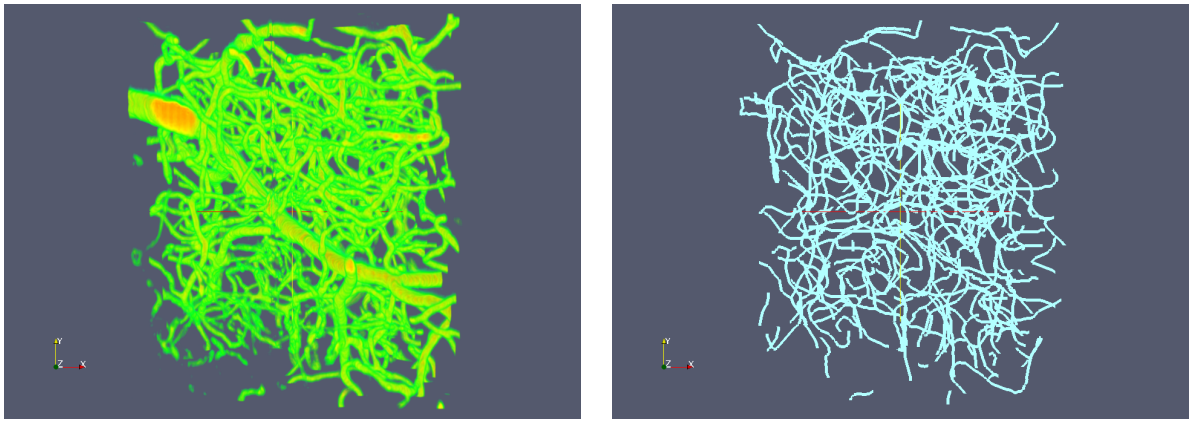
(a) (b)

Fig. 54. A Unit Volume From The Olfactory Bulb. (a) Iso-surface visualization (b) Traced result.



(a) (b)

Fig. 55. A Unit Volume From The Cerebral Cortex. (a) Iso-surface visualization (b) Traced result.



(a) (b)

Fig. 56. A Unit Volume From The Cerebellum. (a) Iso-surface visualization (b) Traced result.

modeled with a central axis curve coupled to a fiber wall surface. Many approaches to segment structure often use a single-scale analysis that can only apply for a range of width. This model-based method uses multi-scale algorithms to cope with fiber width variations.

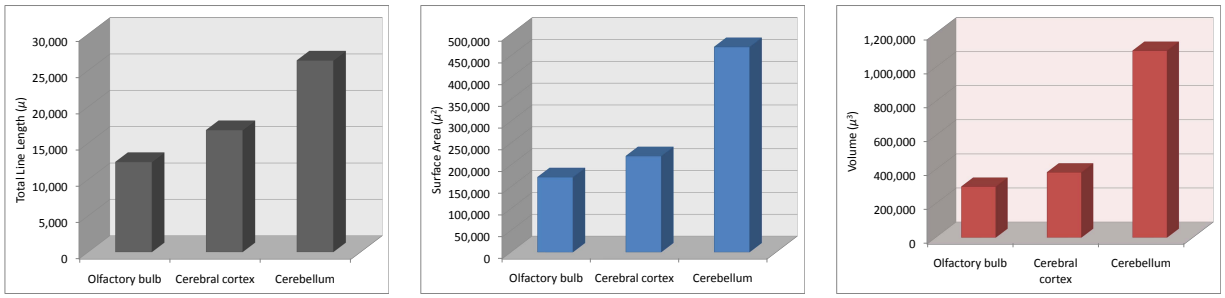
The detail of the implementation of this model-based tracing method can be found in [72]. My main contributions to the tracing implementation is focused on the automation of the tracing for multiple volumetric data sets and the development of a fibrous data file format.

Due to the fibrous data representation framework, statistical information can be automatically calculated since the data structure has all the necessary pieces of information.

B. Results

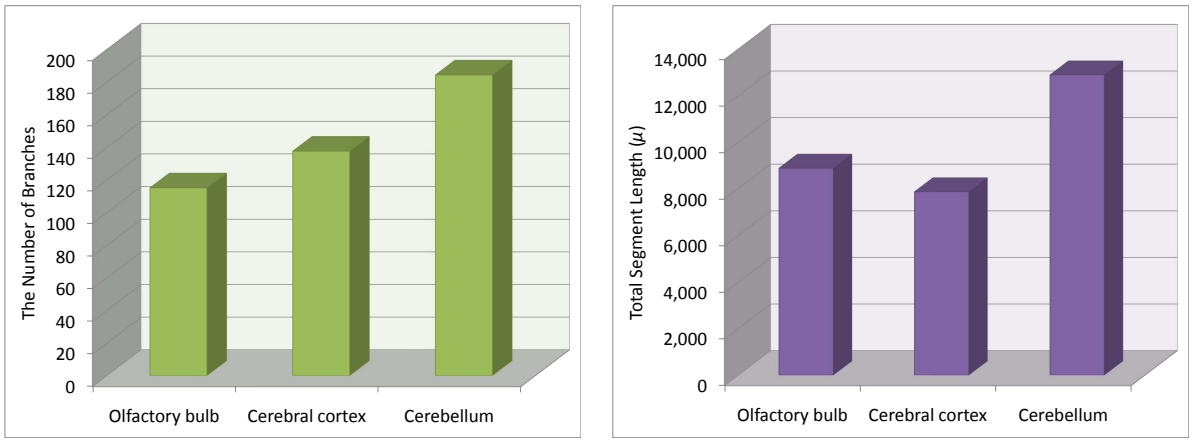
We can compare the three different regions through automatic statistical analysis. A *segment* is defined as an edge between two neighboring branches. Fig. 57 shows the comparison between the three different regions: the olfactory bulb, the cerebral cortex, and the cerebellum. The total length of fibrous structure in an area of the cerebellum is longer than that of the two other areas. This may mean that the fibrous structure in the cerebellum is most densely packed. The number of branches of the area in the cerebellum is also greater than that of others (see Fig. 58). This graph means that the fibrous structure in the cerebellum is not only densely packed but also the most interwoven.

The automatic data analysis framework generates all radii in the traced structures to see the distribution of the data. By looking over Fig. 59 (a), we can tell that there are thicker fibrous structures in the area of the cerebellum (see the bars around



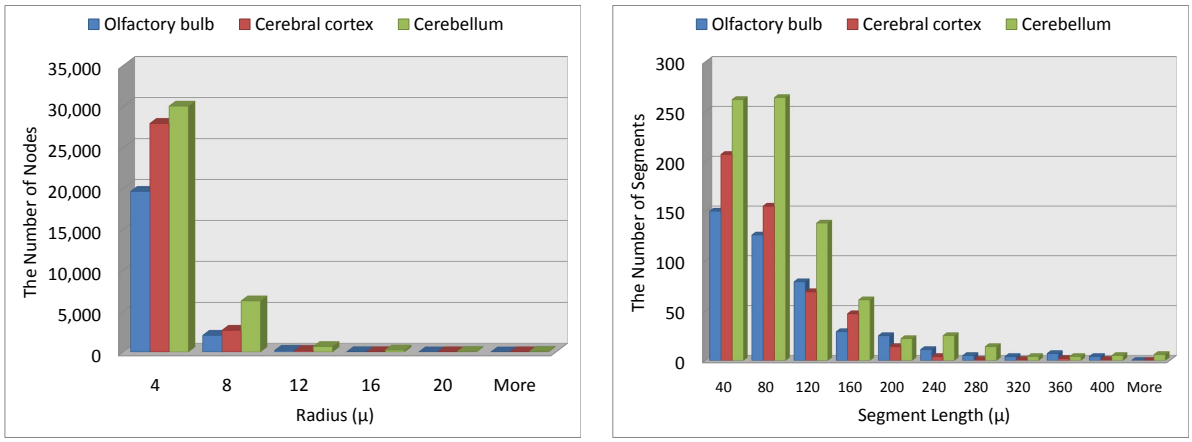
(a) (b) (c)

Fig. 57. Comparison between the Three Different Regions. (a) The total length of fiber structure is shown. The length in the cerebellum is the longest among the three. (b) Surface areas between the three areas. (c) The volume size of fiber structure.



(a) (b)

Fig. 58. Number Of Branches And Total Segment Length. (a) The number of branches in the area of the cerebellum is the largest among the three. (b) The total length of segments. The length of the segments in the areas of the olfactory bulb is longer than that of the cerebral cortex.



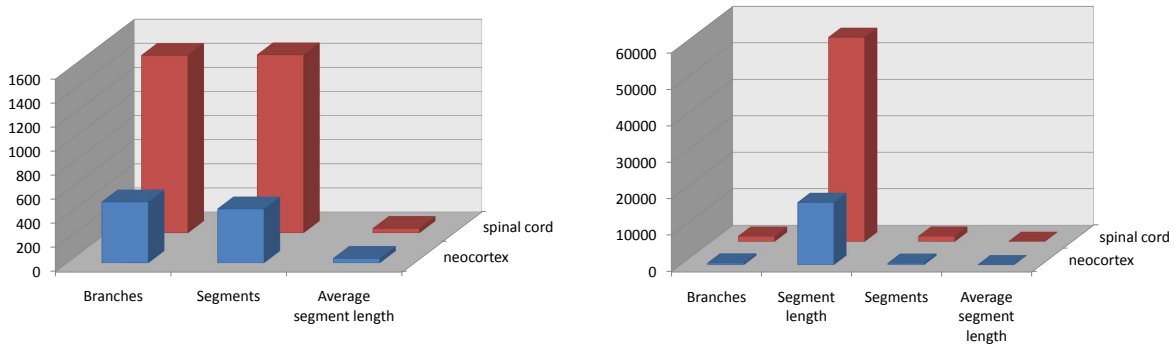
(a) (b)

Fig. 59. Distributions Of Radius and Segment. (a) Histogram of all radii. (b) Histogram of the numbers of segments. Relatively, the cerebellum shows longer distances between two branch points.

radius $12\mu m$). The segment histogram (Fig. 59 (b)) shows that there are longer segments in the area from the cerebellum since small segment lengths are dominant in the areas from the olfactory bulb and the cerebral cortex while the segment lengths up to $80\mu m$ are evenly distributed in the areas from the cerebellum.

Below is another example of the potential usefulness of structural information of tissue samples from the reconstruction system. The two following automated quantitative analyses are the structural information from small blocks ($256 \times 256 \times 256$ cube) from the mouse neocortex and the mouse spinal cord.

The number of branches and segments (Fig. 60 (a)), and the length of segments (Fig. 60 (b)) in the vascular network in the spinal cord are almost three times greater than those in the neocortex. The average length between neighboring branches is almost the same (Fig. 60). One interesting thing is that the surface area and the total volume between the spinal cord and the neocortex samples are significantly different. In other words, the blood vessels in the spinal cord sample seem much



(a) (b)

Fig. 60. Comparison Between The Spinal Cord And The Neocortex Samples. (a) The number of branches and segments. (b) The length of segments.

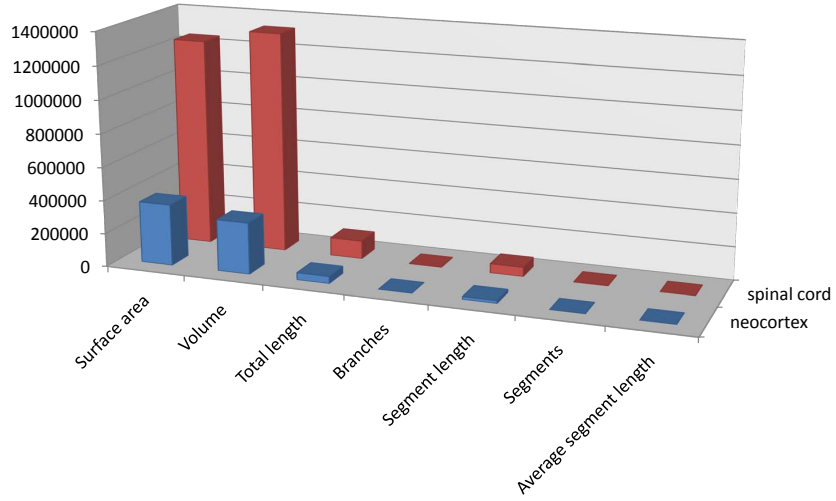


Fig. 61. The Comparison Of Surface Areas And Total Volumes.

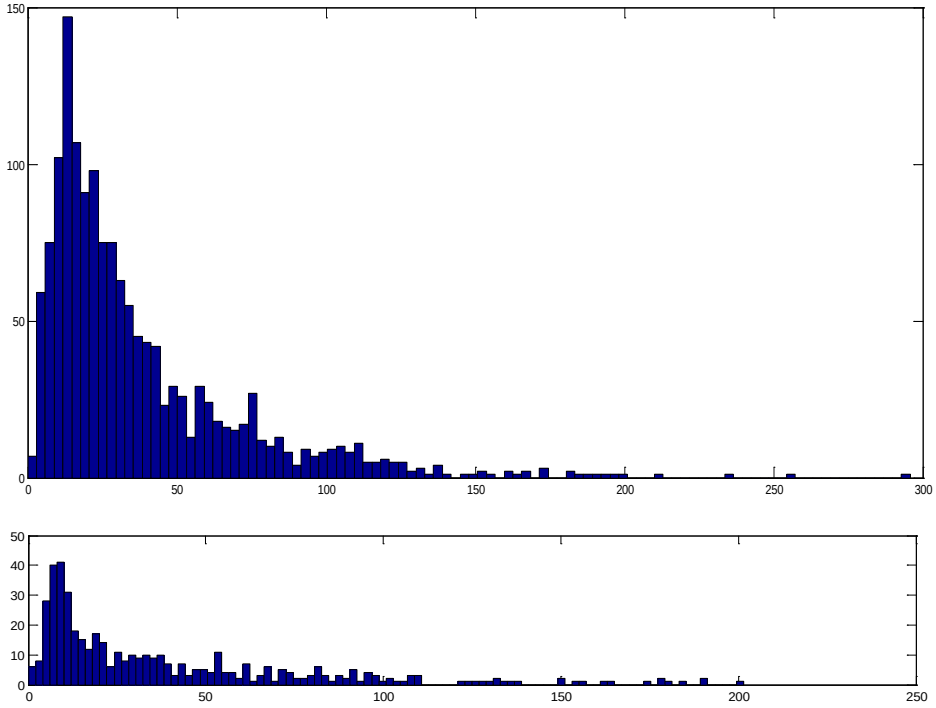


Fig. 62. Distributions Of Segment Lengths In Blood Vessels. (top) an example from the mouse spinal cord. (bottom) an example from the mouse neocortex.

thicker than those in the neocortex sample (Fig. 61). Also the distributions of the lengths of segments are quite different (Fig. 62).

In this chapter, I discussed the utility of the automatic data analysis tool that allows us to open a new opportunity to analyze large amounts of volumetric data sets. In the following, I will describe a software framework for acquisition and mining of microstructure in the brain.

CHAPTER VIII

SOFTWARE FRAMEWORK

Throughout previous sections, I introduced the KESM Image Acquisition System (KIAS), the KESM Image Capturer System (KICS), the 2D Image Viewer (K2IV), the 3D Volume Render (K3VR), and a framework for parallel data reconstruction (KESM Hierarchical Data Representation [KHDR] and KESM Parallel Data Reconstruction System [KPDR]). Here, I present an integrated tool by combining and extending the components listed above. Fig. 63 shows an overall organization of the system.

A. Overall Structure

The software framework has several parts. First, the KESM Image Acquisition System (KIAS) mainly consists of the KESM Image Capturer (KIC), the KESM Stage Controller (KSC), the KESM Stair-Step Controller (KSSC), and the KESM Session Manager (KSM). The data acquisition system, KDAS, was designed and implemented especially for getting large data sets from small animal organs such as the brain, kidney, or spinal cord. In order to expand the scan area beyond the width of the fixed Field of View (FOV), I have developed a lateral sectioning algorithm with my colleagues in the Brain Networks Laboratory (BNL). Sectioning and imaging of a tissue block often takes more than a week, and sometimes more than a month depending on the size of a tissue block. Even though the acquisition system runs by itself after a user set all the parameters, it still needs human intervention. Therefore, the ability to resume sessions is essential where a user restarts the operation at the same condition from the previous session. Suppose that a user stopped the data acquisition operation one day and then and the user needs to resume the task later. The stage should be positioned at exactly the same place where the previous sectioning was being done.

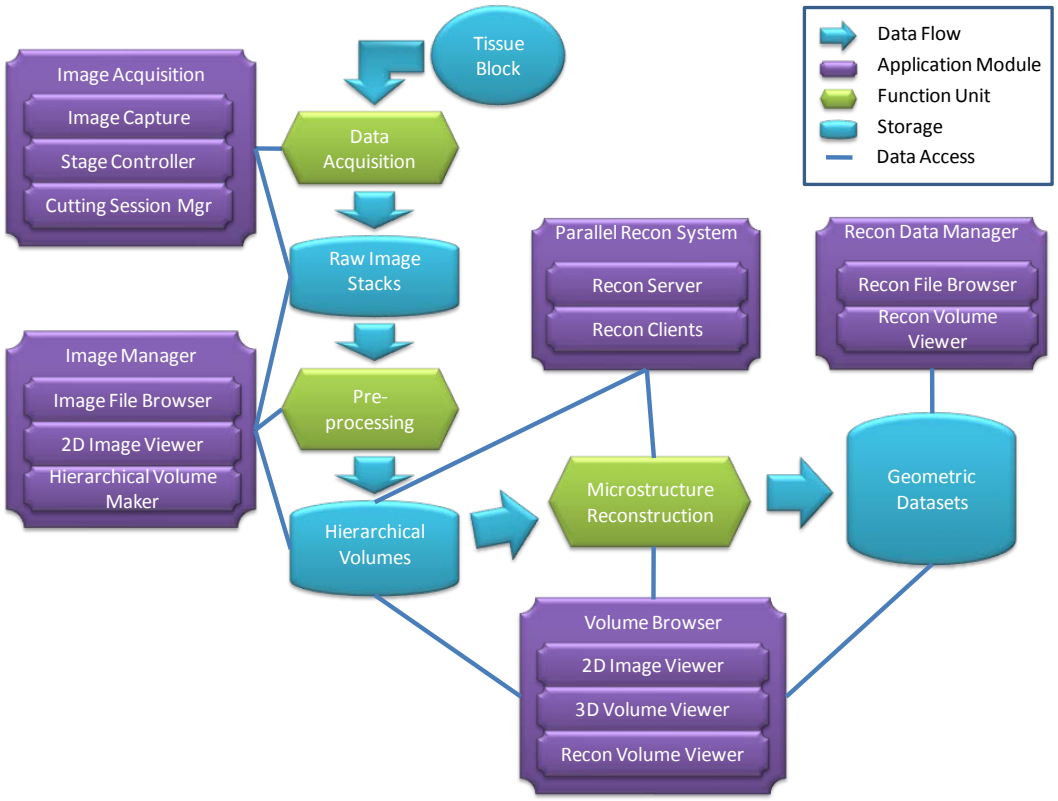


Fig. 63. KESM Software Framework.

Also, all the coordinate values for the lateral sectioning process should be restored even though all the systems would have shut down previously. The KSM manages the status of each cutting session.

When the image acquisition task is finished, all the raw image stacks should be trimmed in order to remove extra region where there is no tissue structure. The KESM Image Manager (KIM) browses raw images and manages the data pre-processing. To build the data for the KESM Hierarchical Data Clustering (KHDC) module, the volumetric data should be divided into small blocks and also multi-resolution data should be generated. These multi-resolution and sub-divided data sets will be inputs to the KESM Parallel Reconstruction System (KPRS).

The KPRS is a system that extract morphological information from the divided

blocks in parallel. When the whole mouse brain is sectioned and imaged, the reconstruction task is almost infeasible not only because of the immense size of the data but also because of the tracing time. KPRS uses a star topology network to configure the reconstruction process and provides an efficient method to extract the geometric structures from a large scale volumetric data set. A communication protocol was also desribed for this parallel processing. When the size of the data becomes even bigger, then multi-server configuration can also be considered.

B. Software

I have provided an integrated tool for KESM data sets. The image acquisition part is dedicated to the KESM design, but the rest of the system such as KIM, KHDC, KPRC, K2IV, and K3VR can be used as a general image management tool for other types of large scale volume data.

An image stack can be acquired by the KESM Image Acquisition System (KIAS). As I described in earlier sections, the amount of the KESM data sets are extremely large in most cases, So a typical desktop computer cannot handle the whole data set at once. In order to use the data sets for interactive data visualization, the raw data sets should be pre-processed. A hybrid representation, KHDC, consisting of octree data representation and MIP map will be used.

1. Data Processing Pipeline

Let us review the data processing pipeline before discussing the software frameworks. First, a tissue block is mounted on top of the stage of the KESM. KIAS generates several raw image stacks from the tissue block. Images in these raw image stacks have extra areas to be removed and intensity noises to be recovered. KIM converts raw

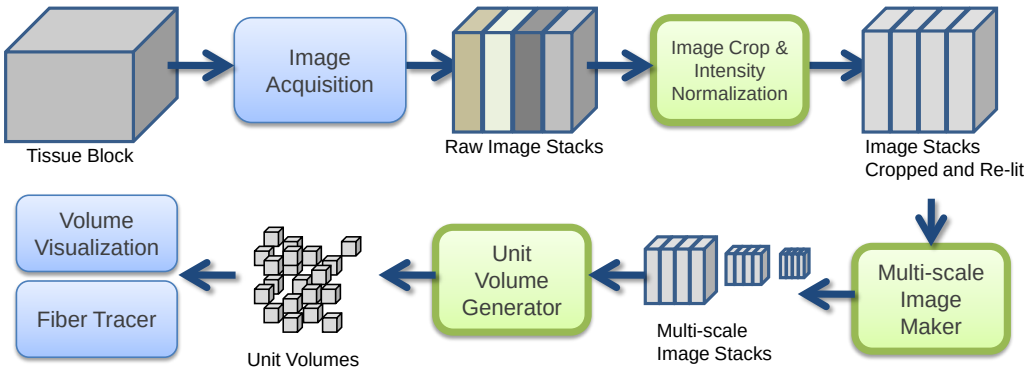


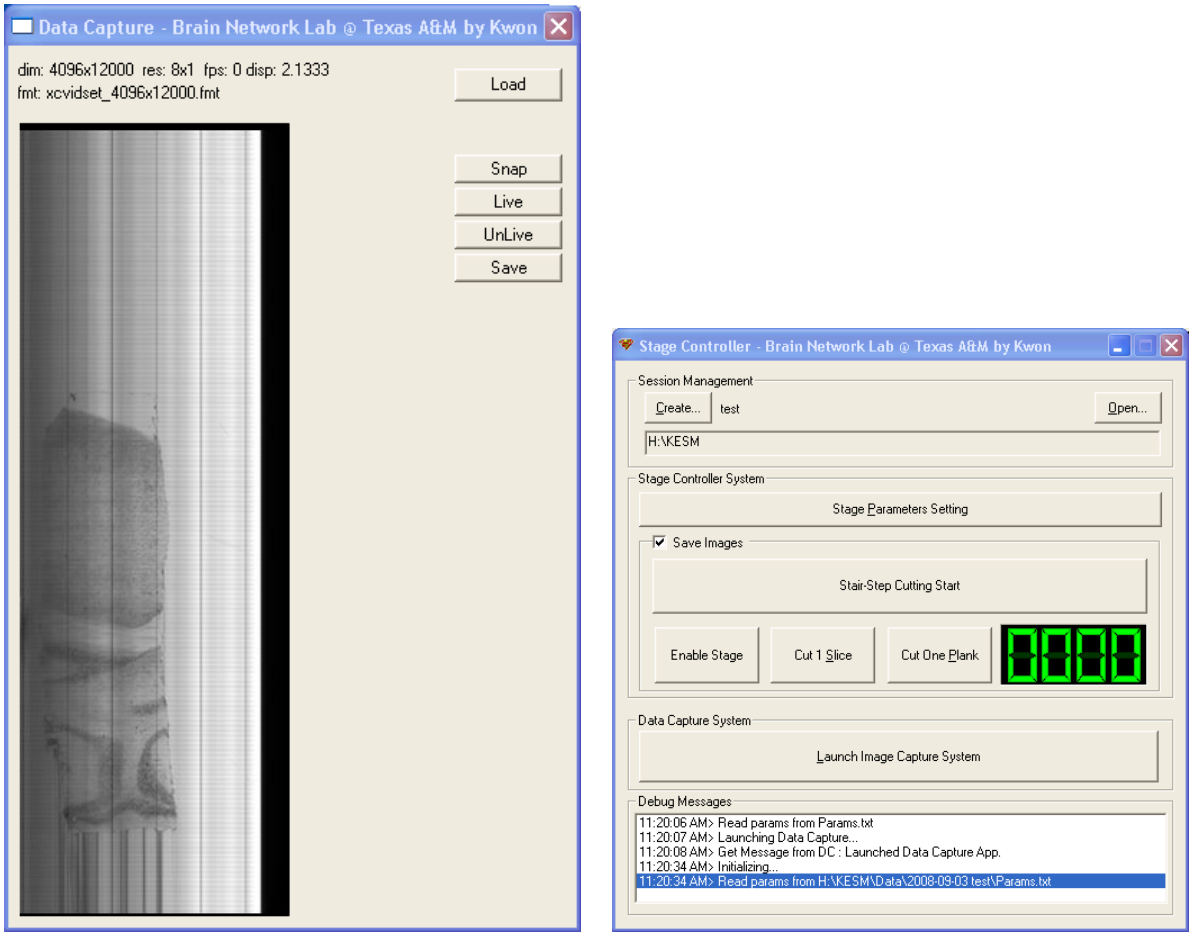
Fig. 64. Data Processing Pipeline. A tissue block turns into raw image stacks by the image acquisition system. The raw stacks result in several images stacks through automatic image crop and intensity normalization. Multi-scale Image Maker generates image stacks in different scales. Unit volumes can be accessed by the Unit Volume Generator.

images stacks to clean-cut image stacks which is ready to be reconstructed into three-dimensional volumetric data sets. These processed image stacks go into Multi-scale Image Maker which produces multiple image stacks in different resolutions. K3VR or Fibrous Data Tracer uses the multi-scale image stacks through the Unit Volume Genereator (Fig. 64).

2. Image Capturer and Stage Controller

Fig. 65 shows a screenshot of the KESM Image Capture System. KESM Stage Controller (KSC) communicates with KICS to synchronize two tasks: stage movement and corresponding image capture.

The operation parameters for image acquisition can be set by the interface shown in Fig. 66 (b) and the task of the stair-step operation can be monitored and managed by the dialog box (see Fig. 66 (c)).



(a) (b)

Fig. 65. KESM Image Acquisition System (KIAS). (a) KESM Image Capture System (KICS) (b) KESM Stage Controller (KSC).

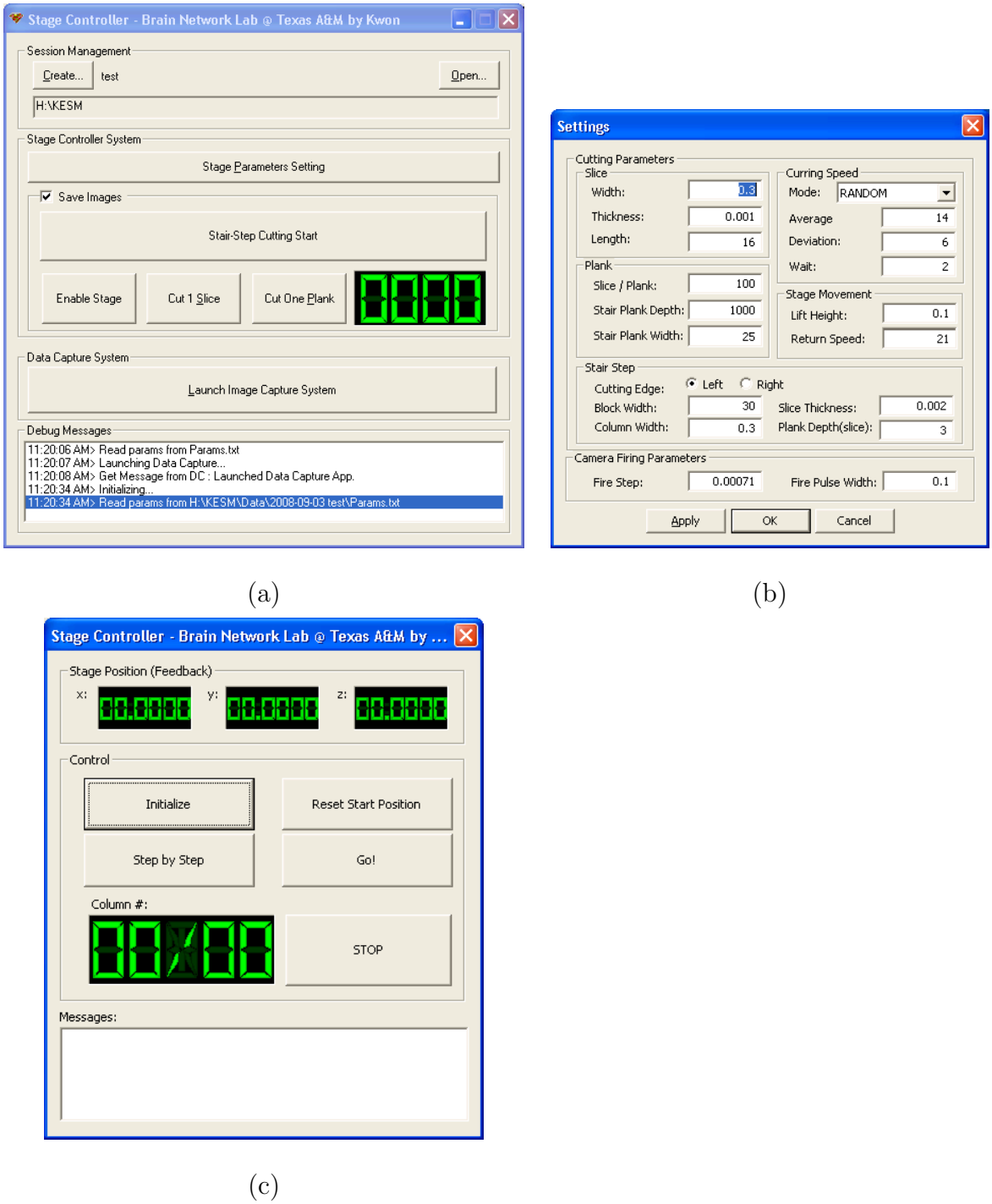


Fig. 66. KESM Stage Controller (KSC). (a) The main window of KSC (b) The dialog box for setting operation parameter (c) Stair-step controller.

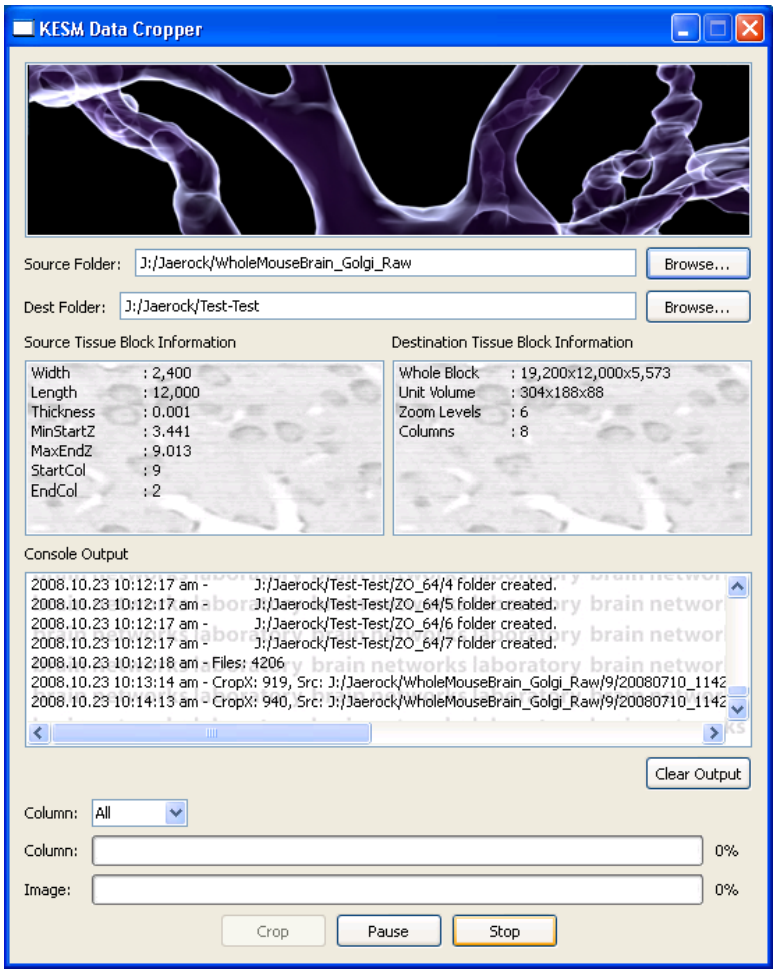


Fig. 67. Image Cropper. This software includes intensity normalization module.

3. Image Cropper and Intensity Normalizer

The Intensity Normalizer is embedded inside the Image Cropper (Fig. 67). The Image Cropper can also be considered as a tool for removing misaligned image chunk issue discussed in Chapter IV. I automated the whole process by implementing this crop software.

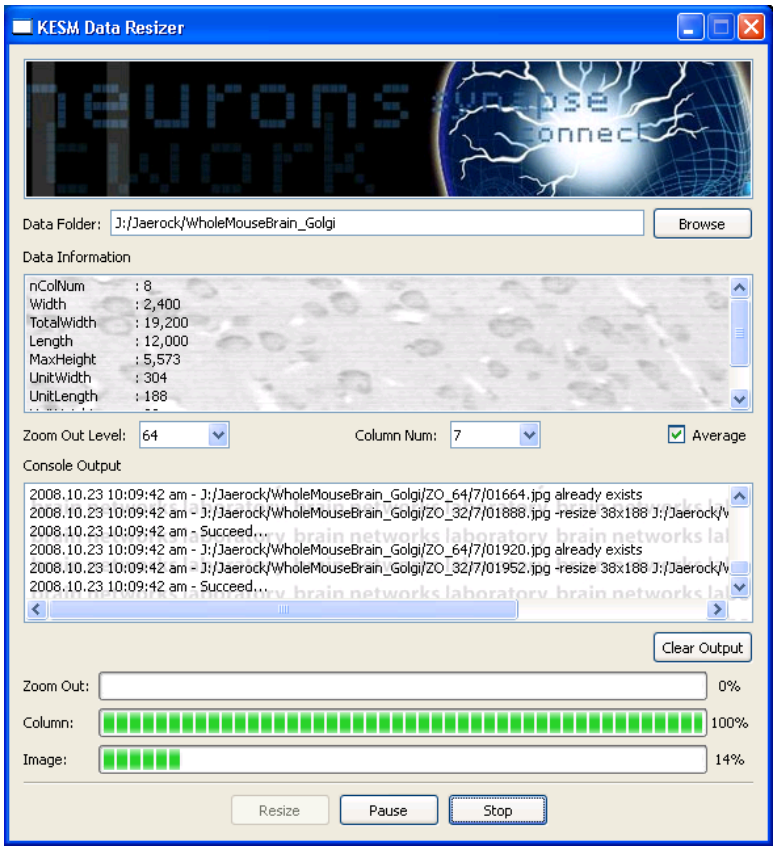


Fig. 68. Multi-scale Image Volume Maker. Users can choose either a specific column number or all columns.

4. Hierarchical Data Generator

In order to generate hierarchical data, I implemented a Multi-scale Image Volume Generator (Fig. 68). The software framework resizes all image stacks throughout all columns. This resize task is continued until the volume size resized reaches that of the unit volume.

Users can choose either a specific column to resize or all columns. After clicking the start button, all processes are done automatically.

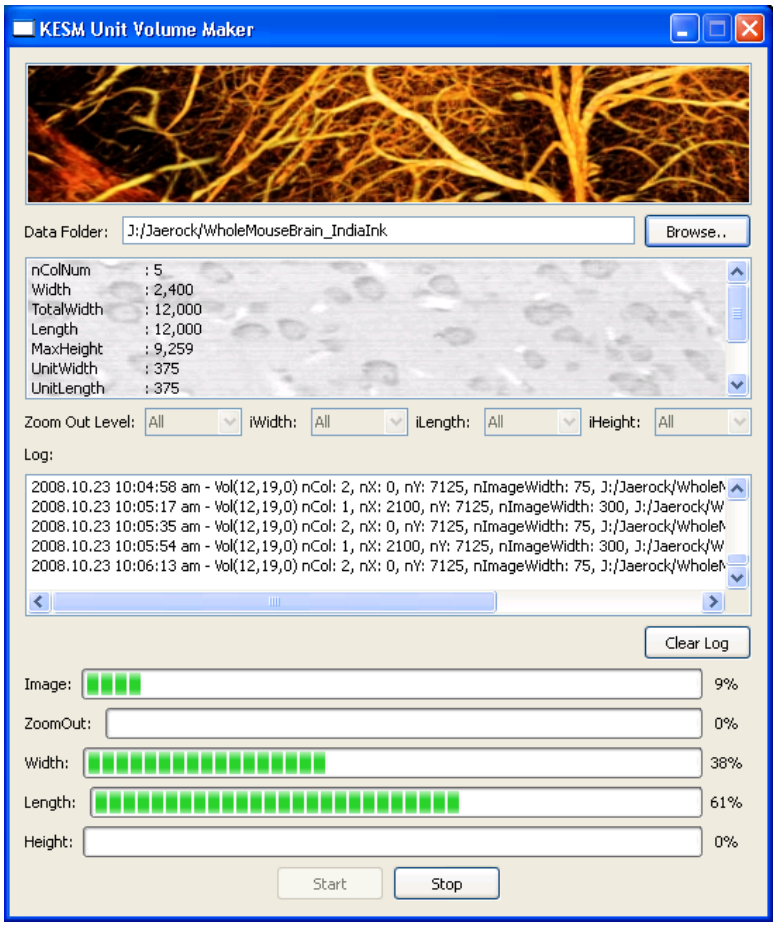


Fig. 69. Unit Volume Maker. Unit volumes are extracted by this software.

5. Unit Volume Maker

Through the Hierarchical Data Generator, now we have multi-scale image volumes in the form of column-wise images stacks. In order to visualize it or trace structures, unit volumes need to be extracted (see Fig. 69).

6. Fibrous Structure Tracer in 3D Volume and Structure Analyzer

Fig. 70 shows a screenshot of the Fibrous Structure Tracer which is an implementation of a model-based multi-scale tracing method [73]. The output files of the tracing are saved both in a VTK-compatible data file format and a data file represented by data

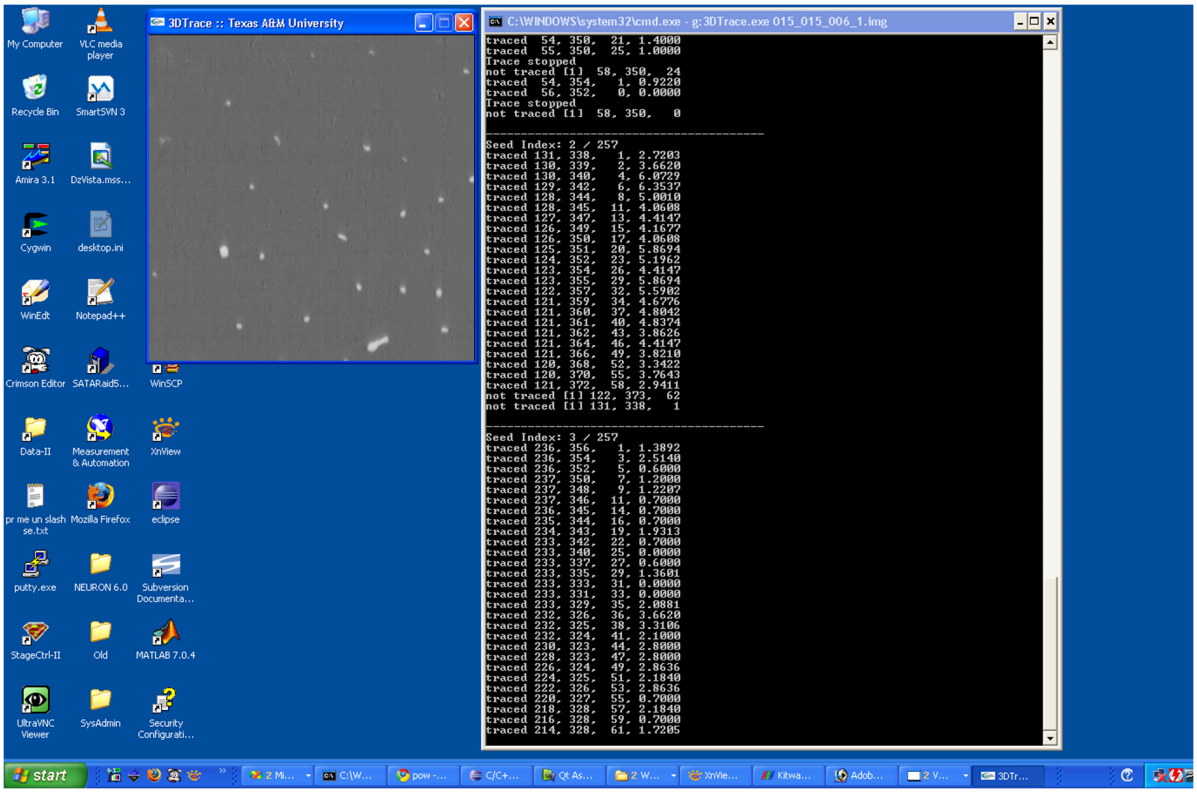


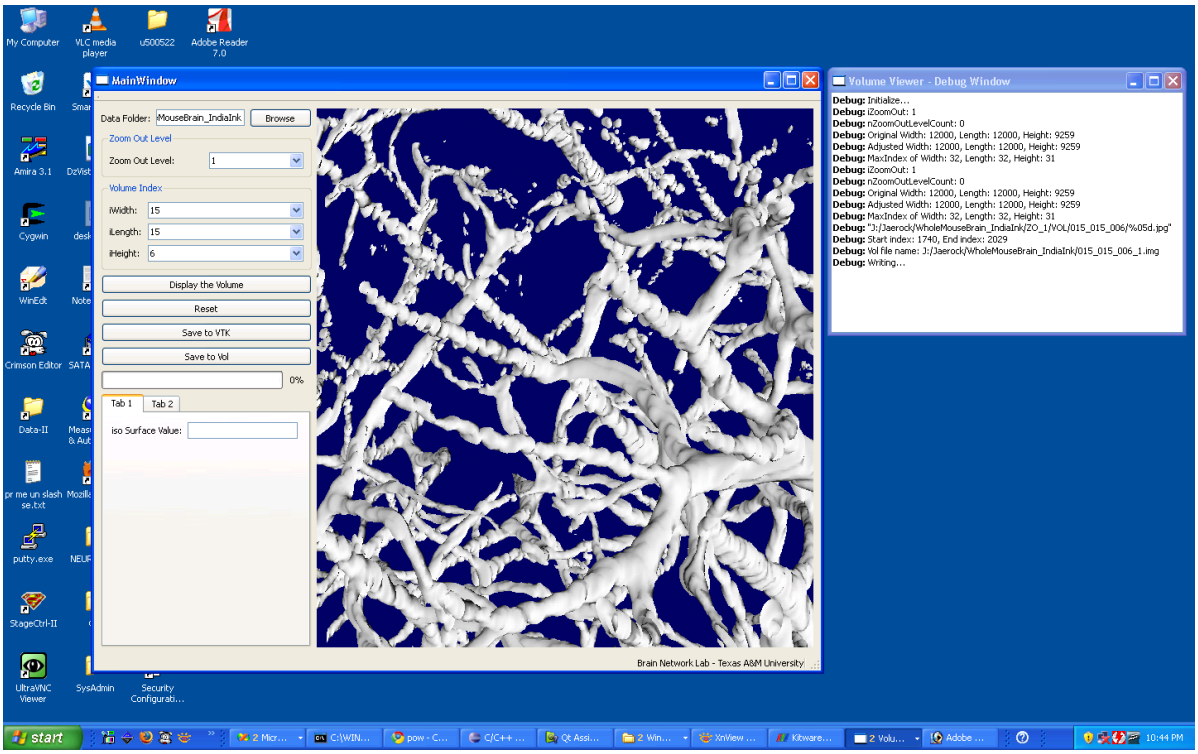
Fig. 70. Fibrous Structure Tracer.

structure and node segments represented in Chapter VI.

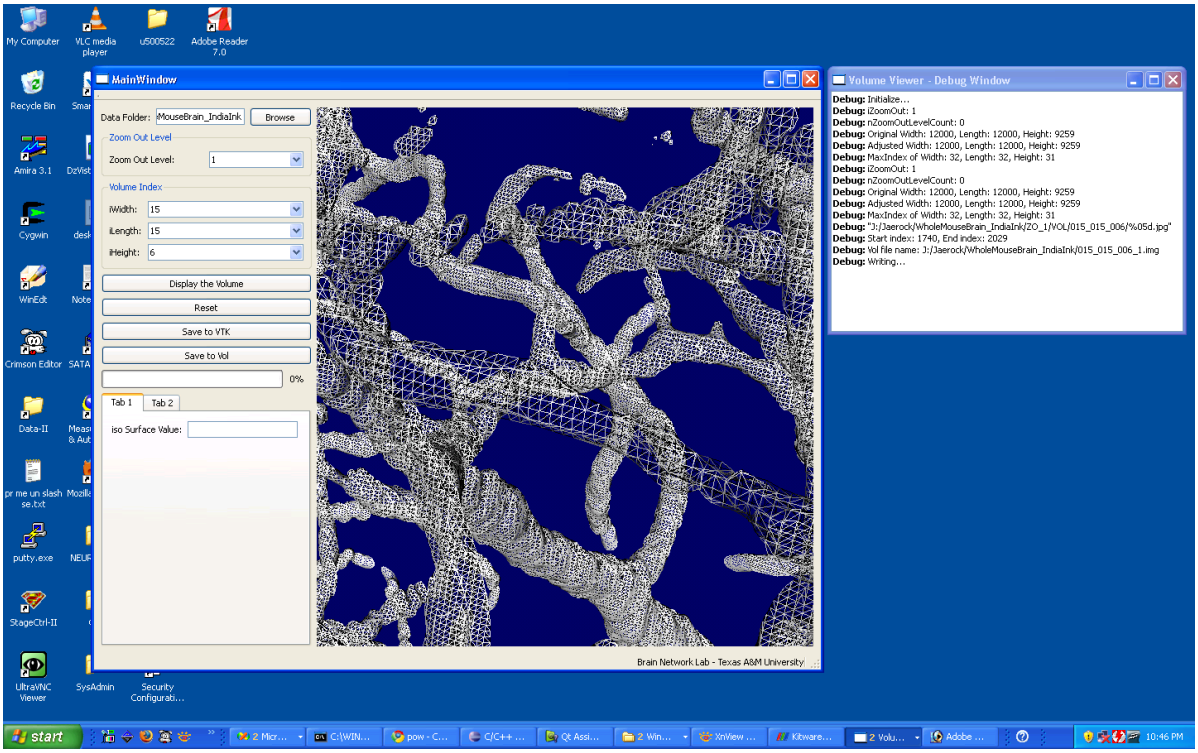
7. Examples of Visualization Software

KESM 3D Volume Renderer (K3VR) is shown in Fig. 71. The rendering software can display not just iso-surfaces but also wired frames.

In conclusion, the software framework is an integrated tool for acquiring, handling, and sharing the data sets. The morphological properties of the data sets can be traced and their structural information can be described by the fibrous data format. Also the data can be stored in standard data formats such as the VTK file format. Fig. 72 shows the extensibility of the KESM data sets. VTK's standard viewer, ParaView [74] can display the KESM data volume stored in the VTK file format (see



(a)



(b)

Fig. 71. Examples Of Visualization Software. (a) Iso-surface visualization (b) Wired-frame visualization

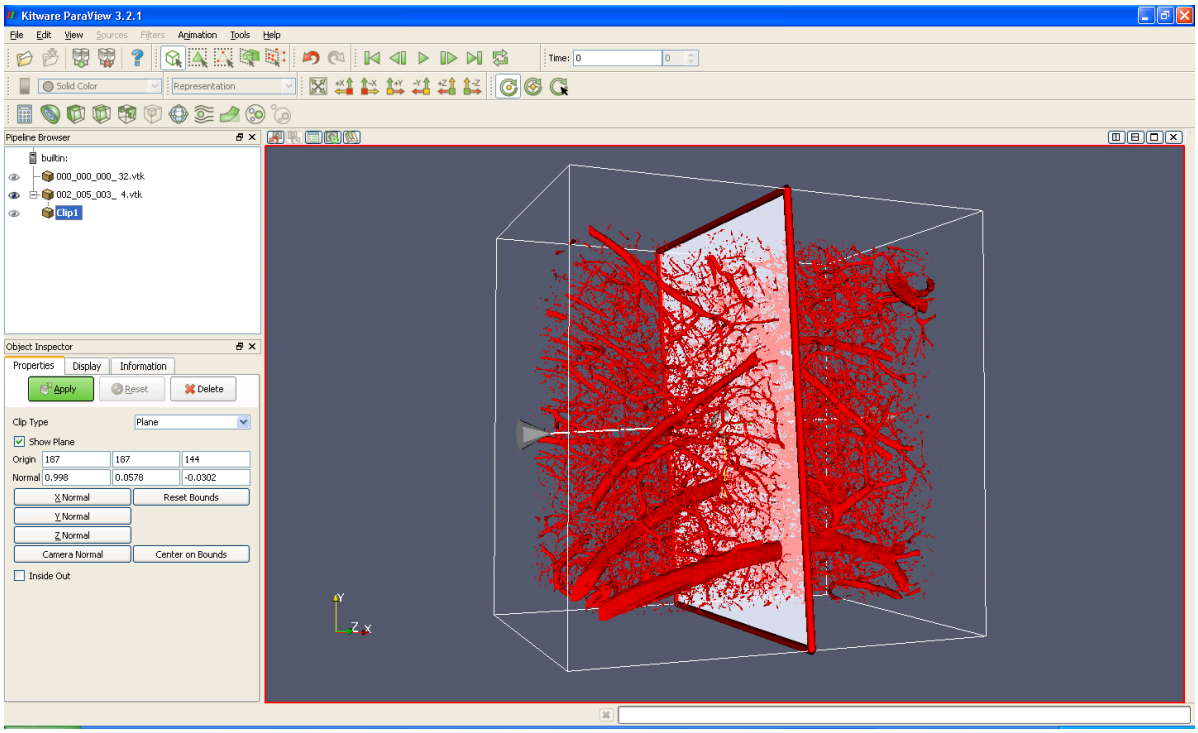


Fig. 72. ParaView [74]. This shows that the KESM data volume can be loaded by other visualization tools supporting VTK [51] data file format.

Fig. 72).

CHAPTER IX

CASE STUDIES

It is important for neuroscientist to have efficient tools to explore the microstructure data sets of the brain, because the morphological information in the brain is closely related to its function. It is well known that the function of the brain depends on the connectivity and the neuronal morphology. Therefore, understanding the relationship between structure and function is a crucial step to understand brain function. Throughout the previous sections, I reviewed a framework that enables the construction of a detailed connectivity and neuronal morphology. All the statistical and morphological information can allow us to explore theoretical issues such as modeling neural circuits, and eventually intelligence.

In relation to this dissertation, I have studied two theoretical issues: (1) A model of compensation for neuronal delay in neural networks [75], and (2) internal state predictability inside a neural system as an evolutionary precursor of self-awareness [76]. Anatomical predictions of these models can be tested with the KESM data.

The case studies here have been conducted for investigating delay compensation mechanism in the neural system and the role of internal state in terms of self-awareness. The delay compensation mechanism in neural systems may be scrutinized if we have exact morphological structure in cellular level and its physiological properties. The second case study is about the predictability of internal state in neural systems. It is interesting to note that anatomically determinable properties such as delay distributions can be related to dynamic properties of neuronal network activation. For example, Thiel et al. have shown that random networks with broadly distributed delay show simple dynamics while those with narrowly distributed delay exhibit complex dynamics [77]. The data from KESM can help assess dynamical

properties of networks or subnetworks in the mouse brain by measuring the delay distribution, estimated through axonal length and thickness. The prediction of my theoretical work indicates that self-awareness may be related to simpler, more predictable dynamics. So, identifying regions in the mouse brain that show more broadly distributed delay could be related to self-awareness.

A. Facilitatory Neuronal Dynamics

1. Introduction

Goal-directed behavior is a hallmark of intelligent cognitive systems. Therefore, understanding such behavior is not only important but also essential to scrutinize intelligence. However, it is not easy to directly investigate goal-directed behavior since there is an implied *agent* behind such behavior, and there is yet not a consensus on what constitutes an agent.

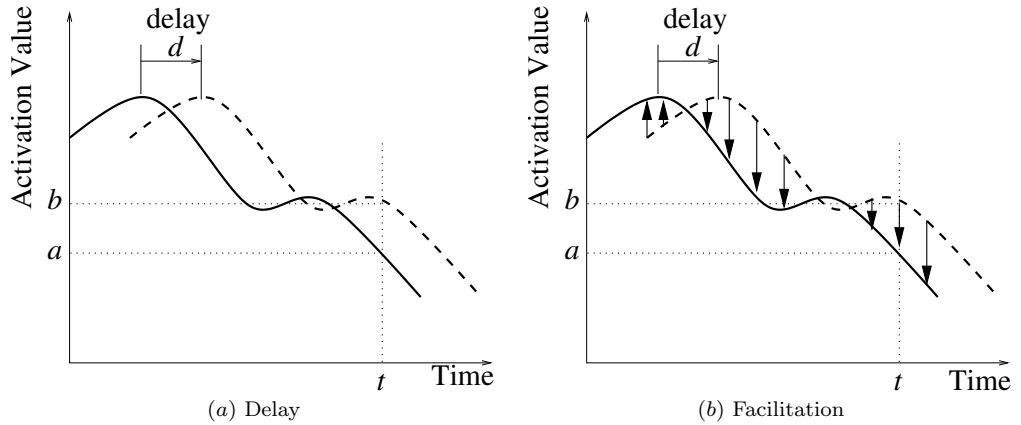
Thus, here we take a different approach to initiate a first step toward understanding goal-directed behavior. Our strategy is to focus on a precondition, or a necessary condition for goal-directed behavior, rather than trying to address the problem head-on.

The main question we will address here is how the precondition could have evolved. Once the prerequisite has evolved, it could have laid a critical stepping stone toward goal-directed behavior. We theorize that one important necessary condition of goal-directed behavior is prediction. Note that a goal is always defined as a future event. Thus, without the ability to anticipate future events, one may not be able to establish a goal. In order to anticipate, one needs to be able to predict. Consequently, by analyzing how prediction has evolved, we could shed light on a potential evolutionary pathway toward goal-directed behavior. Also, we must note that

prediction is increasingly being recognized as one of the core functions of the brain [78][79] (see also [80][81] on prediction in dynamic neural network architectures).

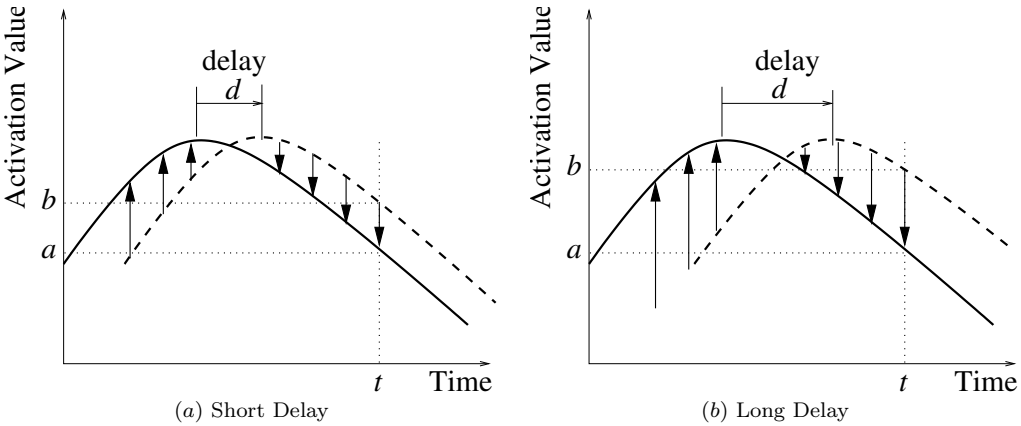
In my previous work with my colleagues [82][83][84][85], we hypothesized that delay in the nervous system could have led to a delay compensation mechanism, which in turn could have further developed into a predictive function. First, let us take a look at neuronal delay in detail before investigating the predictive property of the delay compensation mechanism. Strictly speaking, representations of the present in the brain may not even be precisely aligned with the present in the environment. Our sensory information would reflect the past if the higher perceptual areas in the brain register the signal at the moment the signal is received. Consider visual processing. A series of steps is required for visual stimulus information to reach higher visual processing areas: photoreceptors, bipolar cells, ganglion cells, the lateral geniculate nucleus, the primary visual cortex, and beyond [86]. It could take in the range of 100 to 130 ms for the visual signal to arrive in the prefrontal cortex (in monkeys) [87]. In order to make up for the neuronal transmission delay, the brain should utilize information from the past and *predict* the current state.

Some researchers probed this topic in terms of delay compensation [88][84][83] or prediction [89][90]. Lim and Choe suggested a neural dynamic model for delay compensation using Facilitating Activity Network (FAN) based on short term plasticity in the neuron known as *facilitating synapses* [88][83]. Facilitating synapses have been found at a single neuron level in which the membrane potential shows a dynamic sensitivity to the changing rate of the input [88][91]. As illustrated in Fig. 73, the original signal can be recovered from the delayed signal by using facilitating dynamics. According to the facilitation model, as Fig. 74 illustrates, higher facilitation rates are needed to effectively deal with longer delay. However, the FAN model turns out to have limitations, i.e., oscillation under high facilitation rate (see Sec. 3 for details).



(a) Delay (b) Facilitation

Fig. 73. Delay And Delay Compensation Through Facilitating Neural Activity. (*a*) The solid curve represents the original signal, and the dotted curve corresponds to the delayed signal (delayed by d). (*b*) The original signal can be extrapolated by facilitating the neural activity (further increasing when the signal is increasing, and further decreasing when the signal is decreasing). For example, an activation value b at time t (original signal from $t - d$, delayed by d) can be modulated down to a through facilitating dynamics, where the modulated value a is an approximation of the original signal at time t.



(a) Short Delay (b) Long Delay

Fig. 74. Length Of Delay And Required Degree Of Facilitation For Delay Compensation. The solid curve represents the original signal, and the dotted curve the delayed signal (delayed by d). (*a*) Short delay requires only a moderate amount of facilitation to compensate for the delay. (*b*) More facilitation is needed as the length of delay between the original and the delayed signal becomes greater (the vertical arrows are longer in (*b*) than in (*a*)).

Furthermore, the analysis in [88][84] did not consider differential utilization of facilitation among different neuron types within the context of the entire network (e.g., sensory neurons vs. motor neurons).

Here, we propose an improved dynamic model, *Neuronal Dynamics using Previous Immediate Activation value* (NDPIA) that solves the oscillation problem in FAN. In addition, we conducted experiments in less restricted conditions than in [88, 84]: (1) input delay was applied to the system for the entire duration of each experiment, and (2) we extended the delay to twice the usual value compared to the earlier experiments with FAN, and analyzed the results from the increased delay.

To test NDPIA and to investigate the properties of the neuronal networks with the suggested neuronal dynamics, we employed a 2 degree-of-freedom (2D) pole-balancing [92] agents with evolved recurrent neural networks as their controllers (cf. [93][94]). We used conventional neuroevolution to train the networks (see Sec. b for detailed justification, and Ward and Ward for successful use of such strategy in a different task domain). [95]

Our main findings are as follows: (1) NDPIA can solve the oscillation problem in FAN during heightened facilitation. (2) Motor neurons in a NDPIA network tend to evolve high facilitation rates, confirming similar previous results with FAN. (3) Longer delay leads to higher facilitation rates. (4) Neural network controllers using NDPIA dynamics result in better performance in pole balancing tasks than those based on FAN. (5) NDPIA networks show robust performance under extremely high facilitation rates, especially when only the motor neurons are facilitated. These results suggest that delay and facilitation rate must be positively correlated for effective compensation of delay, and the best part in the system to introduce such dynamics is the motor system.

Below, we first look into related research, then we analyze the limitations in the

FAN dynamics. Then we will propose a new facilitating dynamics (NDPIA). Next, the 2D pole-balancing problem and evolutionary neural networks will be introduced. Finally we will present and analyze the results, followed by discussion and conclusion.

2. Background

The activation level or the membrane potential of the postsynaptic neuron is modulated by the change in the rate of past activation. These dynamic synapses generate short-term plasticity, which shows activity-dependent decrease (depression) or increase (facilitation) in synaptic transmission [91][96]. These activities occur within several hundred milliseconds from the onset of the stimulus [91][97]. [88][83][84] investigated the relationship between these neuronal dynamics and delay compensation, and suggested that facilitating dynamics at a single neuron level may play an important role in the compensation of neuronal transmission delay.

How can such dynamics be realized in a neural network? We can begin with conventional artificial neural networks (ANNs), but ANNs lack such single neuron level dynamics (note the adding recurrent connections can introduce a network-level dynamics). As we can see in Eq. (9.1), the activation values in conventional ANNs are determined by the instantaneous input value and the connection weights.

$$X(t) = g\left(\sum_{j=1}^{m} w_j X_j(t)\right) \tag{9.1}$$

where $g(\cdot)$ is a nonlinear activation function such as the sigmoid function, m is the number of neurons of the preceding layer, w_j is the connection weight, and X_j is an activation value from a neuron of the preceding layer [88][83][84]. Eq. (9.1) shows that there is no room to consider the past values of X_j. Recurrent ANNs could be one simple solution for this, but the dynamics may not be fast enough to cope with

input delays. [98] proposed a neural network based Smith predictor to compensate for large time delay; [99] used the Kalman filter in the internal forward model to predict the next state; and [84] showed that facilitating neuronal dynamics at a single neuron level can play an important role in compensating for input delays.

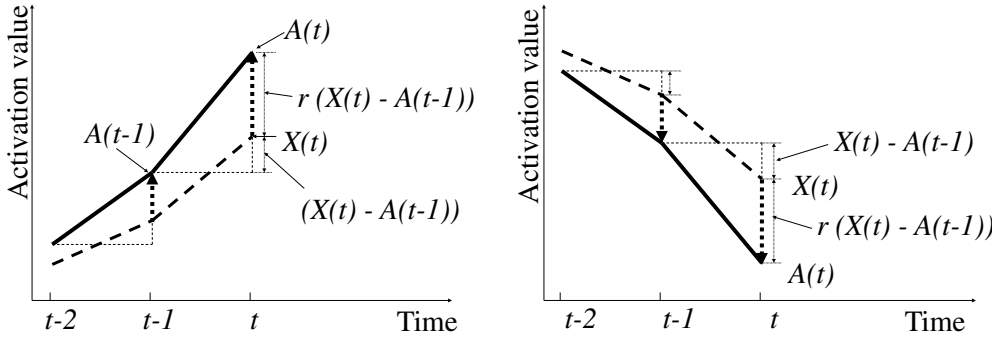
In order to overcome the issues above, the activation value needs to be directly modulated as in the Facilitating Activity Network (FAN) model [88][83][84]:

$$A(t) = X(t) + r\Delta(t) \tag{9.2}$$

where $A(t)$ is the modulated (facilitated or depressed) activation value at time t, $X(t)$ is the immediate activation value, r is a dynamic rate ($-1 \leq r \leq 1$), and $\Delta(t)$ is $X(t) - A(t-1)$.

If $r \geq 0$, and if the signal increases for a while, the activation value is augmented by the difference $\Delta(t)$ of the immediate activation value $X(t)$ and the previous modulated activation $A(t-1)$ with the rate r (see Fig. 75(a)). If $r \geq 0$, but if the signal decreases, the activation value is diminished by $\Delta(t)$, because it becomes a negative value in this case as shown in Fig. 75(b). This results in facilitation.

Suppose $r \leq 0$, and that the signal increases for a while, then the activation value is diminished by the difference $\Delta(t)$ between the immediate activation value and the previous modulated activation with the rate r. If the signal decreases for a while under the same condition, the amount of decrease becomes smaller than the immediate value by $\Delta(t)$ with the rate r, because r is a negative value and $\Delta(t)$ is a negative value as well, so $r\Delta(t)$ becomes a positive value. This makes the signal greater than the immediate signal. Furthermore it means that the signal is decreased less than what it is supposed to be. In other words, the modulated activation values can be considered within the range of $(X(t) - \Delta(t)) \leq A(t) \leq (X(t) + \Delta(t))$ [88] which means that the present activation value could be diminished by $\Delta(t)$ (depressing dynamics) or



(a) Increasing activation (b) Decreasing activation

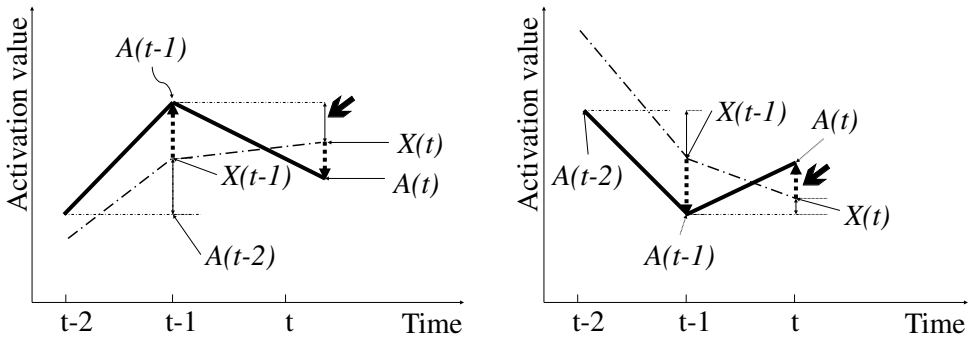
Fig. 75. Facilitating Neural Activity. (a) The immediate activation value $X(t)$ is modulated by the difference between $X(t)$ and the modulated activation value $A(t-1)$ in the previous time step, with facilitation rate r. (b) The same principle can be applied to the decreasing activation case.

augmented by $\Delta(t)$ (facilitating dynamics).

Neural networks using the FAN model showed not only better performance than conventional networks but they were also more robust in various delay conditions [88][82].

3. Methods: Enhanced Facilitating Activity Model

Even though [88][83][84] paid attention to short-term synaptic plasticity, especially facilitating synapses, and suggested a compensation mechanism for neuronal transmission delay, several further challenges remain. As Fig. 74 illustrates, we need to use a higher facilitation rate as the delay increases. However, Lim and Choe did not investigate the effect of higher facilitating rates. rates. When the FAN model is used with high facilitation rates, the modulated activation values become unstable/oscillatory. Furthermore, a systematic analysis of the neuronal dynamics in the network level is needed because Lim and Choe did not investigate differential utilization of facilitating dynamics dependent on neuron type.



(a) Increasing activation (b) Decreasing activation

Fig. 76. Problems In Facilitating Dynamics Of FAN. (a) When the activity is increasing, the immediate activation value $X(t)$ could be smaller than the modulated value $A(t-1)$ from the previous time step, so the modulated value at the present $A(t)$ becomes smaller than the immediate value $X(t)$. This property of the conventional FAN model makes the system output to become unstable. (b) Basically, the same analysis can be applied in the case of decreasing activity. When the activity is decreasing, $X(t)$ is larger than $A(t-1)$. Hence, $A(t)$ becomes even larger than $X(t)$.

Here, we propose an improved dynamics model to address these challenges. The previous FAN model turns out to have a limitation especially when longer delay is applied to the model. Below, we analyze the potential problems of the FAN model in detail and propose an enhanced model to deal with the problems. First, we expand Eq. (9.2) into:

$$\begin{aligned} A(t) &= \left(\sum_{n=0}^{k-1}(-1)^n r^n (1+r) X(t-n)\right) \\ &+ (-1)^k r^k A(t-k) \end{aligned} \tag{9.3}$$

Now we can more clearly see that the current modulated activation value $A(t)$ is a function of $X(t-1), X(t-2), X(t-3)$ and so on. The problem is that, given a positive dynamic rate r, $X(t-1)$, $X(t-3)$, etc. contribute negatively while $X(t-2)$,

$X(t-4)$, etc. positively. These positive and negative components can give rise to abrupt oscillations in $A(t)$ that originally do not exist in the input signal.

To better illustrate the problem, let us take an example in the case of facilitating dynamics. As we can see in Fig. 76(a), even when $X(t)$ keeps increasing from $X(t-1)$, the immediate activation value $X(t)$ could be smaller than the previous modulated value $A(t-1)$. This is not desirable since $A(\cdot)$ will oscillate unlike $X(\cdot)$. The same phenomenon happens when the activity is decreasing as in Fig. 76(b).

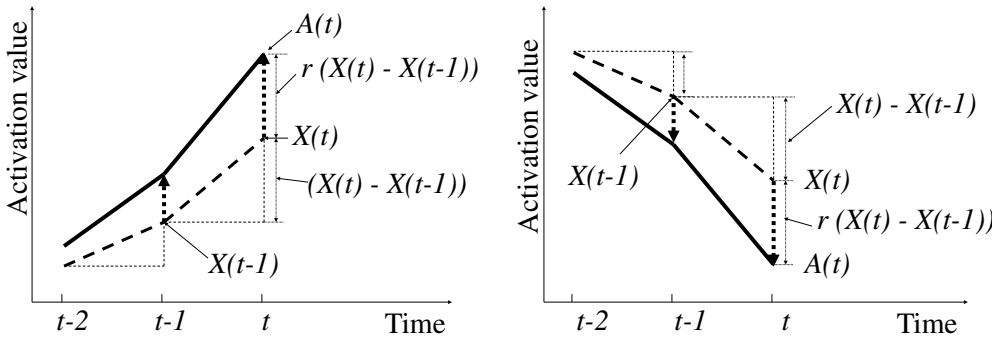
a. Enhanced Facilitating Activity Model

In order to address the above issue, we propose an improved neuronal dynamics model (NDPIA) which considers only the previous immediate activation value (Fig. 77).

$$A(t) = X(t) + r(X(t) - X(t-1)) \tag{9.4}$$

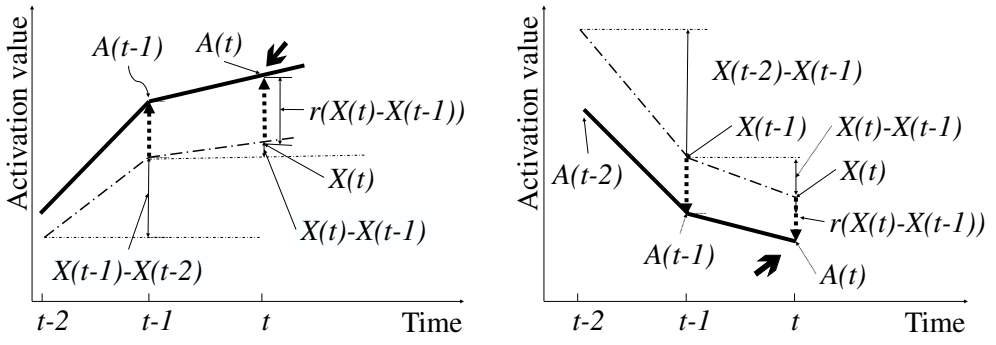
where $A(t)$ is the modulated (facilitated or depressed) activation value at time t, $X(t)$ is the immediate activation value, and r is the dynamic rate. The dynamic rate r can either facilitate or depress the activity, and it is not limited to $-1 \leq r \leq 1$, so that we can either facilitate or depress the immediate activation values as highly as we want. But practically, this value should not be too high.

As we have shown in Eg. (9.3), the effect of $X(t-(n+1))$ disappears very quickly as n increases and r is less than 1. NDPIA accounts for the current and the previous immediate activation values. So in order to consider the previous activation values such as $A(t-1)$ and $A(t-2)$ prior to the immediate one, we used recurrent neural networks in the present paper, and the context inputs that are simply feedback from the hidden layer could make up for the effect of older past activation values. Fig. 78 shows that the proposed facilitation model can solve the problem in FAN.



(*a*) Increasing activation (*b*) Decreasing activation

Fig. 77. Proposed Facilitation Neural Activity. (*a*) By using the previous immediate value $X(t-1)$ instead of the modulated value $A(t-1)$, the present modulated value is stabilized. (*b*) The same change is applied to the decreasing activation case.



(*a*) (*b*)

Fig. 78. The Proposed Facilitation Model Solving The Problem In FAN. (*a*) The modulated activation value $A(t)$ is guaranteed to be larger than the immediate activation value $X(t)$ as long as activation increases. (*b*) $A(t)$ is guaranteed to be smaller than $X(t)$ as long as activation decreases. Compare to Fig. 76.

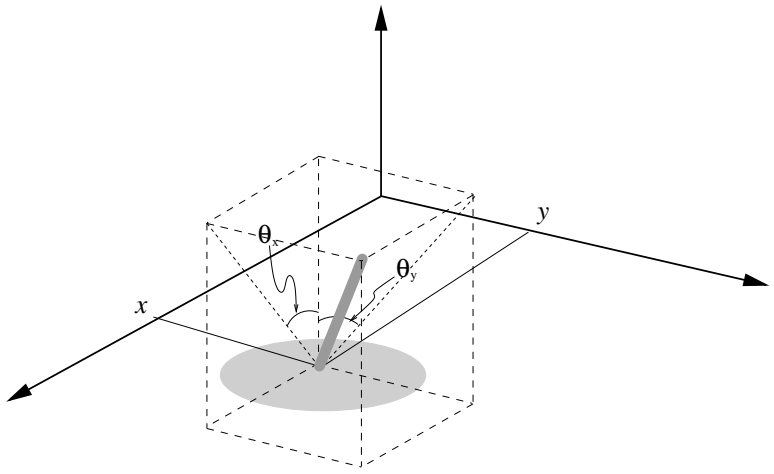


Fig. 79. 2 Degree-Of-Freedom Pole-Balancing Task. The cart (gray disc) with an upright pole attached to it must move around on a 2D plane while keeping the pole balanced upright. The cart controller receives the location (x, y), the pole angle (θ_x, θ_y), and their respective velocities as the input, and generates the force in the x and the y direction. Note that θ_x and θ_y are projected angles to the x-z and the y-z plain respectively.

b. 2D Pole-balancing Problem with Delayed Inputs

We tested our new facilitating dynamics in recurrent neural network controller for a 2D pole-balancing task (Fig. 79). The state of the cart (the gray disc on the bottom Fig. 79) with a pole on top is characterized by the following physical parameters: The cart position in the plane (x, y), the velocity of the cart $(\dot{x}, \dot{y})$, the angle of the pole from the vertical in the x and the y directions (θ_x, θ_y), and their angular velocities $(\dot{\theta_x}, \dot{\theta_y})$ [92].

To test our facilitation dynamics, we employed the 2D pole-balancing problem, following [84]. The differences from [84] are as follows. First, we tested with delays in all inputs, and the delay was applied during the entire test period in all experiments. In [84], delay was applied either to a subset of the input for the entire duration, or to all the inputs only for a limited time period during each trial. Second, we evolved

the controllers under no delay condition and tested them with up to two-step delay with increased facilitation rate. Longer delay may not be acceptable because of the high possibility of phase difference (see Sec. 5 for a discussion). In [84] only one step delay was investigated for measuring the performance of controller networks. Third, we used conventional GA to evolve the controllers instead of Enforced SubPopulation algorithm (ESP) [92][94]. The main reason for using conventional GA was to have a clearly separated role for the sensory and the motor neurons, to investigate the differential utilization of facilitating dynamics in these neuron types.

The cart controller applies force to the cart on a flat surface to balance the pole (the pole must remain within $\pm 15^\circ$ of the vertical). The force was applied in both the x and the y directions at a 0.1 second interval. If the controller balances the pole more than 5,000 steps (1 step = 100 milliseconds), we consider it as a success. The fitness function returned the number of steps the agent balanced the pole within $\pm 15^\circ$ from the vertical and stayed inside a 3m $\times$ 3m area (each axis ranging from -1.5m to 1.5m). We used recurrent neural networks to control the cart (Fig. 80). See Sec. c for details on the neural network controller. Fifty recurrent networks were evaluated in each generation, and to avoid situations where some neurons evolve to have accidentally good fitness values, we used the roulette wheel sampling method [100, 101, 102]. We used a pole length of 0.5m tilted 1 degree from the vertical towards the $+y$ direction with $(\dot{\theta_x}, \dot{\theta_y}) = (0, 0)$ in the initial state. Force within the range of -10N to 10N was applied to the cart at a time step of 0.1 second, based on the output (F_x, F_y).

With this setup, we (1) investigated the effect of dynamic rates in a single neuron level by evolving the rates from depressing to facilitating property, (2) compared the performance between FAN and NDPIA, and (3) showed that facilitating motor neurons are better at coping with longer delays than facilitating sensory inputs or

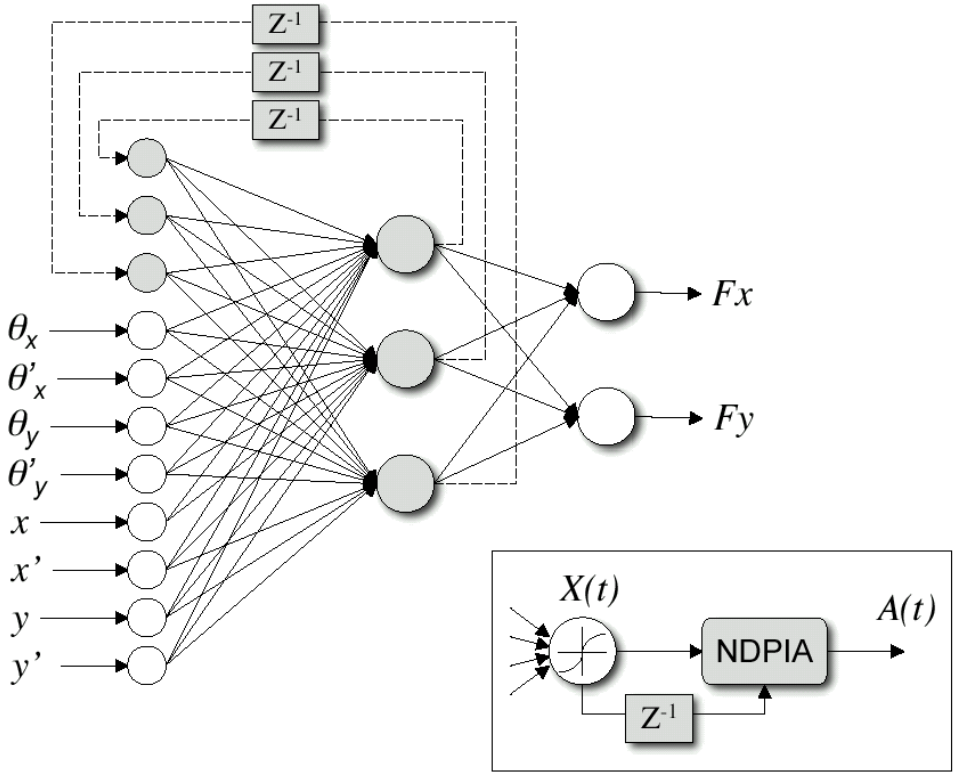


Fig. 80. Recurrent Neural Network For 2D Pole-Balancing. The signal flow inside each neuron is shown in the box. Z^{-1} means unit delay.

both sensory and motor neurons.

c. Neuroevolution

[88][84] investigated dynamic activation rates in a single neuron level, however, they have not tested the effect of facilitation in different parts of the network (e.g., by neuron type).

We evolved controllers having recurrent neural networks with dynamics neuronal activities. These activity rates were evolved to range across $-1 \leq r \leq 1$ which means it could be facilitating or depressing.

We used a recurrent neural network with eight input nodes, three context input nodes, three hidden neurons, and two output neurons in order to control the cart

in the plane (3m $\times$ 3m). Fig. 80 shows the recurrent network that we used in the experiments. Input nodes correspond to the cart position (x, y), the velocity of the cart $(\dot{x}, \dot{y})$, the angle of the pole from the vertical in the x and the y directions (θ_x, θ_y), and their angular velocities $(\dot{\theta_x}, \dot{\theta_y})$. The hidden layer activations are fed back as contextual input, with a unit delay. Output neurons F_x and F_y represent the force in the x and the y direction, respectively. Each neuron's immediate activity is calculated by Eq. (1), and subsequently facilitated using the dynamics in Eg. (4).

In training these non-linear controllers, neuroevolution methods proved efficient [93][94]. Unlike [93][94], we used a conventional neuroevolution method instead of ESP. The chromosome encoded the connection weights between input nodes and hidden layer neurons, and between hidden layer neurons and output neurons. In the experiment of the evolution of dynamic activation rates, we additionally included a dynamic rate parameter in the chromosome. Crossover occurred with probability 0.7 and the chromosome was mutated by ± 0.3 (perturbation rate) with probability 0.2. These parameters were determined empirically.

4. Experiments and Results

First, we tested whether NDPIA helps fix the unstableness/oscillation problem in FAN, using a fixed time series as input. Next, to test the rest, we evolved recurrent neural networks with FAN or NDPIA dynamics, where the connection weights and also the dynamic rates were allowed to evolve. The networks were trained in a 2D pole-balancing task, and then tested with added delay in the sensory signals (the input).

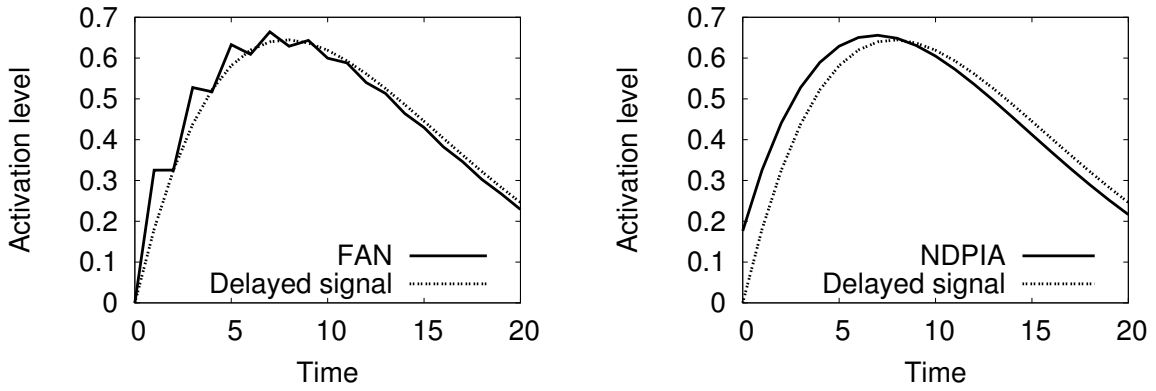
a. Enhanced Facilitating Activity Model

With NDPIA, we were able to correct the oscillation problem in FAN, discussed in Sec. 2. We compared the two models with a signal taking a simple functional form: $f(t) = 2 \times \exp(-t) \times \sin(t)$ (t is time). Figs. 81 and 82 show portions of the function where the change in the signal is either fast or slow. First we observed the part of the signal where the signal changes rapidly. Here we used a dynamic activation rate of 0.8 (facilitation). In Fig. 81(a), which shows FAN, the immediate activation value cannot keep up with the modulated one, thus oscillation occurs in the modulated activation values. In other words, facilitation did not occur properly in this case. Fig. 81(b) shows that using NDPIA these oscillations can be removed. Even when the signal changes slowly, if the facilitation rate is high enough (0.9 was used in this case), the oscillation would occur again (Fig. 82). Facilitation rate of 0.9 might seem too high, but as we examined in Sec. 1, high facilitation rates can be necessary. As before, FAN results in oscillation (Fig. 82(a)) while NDPIA results in no oscillation (Fig. 82(b)).

b. Evolved Dynamic Activation Rates

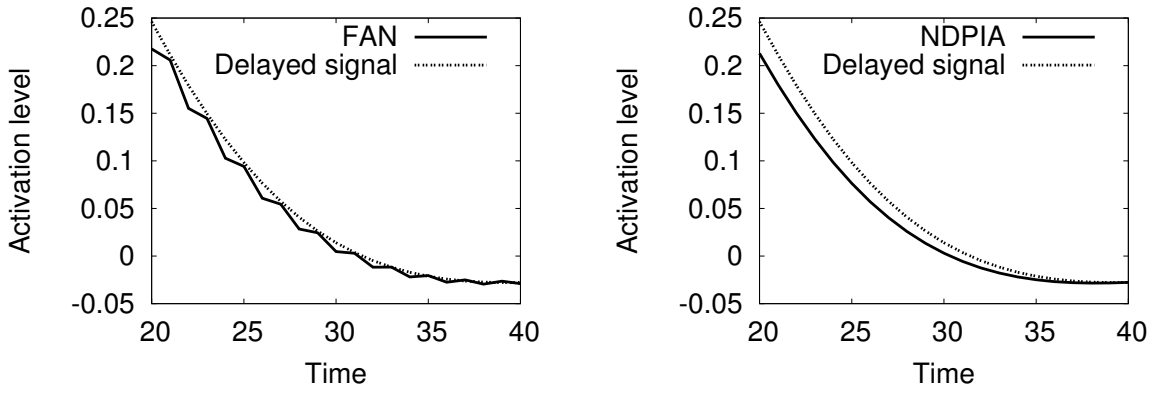
In the recurrent neural network controller, the hidden units receive direct sensory input, so we can say these are sensory neurons. In a similar manner we can consider the output neurons as motor neurons since they are directly coupled to the cart motion. Our main question here is if all types of neurons (sensory or motor) evolve to utilize facilitating dynamics under delayed input conditions. Furthermore, we question if increase in input delay leads to stronger facilitating dynamics.

In order to test these, we encoded into the chromosome the dynamic activation rates of sensory neurons (hidden) and motor neurons (output) as well as the synaptic



(*a*) FAN (*b*) NDPIA

Fig. 81. Facilitation Under Fast Signal Change Condition. This graph shows a small portion of the function $f(t) = 2 \times \exp(-t) \times \sin(t)$ in the interval [0..20] (dotted line, simulating a delayed signal). The x axis represents time t and the y axis the activation value $f(t)$. Facilitation rate of $r = 0.8$ was used in this example. (*a*) When the signal changes quickly (increasing leg, from time 0 to 7), the FAN results in jagged oscillation. Note that when the signal change is slow relative to the facilitation rate (decreasing leg, from time 7 to 20), the oscillation disappears. (*b*) The proposed method eliminates the oscillation problem in (*a*). Note that due to the facilitation, the resulting curve (solid line) appears shifted to the left (i.e., we can say that delay was compensated).

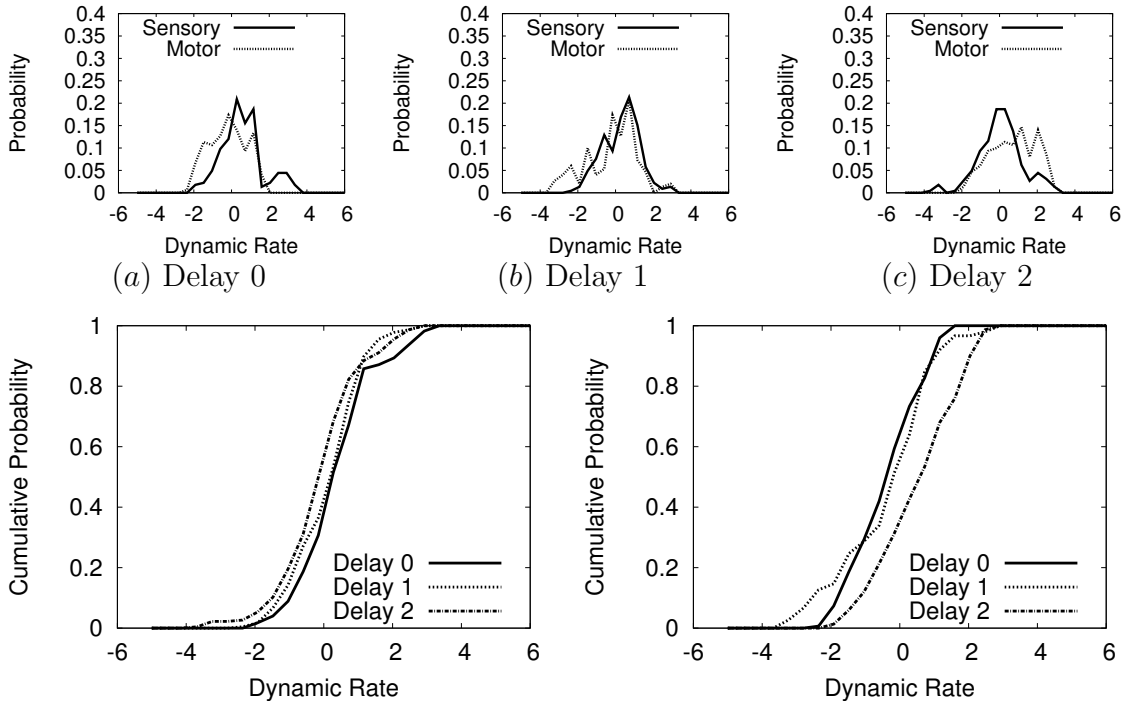


(*a*) FAN (*b*) NDPIA

Fig. 82. Facilitation Under Slow Signal Change Condition. The same function as in Fig. 81 is shown, but in a different interval [20:40]. A higher facilitation rate of $r = 0.9$ was used, to demonstrate the oscillation problem in FAN even under slow signal change conditions. (*a*) Even when the signal changes slowly, if r is high, FAN results in oscillatory activation. (*b*) The proposed dynamics again eliminates the oscillation problem, with the same delay compensation property as in Fig. 81.

weights in the connections in the neural network controller.

Fig. 83 shows the distribution of evolved dynamic rates of top 5 individuals from 25 separate populations (each population had 50 individuals), under three different delay conditions (0, 1, and 2). The motor neurons exhibit higher utilization of facilitating dynamics (high dynamic rate) compared to sensory neurons, when the delay is high (Fig. 83(*c*)). The cumulative distribution of the dynamic rate shows more clearly the positive correlation between increasing input delay and higher dynamic rate in motor neurons (Fig. 83(*e*)) but not in sensory neurons (Fig. 83(*d*)). In sum, motor neurons are more likely to be facilitated, and increasing input delay leads to higher facilitation.



(*a*) Delay 0 (*b*) Delay 1 (*c*) Delay 2

(*d*) Sensory neurons (*e*) Motor neurons

Fig. 83. Dynamic Rate Distribution. The distribution of dynamic rates for the controllers trained with different delay conditions are shown (top 5 individuals from the last generation in 25 separate evolutionary trials). (*a*–*c*) shows the dynamic rate distribution under different delay conditions. As the delay increases, motor neurons increasingly utilizes higher dynamic rate (i.e., facilitation). (*d*-*e*) shows the cumulative distribution, directly comparing the different delay conditions. Only for the motor neurons (*e*), increasing delay shifts the cumulative distribution toward the right (i.e., higher facilitating rates: the mean increases from -0.196 to -0.184 to 0.724).

c. Pole-balancing Performance in FAN and NDPIA

In this experiment, we compared the effectiveness of FAN and NDPIA in the 2D pole-balancing task, under various delay conditions.

To test the effectiveness of facilitation under a strictly controlled environment, we set the facilitation rate to a fixed value of 0.7, with different types of neurons being facilitated in three different sets of experiments. We trained (evolved) 60 FAN networks and 60 NDPIA networks under no delay. Among the 60 networks, 20 were evolved with facilitated motor neurons, 20 networks were evolved with facilitated sensory neurons, and 20 remaining networks were evolved with facilitation in both sensory neurons and motor neurons for each facilitation model (FAN and NDPIA respectively).

For testing, we put the evolved networks under no delay, 1-step delay, and 2-step delay environment to see the effect of dynamic activation rates. Fig. 84 shows that NDPIA has better performance than FAN in most cases, especially under longer delays. Note that if both sensor and motor neurons are facilitated at the same time, no significant difference is found (Fig. 84(c)). There was not much difference when there was no delay in the input (see delay 0 cases in Fig. 84), but the difference of performance becomes clear when motor neurons were facilitated as delay increases (see Fig. 84(b)).

d. Pole-balancing Performance under Extremely High Facilitation Rates in Different Neuron Types

In this experiment, we investigated the effect of extremely high dynamic activation rates under long delay. Controller networks with NDPIA maintained their performance under longer delay, with a fairly high facilitation rate of 0.7, especially for the

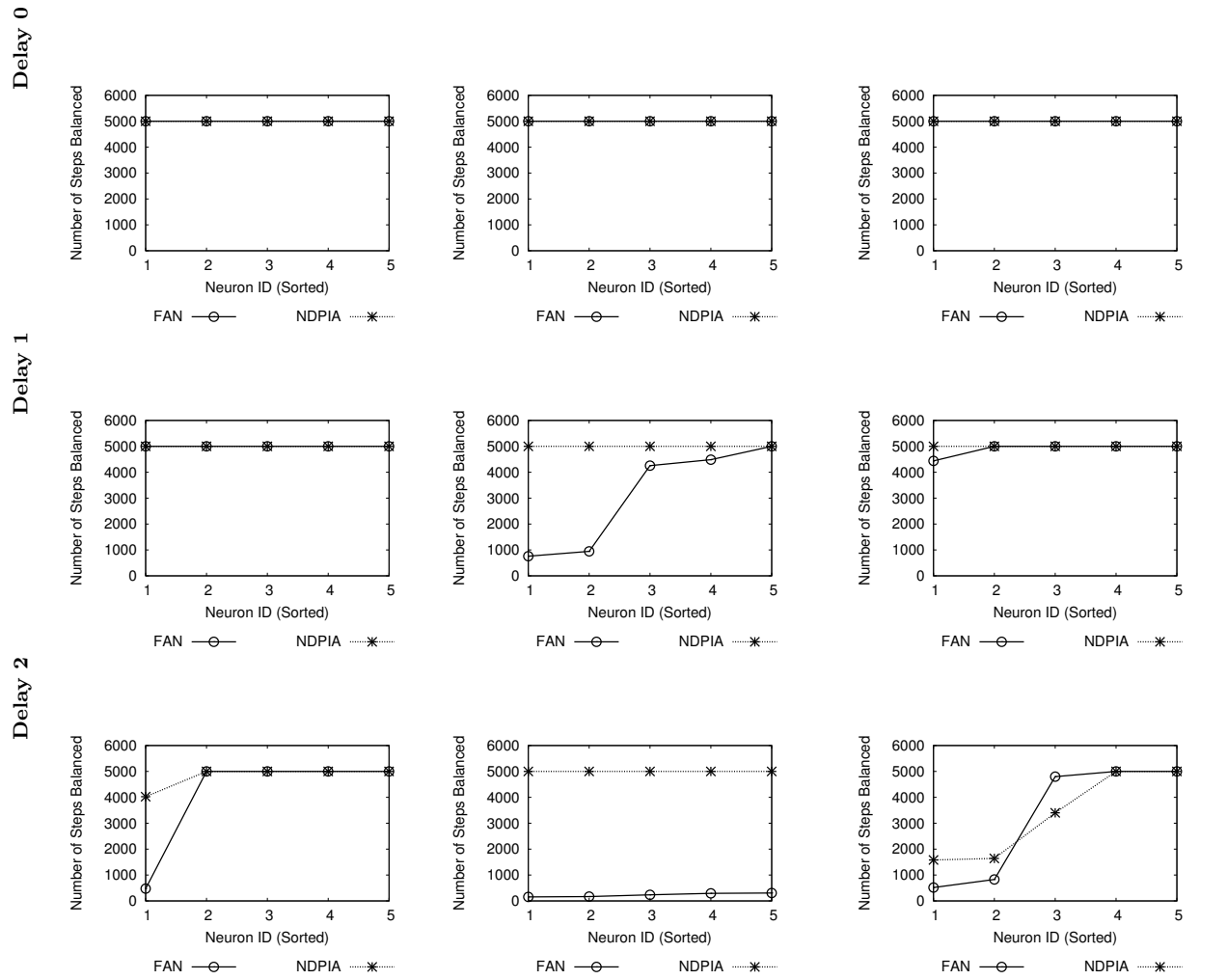
motor-neuron only facilitation (Fig. 84). How would the performance change if we push the facilitation rate to an even higher value? We tested how extreme facilitation like that affects performance when different types of neurons are facilitated: sensory, motor, or both. This is an interesting question since longer delay might necessitate higher facilitation rate.

When either the sensory neurons or the motor neurons were facilitated with a high facilitation rate of 0.7, the performance remained high (a and b), but the performance degraded when both neuron types were facilitated at that rate. As facilitation rate was further increased to even higher values (1.2 to 1.5 to 2.0), performance started to degrade for the case where sensory neurons were facilitated (a), but it was not the case when only motor neurons were facilitated (b). The case with both sensory and motor neurons facilitated showed consistently low performance regardless of the facilitation rate. These results suggest that motor neurons could be the best type to facilitate at higher rates, for the compensation of longer delays (Fig. 85).

5. Discussion

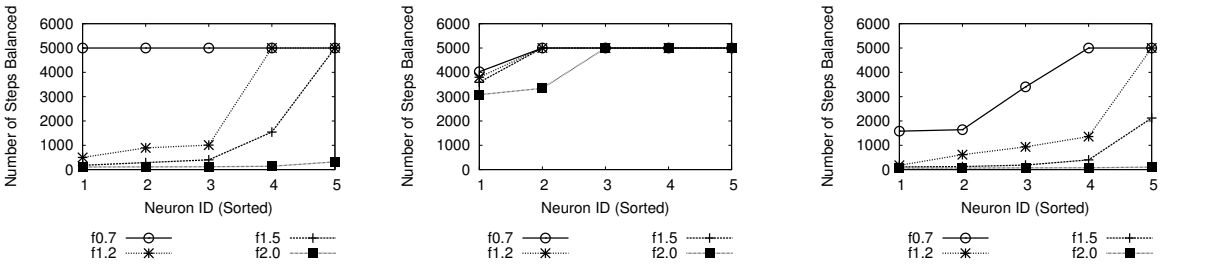
The main contribution of our work is to have shown the link between facilitating neuronal dynamics and delay compensation in a systematic study. In particular we have shown that facilitation is more effective in motor neurons, and that longer input delay leads to higher rates of facilitation. We have also improved the previous facilitation model (FAN) [88, 83, 84] so that higher facilitation rates can be used without side effects (oscillation). As a consequence, our new approach allowed our model to deal with longer delay applied over the entire duration of each trial.

One of the main results of our investigation was that facilitating dynamics is more effective in counteracting delay in certain classes of neurons (i.e., motor neurons). This could be due to two high-level reasons: (1) local connection topology,



(*a*) Sensory neurons facilitated (*b*) Motor neurons facilitated (*c*) Sensory and motor neurons facilitated

Fig. 84. Comparison Of FAN And NDPIA Under Different Delay Conditions And For Different Types Of Neurons Facilitated. (*a*) Top five individuals from the final generation of those with facilitated sensory neurons are shown, for delay 0 (top) to delay 2 (bottom). The number of pole balancing steps in FAN (*) and NDPIA (∘) are plotted. NDPIA shows a slight advantage under longer delay conditions (bottom row). (*b*) The same information is plotted as in (*a*), for top 5 individuals with facilitated motor neurons. NDPIA has better performance than FAN under longer delays (middle and bottom rows). (*c*) The same information is plotted as in (*a*), for top 5 individuals with facilitated sensory and motor neurons. There is no significant difference between FAN and NDPIA.



(a) Sensory neurons facilitated (b) Motor neurons facilitated (c) Sensory and motor neurons facilitated

Fig. 85. The Performance Of Ndpia In 2-Step Delay Under High Facilitation Rates. Performance of top 5 controllers under extremely high facilitation rates are shown for facilitated (*a*) sensory neurons, (*b*) motor neurons, and (*c*) sensory and motor neurons. With a facilitation rate of 0.7, there is minimal degradation of performance in (*a*) and (*b*), but much degradation in (*c*). Also, (*b*) is the only case where very high facilitation rate (= 2.0) does not affect the performance. Thus, when very high facilitation is needed (e.g., to compensate for longer delays), motor neurons should be facilitated.

and (2) series of delay compensation happening in a chain of neurons, and both could cause over-compensation. First, if the local topology of the neuron has a recurrent link, then facilitation could be amplified, thus leading to over-compensation. This is why we believe the sensory neurons did not do well compared to the motor neurons (recall that the sensory neurons are the hidden units that receive not only the input but also the context input from the recurrent connections). Second, if a chain of neurons originating from the input and ending in the output are all facilitated, again it could lead to over-compensation. This could be the reason why the networks with both sensory and motor neurons facilitated performed poorly. These observations can lead to concrete predictions that can be experimentally validated: (1) Facilitating neural dynamics, when employed to perform delay compensation, may be more prevalent where the local connection topology is feedforward. (2) Within a sensorimotor processing chain, facilitating dynamics may be found in only few parts of the chain.

As we have seen already in Fig. 83, in order to compensate for longer delay, higher facilitation should be used. However, since higher facilitation in longer delay may cause higher error rates, this facilitation dynamics is applicable only when the delay is within a certain bound. In other words, if the signal changes more rapidly than the delay duration (or equivalently if the delay duration is longer than the time scale of signal change), our approach may not work well. In order to deal with such situations, sensorimotor anticipation [103] or sensory prediction [89][103] may be needed. To address the challenge of longer delay, internal representations, internal models, or forward models [104][105][106] can also be used. These works suggest the use of anticipation for the future inputs or states in a higher level than at the level of neuronal circuits.

In the beginning, we started out with the insight that predictive function could have originated from delay compensation mechanisms. What our results show is that an intimate relationship exists between longer delay and higher facilitatory dynamics (i.e., the compensation mechanism), and that such compensation mechanisms can emerge during the process of evolution. Since the final outcome of delay compensation is the estimation of the present state (based on information from the past), one might argue that it is not prediction. However, the task itself can be systematically mapped to that of prediction, since it all boils down to the estimation of the state in the relative future, whether that future is now or whenever (see [107]). An interesting future direction is to see if actual predictive capability can evolve in a similar simulated evolution environment, when the task itself requires prediction, rather than just delay compensation. We expect facilitating dynamics to again play an important role in such a case.

Another matter of debate is the biological significance of the proposed facilitating dynamics. As we briefly mentioned in the beginning, part of the motivation for this

work came from the facilitating synapses reported in the experimental literature [108], which provides the feasibility. Also, in return, our proposed mechanism assigns a specific role regarding the function of such synapses. In a similar line, we can ask whether the proposed method is biologically feasible, especially within an individual's lifetime. Although our experiments were done using simulated evolution, it could be seen as just another optimization process, so there is no reason why such a delay compensation mechanism can be developed over time within a single individual's lifetime.

Finally, we would like to take the discussion further and speculate on the role of prediction in brain function. Prediction is receiving increasing attention as a central function of the brain [78][79]. The brain is fundamentally a dynamical system, and understanding the dynamics can lead to deep insights into the mechanisms of the brain. For example, according to Kozma and Freeman [109], the brain state trajectory transitions back and forth between chaotic high-dimensional attractors to periodic low-dimensional attractors. An interesting property of these two different attractors is that for the chaotic attractor, predicting the future state in the state trajectory may be difficult compared to that of the periodic attractor. Such predictive function could form an important necessary condition for more complex and sometimes subjective phenomena as consciousness or self-awareness [76]. It could also be thought of as the "unconscious" process discussed in [110] that drives brain function, that is later confirmed or described by consciousness, post hoc. Prediction can also be useful in other ways, including predicting the upcoming input, based on the current model of the world [111] This kind of mechanism, coupled with emotional circuits (e.g., Levine [112]) could serve as a fundamental component in goal-directed behavior. In sum, how such humble delay compensation mechanisms as presented in this paper can develop into a fully functioning predictive system, laying the foundation for high-level

cognitive processes, is an important future question to be addressed.

6. Conclusions

In this paper, we proposed an improved facilitating dynamics, NDPIA, to address shortcomings in the previous FAN model. We showed that our approach overcomes the limitations and results in higher performance in a standard 2D pole-balancing task, with longer delay in the input during the entire duration of the task. More importantly, we have found that facilitating dynamics is the most effective in motor neurons, and increasing input delay leads to higher utilization of facilitating dynamics. Our findings are expected to help us better understand the role of facilitating dynamics in delay compensation, and its potential development into prediction, a necessary condition for goal-directed behavior.

B. Internal State Predictability

1. Introduction

To build intelligent agents that can interact with their environments and also their own internal states, the agents must identify the properties of objects and also understand the properties of other animated agents [113]. One of the fundamental steps in having such abilities is to identify agents themselves from others. Therefore, finding *self* has been a grand challenge not only among cognitive scientists but also in computer scientists. Even though Feinberg and Keenan strongly suggested that the right hemisphere has a crucial role in the creation of the self [114], localizing the self does not answer many intriguing questions about the concept of the self. On the other hand, a Bayesian self-model that can distinguish *self* from *others* was proposed, and Nico, an upper-torso humanoid robot, was able to identify itself as *self* through the

dynamic Bayesian model using the relationship between its motor activity and perceived motion [115]. Bongard, Zykov, and Lipson made a self-aware robot which can adapt to the environment through continuous self-modeling [116]. We believe that Autonomous Mental Development (AMD) [117] can also lead to a self-model, e.g., as in Self-organizing Autonomous Incremental Learner (SAIL) [117] and Dav [118].

Nico [115] focused more on higher level modeling of self-awareness; SAIL [117] and Dav [118] concentrated on modeling autonomous mental development; and the resilient machine [116] continuously re-modeled its physical body rather than conceptual self. As far as we know, neuronal substrate of *self* has not been discussed. It is not easy to answer the question about the self without invoking complex and controversial issues. However, an alternative way exists to address the problem: If we uncover *necessary conditions* for the emergence of self-awareness, then we might be able to make some progress. An interesting insight is that predictability in the internal state dynamics can be such a necessary condition. We postulate that the *predictability* of the neural activities in the internal dynamics may be the initial stepping stone to self-awareness. Such predictability could lead to authorship (and eventually agency and self-awareness), since a distinct property of one's own actions is that they are always perfectly predictable.

a. Self-awareness

Self-awareness has an important role in cognitive processes [119]. Self-aware system has an ability to distinguish itself from *others*. Being self-aware can be a good beginning to have cognitive capabilities. However why have intelligent agents such as humans evolved to have self-awareness? Is self-awareness simply an evolutionary by-product of self-representation as Menant pointed out [120]? Otherwise, if cognitive agents always have to be self-aware, there must be an associated evolutionary

pressure. However, the attributes of self-awareness is still uncertain [121]. So, it is difficult to track down the roots of the emergence of self-awareness or agency. One way to circumvent the problem is to find circumstances that can serve as necessary conditions for the emergence of self-awareness, and assess their evolutionary value.

In this paper, we focus on finding these necessary conditions. One possible requirement would be the predictability of one's own internal state trajectory (another possibility is direct prediction of one's own action, as in Nolfi et al.'s work on neuroevolution combined with learning [122]). We postulate that Internal State Predictability (ISP) can have a strong impact on performance of the agents, and ISP could have lead to intelligent agents developing self-awareness.

b. Internal State

Many researchers have focused on external environments and behaviors when developing intelligent robots or agents. This was especially true when the investigations were carried out in an evolutionary context.

However, researchers started to take a serious look at the internal dynamics of an intelligent agent as well. The central nervous system models sensorimotor dynamics, and the model seems to reside in the cerebellum [123]. Exploring one's internal state can lead to a sense of self. The sense of self may be a prerequisite to building a machine with consciousness [124].

There may be a consensus that neuronal activation levels can be considered as the state of a neural system. Bakker and de Jong pointed out that the state of a neural network could be defined by the current activation levels of the hidden units [125]. Also, the system state could be viewed as consciousness, in a way [126]. There are also physiological arguments about this idea. The firing rate of each neuron in the inferior temporal visual cortex tells much about the stimuli applied to the cortex [126]. On

the other hand, spiking activities from place cells in the hippocampus can be used to rebuild certain features of the spatial environment [127]. These results tell us that spiking patterns of neurons that form one's internal state might influence task performance. In sum, knowing internal state of oneself may be the first step of being conscious and internal state itself can be simply stated as spiking patterns of neurons during task performance.

The idea that self-awareness has evolutionary advantages is not new [120]. Menant hypothesized that noticing agony of conspecifics may be the first step in developing self-awareness. But as far as we understand, the precondition of identifying agony in conspecifics is self-awareness and the identification of agony is also a requirement of being self-aware. It falls into a circular argument. Namely, self-awareness is a requirement of identifying agony and also, identifying agony develops self-awareness. Moreover, Menant's argument is more like a hypothesis, without giving plausible evidence.

We present experimental results that indicate "understanding" internal states has an actual evolutionary benefit.

2. Method

We hypothesized that activation values from neurons in the hidden layer can represent the internal state of an agent. Understanding one's own internal state can be strongly linked to knowing what is going to happen in one's internal state. We quantified such an understanding as the predictability of the internal state trajectories.

In order to examine whether internal state predictability has any evolutionary value, we evolved sensorimotor control agents with recurrent neural network controllers. The neural activity in the hidden layer of the network was viewed as the internal state of an agent. A two-degree-of-freedom (2DOF) pole balancing task was

chosen to scrutinize the internal state trajectories. The neural network controllers were trained by a neuro-evolution method. The activation values from each neuron in the hidden layer from the neural network were stored to measure the predictability of each neuron's activation trace over time. The predictability of each neuron was quantified by a supervised learning predictor which forecasted the next activation value based on the past activations. Note that any reasonable predictor can be used for this, e.g. Hidden-Markov models, and the choice is orthogonal to the main argument of this paper.

a. Two-Degree-of-Freedom Pole Balancing

Pole balancing task has been used to demonstrate complex and unstable nonlinear dynamical systems in the field of artificial neural networks for decades because it is straightforward to understand and easy to visualize. A conventional 1D pole balancing task has dealt with the following situation: A pole is hinged atop a cart that travels along a single straight line track. The pole can only move on the vertical plane along the track [128, 129]. It makes the task simple enough to be analyzed, but it is not complex enough to show interesting behavior. Here, we used 2D pole balancing where force to the cart can be applied in both the x and the y directions, so the cart moves around on a 2D plane within a boundary and a pole attached on top of the cart can fall in any direction [92, 130]. As a result, the task is more complex and difficult to master than 1D version. Fig. 86 shows a 2D pole-balancing system with which we conducted our experiment. (Removing velocity information from the 1D problem could make the task more difficult and thus more interesting, but we did not pursue this direction.)

The state of the cart (the gray circle on the bottom Fig. 86) with a pole on top is characterized by the following physical parameters: The cart position in the

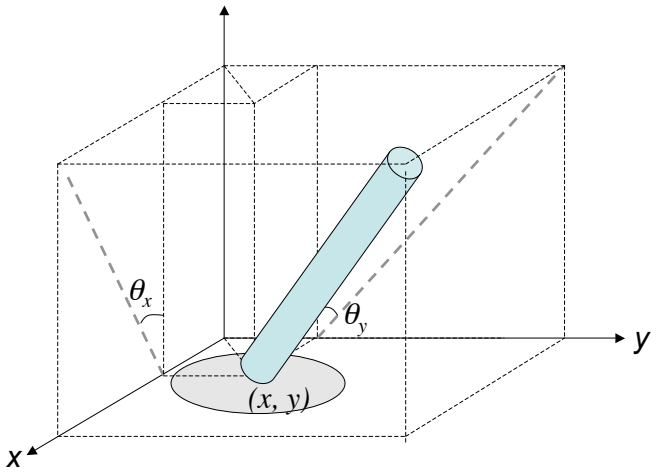


Fig. 86. Two-Degree-Of-Freedom Pole-Balancing System.

plane (x, y), the velocity of the cart $(\dot{x}, \dot{y})$, the angle of the pole from the vertical in the x and the y directions (θ_x, θ_y), and their angular velocities $(\dot{\theta_x}, \dot{\theta_y})$ [92]. These parameters were used as eight input values to a neural network. Fourth-order Runge-Kutta method was used to simulate the real world physics.

b. Time Series Prediction

A time series is a sequence of data from a dynamics system. The measurements of one or more variables of the system take place at a successive and regular time interval [131]. The system dynamics changes the state over time, so it can be considered as a function of the current state vector $x(t)$. A time series is a sequence of either vectors or scalars.

$$\{x(t_0), x(t_1), \cdots, x(t_i), x(t_i), x(t_i + 1), \cdots\}$$

The activation level of hidden neurons in a neural network can be considered as a time series. In our case, three sets of time series exist since there are three neurons in the hidden layer of our neural network. Let us assume that we predict value x at

time $t+1$ which is the very next state from present. If we can look back N time steps including the current one from time t, we can say that forecasting $x(t+1)$ means finding a function $f(\cdot)$ using a time series $\{x(t), x(t-1), x(t-2), \cdots x(t-N+1)\}$(Fig. 87):

$$\hat{x}(t+1) = f\left(x(t), x(t-1), x(t-2), \cdots, x(t-N+1)\right).$$

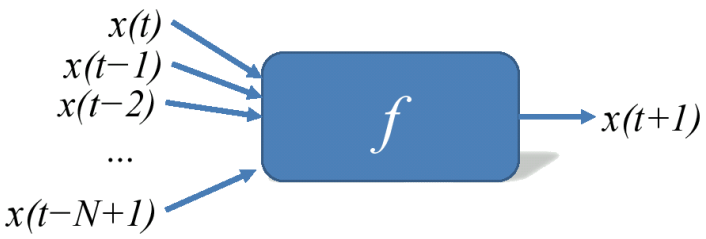


Fig. 87. Predicting Future Using The Past.

Feed-forward neural networks have been widely used to forecast a value given a time series dataset [131]. The neural predictors use a set of N data as inputs, and a single value as an output for the target of the network. The number of input data is often called the sliding window size [131]. Fig. 88 gives the basic architecture of a feed-forward neural network predictor.

When a neural network predictor forecasts a future state, the outcome of the predictor, which is a predicted value, should be compared with a real activation value. If a prediction error, the difference between a predicted and a real value, is greater than a certain amount (we call it minimum error threshold) then it is fair to say the prediction had failed. However, we cannot use a fixed minimum error threshold, because amplitude envelope of activation values could be different from neuron to neuron. Why does the envelope of activation matter?

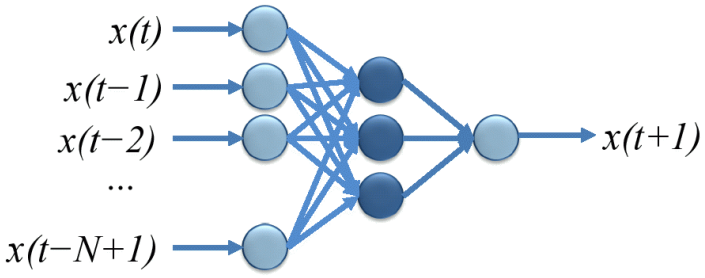
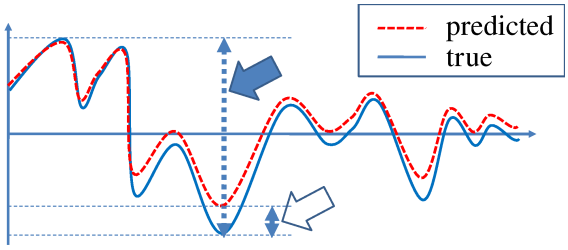
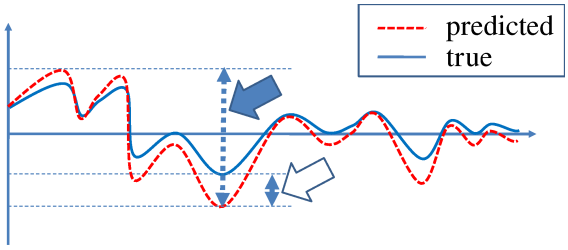


Fig. 88. A Neural Network Predictor For A Time Series.



(a) Big amplitude values and the amount of error



(b) Small amplitude values and the amount of error

Fig. 89. An Example Of Adaptive Error Rates. The big solid arrows indicate the amplitude of activation values, and the hollow big ones indicate the amount of error. When estimating the error, the activation amplitude envelope should be considered.

As Fig. 89 shows, it cannot be stated that two cases have the same amount of error, although the actual amount of error is almost the same in (a) and (b). The minimum error threshold value should be adapted to the variance of the time series. Namely, if the amplitude of a time series is also small, the minimum error threshold should be small, and if it is large, then the threshold should become large as well.

$$Err_{th} = |ActValue_{max} - ActValue_{min}| \times R,$$

where Err_{th} is the minimum error threshold, $ActValue$ means an activation value of a neuron in a hidden layer, and R is the adaptive rate for the minimum error threshold adjusted based on the activation amplitude as shown in Fig. 89.

3. Experiments and Results

We evolved agents driven by a recurrent neural network in a 2D pole balancing task, and then partitioned successful individuals into two groups: One group had high ISP, and the other had low ISP. The high-ISP group showed better performance than the low ISP group in tasks with harsher initial conditions.

a. Training the Controllers

We implemented the pole balancing agent with a recurrent neural network controller. The artificial neural networks were trained by genetic algorithms. Network connection weights of an agent were evolved to balance the pole during the training sessions.

Force between $-10N$ and $10N$ was applied at regular time intervals (10 millisecond). The pole was 0.5 meter long and was initially tilted by 0.573° (0.01 radian) on the x-z plane and the y-z plane respectively. The area where the cart moved around was $3 \times 3\ m^2$.

The configuration of a controller network was as follows: eleven input nodes

(eight input values from the simulated physical environment and three context input values from the hidden layer), one hidden layer with three neurons, and two output neurons (Fig. 90). The eight parameters describing the current state of the cart were used as the input values, and two values from the output neurons, F_x and F_y, represented the force in the x and the y direction.

Fig. 90 shows the recurrent network that we used in the experiments.

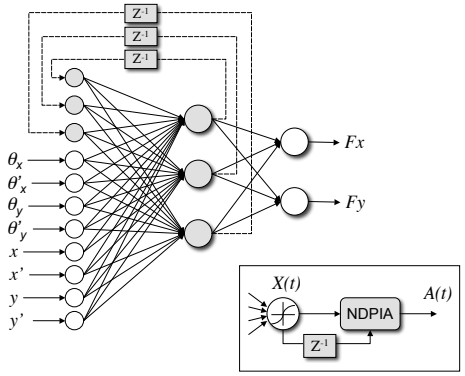


Fig. 90. Recurrent Neural Network Controller For 2D Pole-Balancing. The signal flow inside each neuron is shown in the box. Z^{-1} means unit delay. [75]

In training these non-linear controllers, neuroevolution methods have proved to be efficient [93, 94]. Fitness was determined by the number of time steps where a network was able to keep the pole within $\pm 15^{\circ}$ from the vertical in the x and the y directions and kept the cart within the 3×3 m^2 area. The chromosome encoded the connection weights between input and hidden layer neurons, and between hidden and output neurons. Crossover occurred with probability 0.7 and the chromosome was mutated by ± 0.3 perturbation rate with probability 0.2. The force was applied in both the x and the y directions at 10 millisecond intervals. The number of networks in a population was 50 for an evolutionary epoch. If an agent balanced the pole more than 5,000 steps, we considered it as a success.

b. Training the Neural Network Predictors

Neuronal activities in the hidden layer of the recurrent neural network were viewed as the internal state of the agent. The predictability in the internal state trajectory was able to be measured using a feed forward neural network predictor.

The size of the sliding window was four. The activation values of neurons in the hidden layer formed the network input. 3,000 activation values were used as training data for each input, and a test set used the next 1,000 steps (3,001 to 4,000). Time series from 1 to 1,000 steps and from 4,001 to 5,000 steps were not used because we did not want to use the somewhat chaotic initial movements and finalized stable movements. Back-propagation algorithm was used to train the predictors (learning rate 0.2). In the test sessions, we compared the predicted value with the real activation value. We chose 10% threshold error rate to calculate the adaptive minimum error threshold when comparing the forecasted activation with the real activation value. The adaptive error rate was determined empirically, based on the performance of the time series predictor.

c. Performance Measurement in High- vs. Low-ISP Groups

We evolved approximately 130 pole balancing agents. By definition, all the agents were necessarily good pole balancers during the training phase. Some of them turned out to have high ISP and others low ISP. Fig. 91 shows all the agents sorted by their prediction success rates.

We chose pole balancers having top 10 highest ISPs, and bottom 10 lowest ISPs. High predictability means all three neurons from the hidden layer have highly predictable internal state trajectories. Most of their prediction rates in a high ISP group were over 99%, and only two pole balancers had average prediction success rates

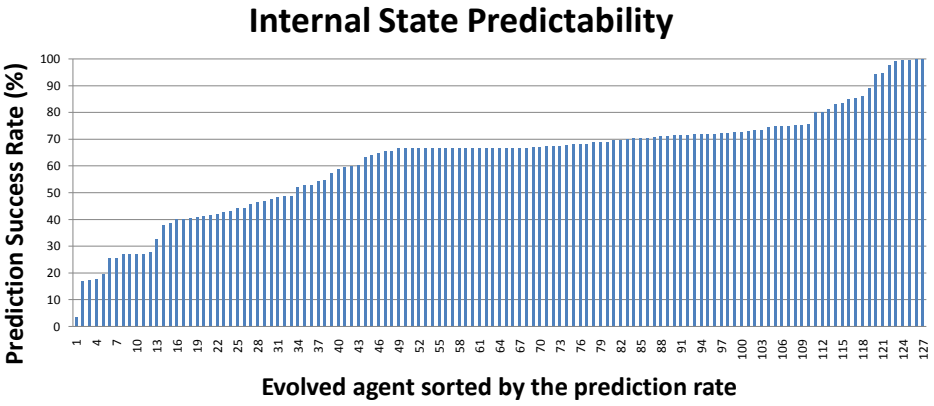


Fig. 91. All Trained Agents Sorted By Their Prediction Success Rates. A small number of agent toward the left end show very low predictability, while those near the right end show very high predictability.

83.30% and 88.93% ($\mu = 95.61\%$ and $\sigma = 5.55\%$). As for low ISP pole balancers, their average prediction performances from the three neurons were between 17.37% and 48.53% ($\mu = 31.74\%$ and $\sigma = 10.79\%$). Fig. 92 shows the predictability in the high and the low ISP group.

The learning time of the two different groups was also investigated, but we could not find a significant difference, even though low ISPs took slightly less time than high ISP (Fig. 93). Note again that the performance of both groups (high and low ISP) were comparable during the evolutionary trials.

In order to further test and compare the performance between the two groups, we made the initial condition in the 2D pole balancing task harder than that during the training phase. All the neural network controllers were evolved in a condition where both projected initial angles to the x-z and the y-z plain were 0.573° (0.01 radian). In the test condition, we had those trained neural network controllers balance the pole in a more difficult initial condition where the initial projected angles were 4.011° (0.07 radian) on the $x - z$ plain, and 2.865° (0.04 radian) on the y-z plain. This strategy

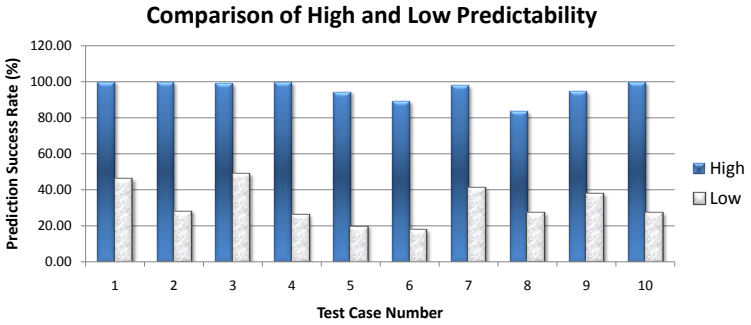


Fig. 92. A Comparison Of The Average Predictability From Two Groups: High ISP And Low ISP. The predictive success rate of the top 10 and the bottom 10 agents from Fig. 91 are shown. Each data point plots the mean predictive success rate of three hidden neurons of each agent. The error bars indicate the standard deviation.

was used to push the controllers to the edge so that differential behavior results. Our main results were that networks with higher ISP show better performance in harder tasks than those with lower ISP.

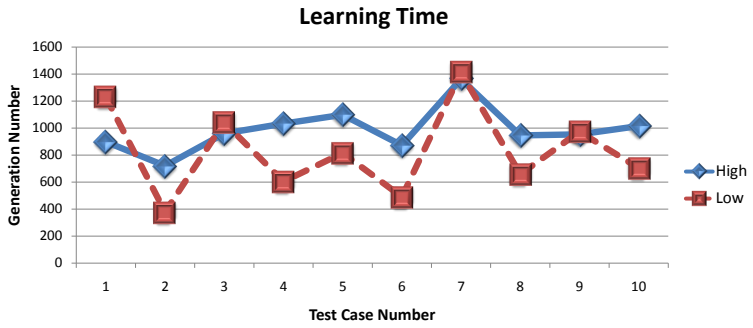


Fig. 93. Learning Time In High Vs. Low ISP Groups. Agents are ordered in the same order as in Fig. 92.

Fig. 94 shows that the evolved pole balancers with higher ISP have better performance than the other group.

One might argue that this result seems straightforward, because simple internal state trajectories simply reflect behavioral properties. A trivial solution to a pole

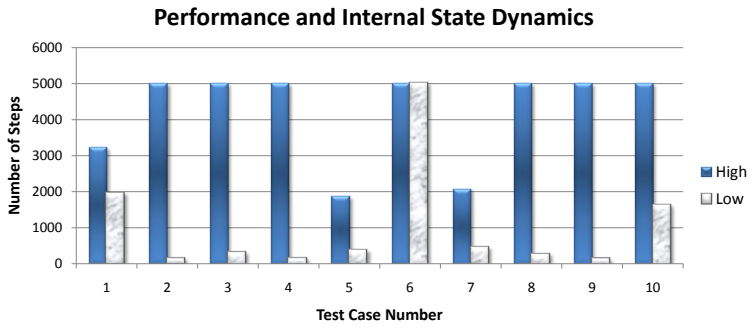


Fig. 94. Novel Task Performance In High Vs. Low ISP Groups. Each test case (with low vs. high ISP results) corresponds to an experiment ran with identical initial condition, so the pairing is meaningful.

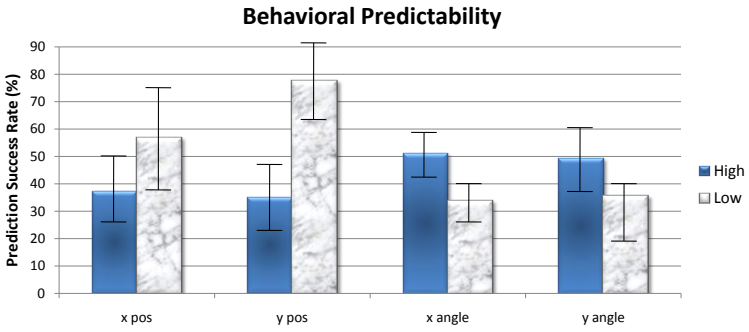


Fig. 95. Mean Behavioral Predictability (error bars indicate the standard deviation).

balancing problem would be to quickly make the pole stand up vertically, and then make minimal adjustments. But according to our experimental results (see Fig. 95), higher ISP does not necessarily mean that their behavioral trajectory is also simple. Fig. 96 (compared to Fig. 98) and Fig. 97 (compared to Fig. 99) show that behavioral complexity is not necessarily directly related to the complexity of internal states. That is, even an agent having high ISP may have complex behavioral properties, and those with low ISP may have simple behavioral attributes.

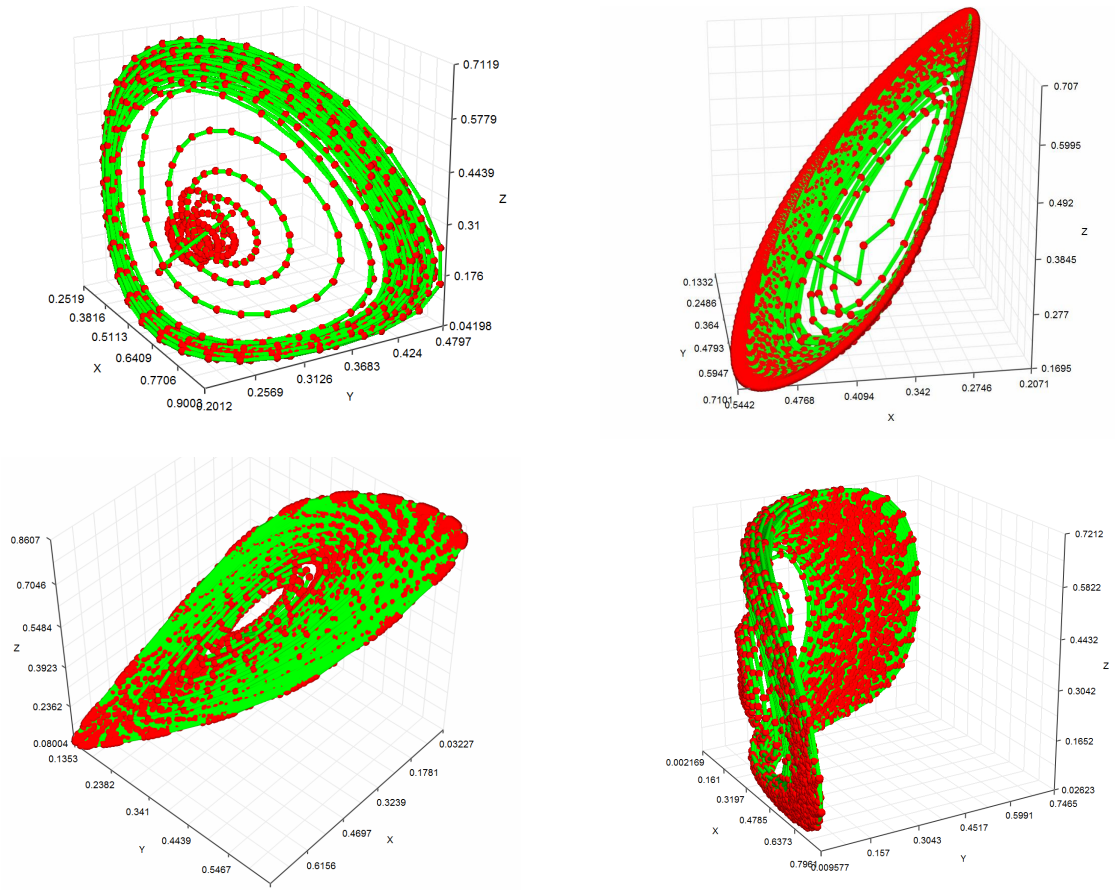


Fig. 96. Examples Of Internal State Dynamics From The High Isp Group Showing Smooth Trajectories.

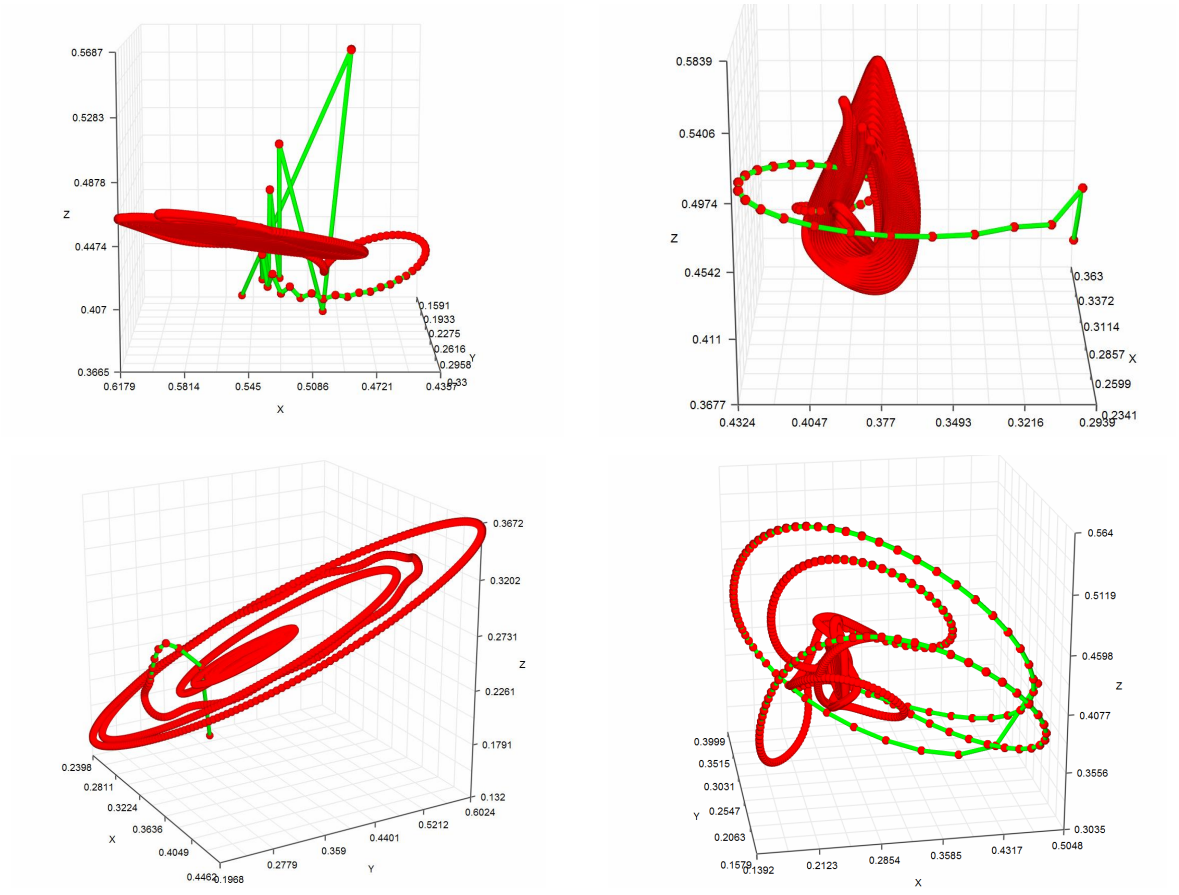
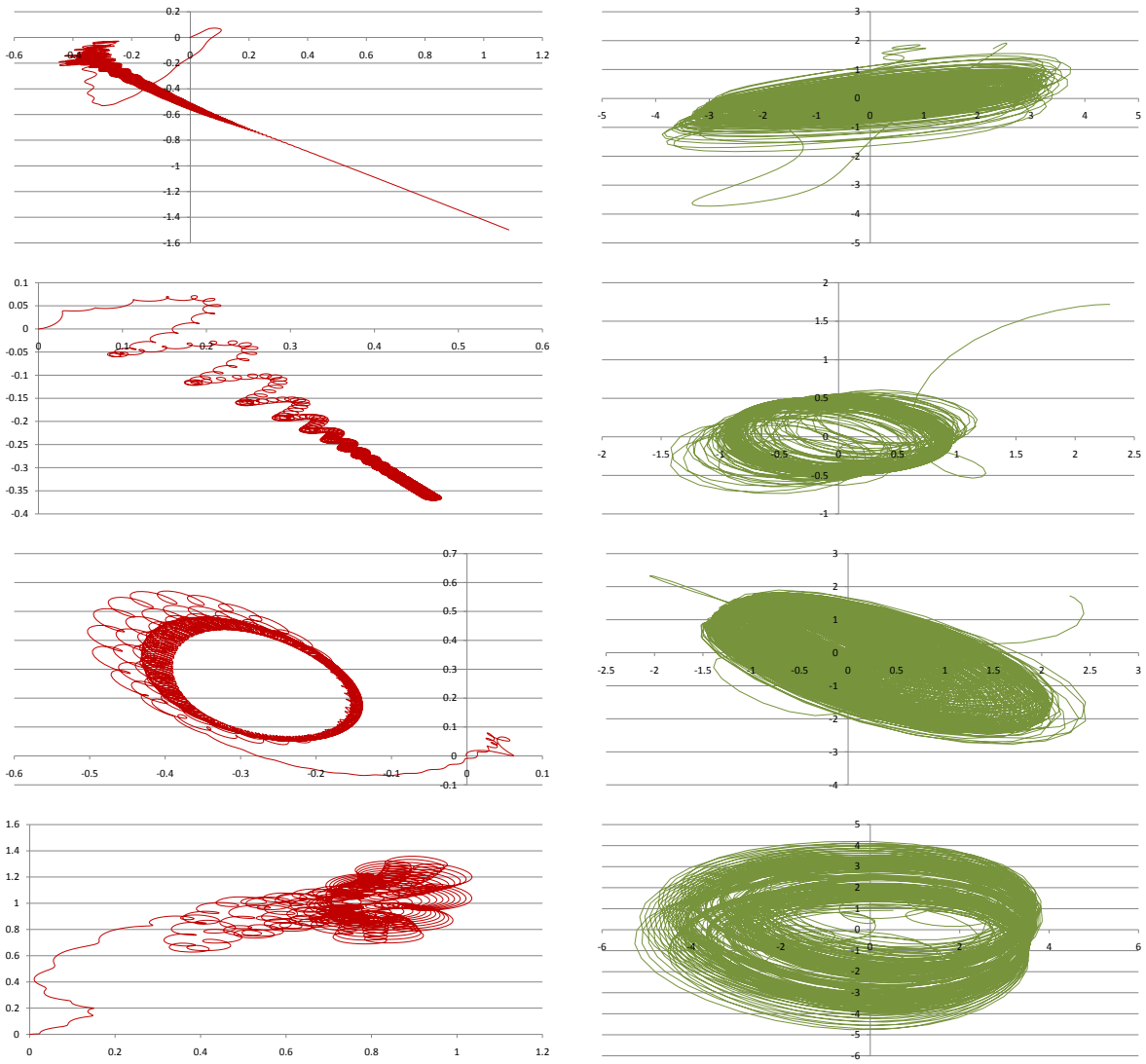


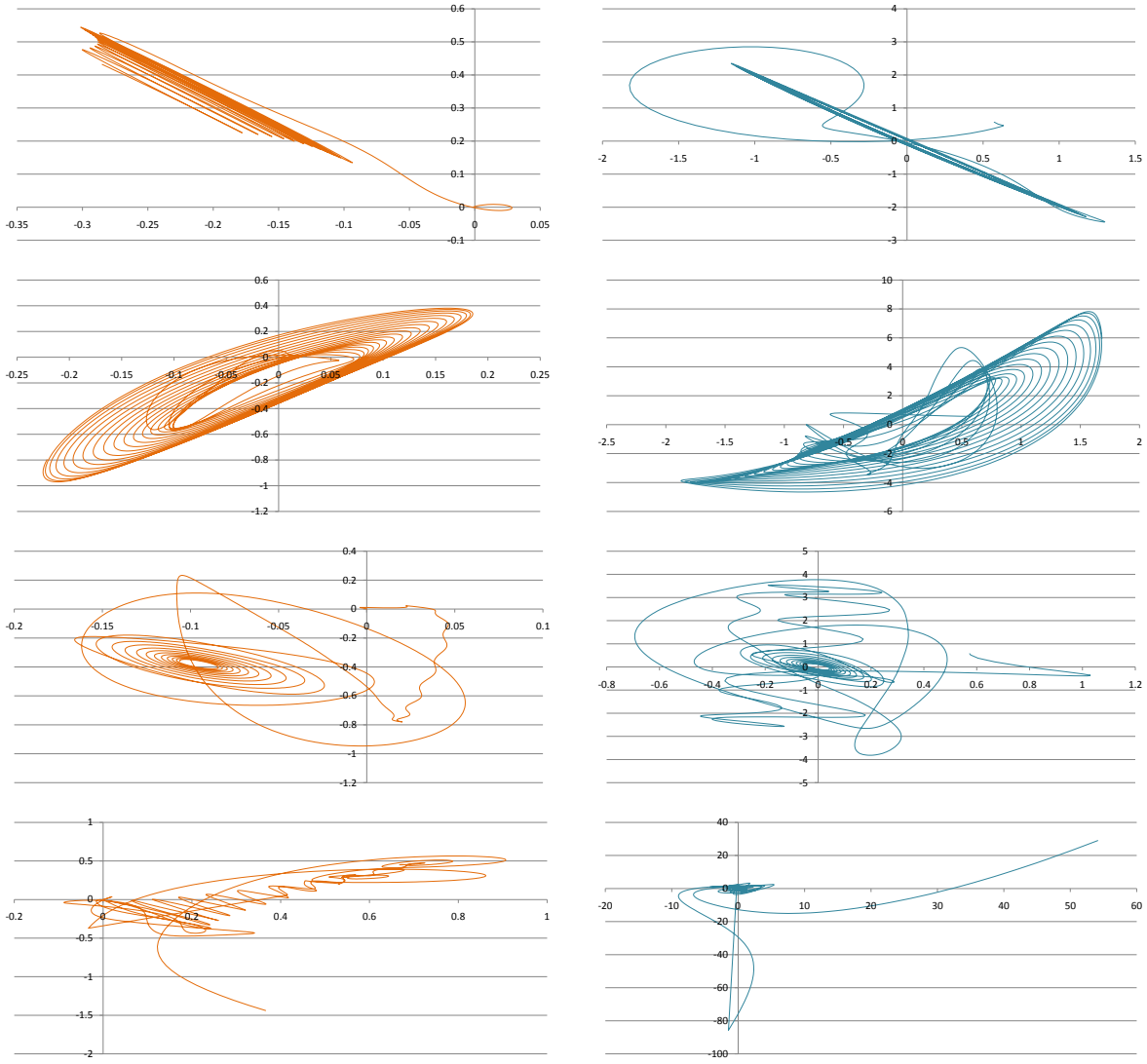
Fig. 97. Examples Of Internal State Dynamics From The Low Isp Group Showing Abrupt, Jittery Trajectories.



(a) Cart x, y position

(b) Pole angle

Fig. 98. Examples Of Behavioral Trajectories Of x, y Positions And Pole Angles From The High ISP Group Exhibiting Complex Trajectories.



(a) Cart x, y position

(b) Pole angle

Fig. 99. Examples Of Behavioral Trajectories Of x, y Positions And Pole Angles From The Low ISP Group Exhibiting Complex Trajectories.

4. Conclusion

Starting with individuals evolved to give the same level of behavioral performance, we showed that those with simpler (more predictable) internal dynamics can achieve higher levels of performance in harsher environmental conditions. These results suggest that internal agent properties such as simpler internal dynamics may have a survival value. We also showed that the increased survival value is not always due to smoother behavior resulting from the simpler internal states. The implication of these findings is profound, since they show that, in changing environments, apparently circumstantial internal agent properties can affect external behavioral performance and fitness. The results also show how an initial stepping stone (or a necessary condition) in the evolutionary pathway leading to self-awareness and agency could have formed. We expect the framework we developed here to help us better address hard issues such as self-awareness and agency in an evolutionary context. Future directions include evolution of model-based prediction of both internal and external dynamics (cf. [116][122]), and generalization of our framework to other more complex tasks.

CHAPTER X

DISCUSSION

A. Summary

In this dissertation, I described the data acquisition and mining of the whole mouse brain microstructure. A new high-throughput and high-resolution microscopy technique, Knife-Edge Scanning Microscopy (KESM) was introduced as an ideal data acquisition method. KESM is capable of generating high-resolution data sets from large tissue volumes much faster than any other known methods currently available. Yet, its full potential was not realized due to many steps that required maximal automation.

To address the problem, I introduced methods to automate the whole process involved in the data acquisition. I also provided proper software frameworks throughout the data processing pipeline through which the data flow. These include a parallel processing framework, an automatic method for alignment of image chunks in an image stack, removal of intensity irregularities, a hierarchical data representation framework, and automatic data analysis tools.

Finally, I conducted two computational neuroscience case studies to discuss the utility of the data sets from the KESM. First, facilitatory neural dynamics was introduced as a possible mechanism to compensate for delay in transmission of neuronal signals. Second, internal state predictability was investigated as a precursor of emergence of self-awareness.

The techniques and studies discussed in this dissertation are expected to greatly enhance our ability to explore microstructures in the brain in three-dimensions. The data acquisition methods that I automated allows us to scan three-dimensional tissue

volumes much faster than other sectioning and imaging technologies. Automatic data analysis methods can shed new light on large-scale data investigation by proposing much easier exploration of highly complex network structures.

B. Future Work

In this section, I discuss possible future work regarding my research.

1. Data Validation

A large-scale data validation framework is a possible extension of my research. Currently, the BNL does not have proper validation methods by which we can measure accuracy of the reconstruction and the three-dimensional structure tracing. Recursive and iterative refinement of the ground true data could be one of the options.

2. Data Dissemination

Currently all data reside inside local storage (hard drive) of the KESM server. If such large-scale databases of tissue volumes can be accessible from other researchers who have various data sets which can be compared with the KESM data sets, it could have a big impact on the research community since experts from a wide variety of areas can share their insights regarding the data sets.

3. Data Simulation

Structural information in three-dimensions can also be used as input for simulation of specific tissue function. Simulating blood flow can be a good example by using the traced fibrous data sets.

4. Full-scale Data Analysis

The ultimate goal of figuring out structural information is to understand brain function related to its structure. In order to achieve the goal, large-scale (possibly full-scale) data analysis is necessary. Tracing micro structures in three-dimensional volumes involves a lot of manual work. Consequently, those time consuming tasks should be effectively removed and replaced by automated algorithms to effectively analyze large-scale data sets.

CHAPTER XI

CONCLUSION

The mouse brain microstructure requires very large image volumes of tissue containing great detail. Accurate estimation of morphological parameters is a key in building exact models of connection matrices. Considering the massive amounts of data, automation of sectioning and imaging is a crucial factor. Given these conditions, high-resolution and high-throughput data acquisition systems become mandatory. To address the challenge, I proposed an automated technique for physical sectioning microscopy to acquire large volumes of neuronal microstructure data. In order to provide a method that can be used in analyzing the data and reconstructing their morphological properties, a software framework has been developed. The framework allows us to properly interpret both the reconstructed geometric data and raw image volumes of anatomical microstructure data sets, and also helps us analyze the data sets more accurately and completely.

The obtained KESM data may be used in validating neuroscience hypotheses. Through two case studies, I have shown how the structural properties acquired from the KESM data could be used in testing assumptions and results in neuroscience research.

I expect the KESM data, tools, and my case studies to initiate a new area of investigation in computational neuroscience.

REFERENCES

[1] J. Horgan, "The brain: The final frontier of science," Globe and Mail, April 2000, http://www.theglobeandmail.com/. Accessed on May 2009.

[2] J. Horgan, "The final frontier," October 2006, http://discovermagazine.com /2006/oct/cover. Accessed on May 2009.

[3] O. Sporns and G. Tononi, "Classes of network connectivity and dynamics," *Complexity*, vol. 7, no. 1, pp. 28–38, 2002.

[4] C. M. Pechura and J. B. Martin, Eds., *Mapping the Brain and Its Functions: Integrating Enabling Technologies into Neuroscience Research*, Washington, D.C.: National Academy Press, 1991.

[5] J. H. McCarty, "Cell biology of the neurovascular unit: Implications for drug delivery across the blood-brain barrier," *ASSAY & Drug Development Technologies*, vol. 3, no. 1, pp. 89–95, 2005.

[6] L. W. Swanson, *Brain Maps (Structure of the Rat Brain)*, Amsterdam: Elsevier Publishing Company, December 1992.

[7] 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. 936–949, 2008.

[8] F. Cassot, F. Lauwers, C. Fouard, S. Prohaska, and V. Lauwers-Cances, "A novel three-dimensional computer-assisted method for a quantitative study of microvascular networks of the human cerebral cortex," *Microcirculation*, vol. 13, no. 1, pp. 1–18, January 2006.

[9] S. L. Bressler, "Large-scale cortical networks and cognition," *Brain Research Reviews*, vol. 20, pp. 288–304, 1995.

[10] M. J. Ackerman, T. Yoo, and D. Jenkins, "From data to knowledge the visible human project continues," *Medinfor*, vol. 10, no. 2, pp. 887–890, 2001.

[11] "The visible human project," National Library of Medicine, February 2009, http://www.nlm.nih.gov/research/visible/visible_human.html. Accessed on May 2009.

[12] "The human genome project and beyond," The Office of Biological and Environmental Research of the U.S. Department of Energy Office of Science, 2003, http://genomics.energy.gov. Accessed on May 2009.

[13] R. L. Sidman, B. Kosaras, B. Misra, and S. Senft, "High resolution mouse brain atlas - research goal," 2009, http://www.hms.harvard.edu/research/brain/goal.html. Accessed on May 2009.

[14] G. A. Burns and M. P. Young, "Analysis of the connectional organization of neural systems associated with the hippocampus in rats," *Philosophical Transactions of the Royal Society B: Biological Sciences*, vol. 355, no. 1393, pp. 55–70, January 2000.

[15] "The Allen Institute for Brain Science , Allen Brain Atlas.," 2009, http://www.alleninstitute.org/. Accessed on May 2009.

[16] G. D. Rosen, A. G. Williams, J. A. Capra, M. T. Connolly, L. L. Brian Cruz, D. C. Airey, A. Kulkarni, and R. W. Williams, "The mouse brain library @

www.mbl.org," Narita, Japan, 2000, 14th International Mouse Genome Meeting.

[17] G. M. G. Shepherd, A. Stepanyants, I. Bureau, D. Chklovskii, and K. Svoboda, "Geometric and functional organization of cortical circuits," *Nature Neuroscience*, vol. 8, no. 6, pp. 782–790, 2005.

[18] A. Chenn, "Making a bigger brain by regulating cell cycle exit," *Science*, vol. 298, no. 5594, pp. 766–767, 2002.

[19] S. Russell, "Of mice and men: Striking similarities at the DNA level could aid research," San Francisco Chronicle, December 5, 2002.

[20] B. Zlokovic, "The blood-brain barrier in health and chronic neurodegenerative disorders," *Neuron*, vol. 57, pp. 178–201, 2008.

[21] C. Iadecola, "Neurovascular regulation in the normal brain and in alzheimer's disease," *Nature Reviews Neuroscience*, vol. 5, pp. 347–360, May 2004.

[22] C. I. Moore and R. Cao, "The hemo-neural hypothesis: On the role of blood flow in information processing," *Neurophysiology*, vol. 99, pp. 2035–2047, October 2007.

[23] G. C. Petzold, D. F. Albeanu, T. F. Sato, and V. N. Murthy, "Coupling of neural activity to blood flow in olfactory glomeruli is mediated by astrocytic pathways," *Neuron*, vol. 58, no. 6, pp. 897 – 910, June 2009.

[24] G. J. del Zoppo, "Stroke and neurovascular protection," *The New England Journal of Medicine*, vol. 354, no. 6, pp. 553–555, February 2006.

[25] "The other brain cells," Genetic Science Learning Center, 2006, http://learn.genetics.utah.edu/units/addiction/reward/cells.html. Accessed on May 2009.

[26] M. Zonta, M. C. Angulo1, S. Gobbo, B. Rosengarten, K.-A. Hossmann, T. Pozzan1, and G. Carmignoto1, "Neuron-to-astrocyte signaling is central to the dynamic control of brain microcirculation," *Nature Neuroscience*, vol. 6, no. 1, pp. 43–50, January 2003.

[27] "Introduction to connectomics," 2007, http://hebb.mit.edu/courses/connectomics/. Accessed on May 2009.

[28] A. Madrigal, "Mapping the most complex structure in the universe: Your brain," January 2008, http://www.wired.com /science/discoveries/news/2008/01/connectomics. Accessed on May 2009.

[29] O. Sporns, "Connectome," http://www.scholarpedia.org/article/Connectome. Accessed on May 2009.

[30] S. Seung, "Connectomics," http://dana.org/news/cerebrum/detail .aspx?id=13758. Accessed on May 2009.

[31] J. W. Lichtman, J. Livet, and J. R. Sanes, "A technicolour approach to the connectome," *Nature Reviews Neuroscience*, vol. 9, pp. 417–422, 2008.

[32] O. Sporns, G. Tononi, and R. Kötter, "The human connectome: A structural description of the human brain," *PLoS Computational Biology*, vol. 1, no. 4, June 2005.

[33] J. G. White, E. Southgate, J. N. Thomson, and S. Brenner, "The structure of

the nervous system of the nematode c.elegans (the mind of a worm)," *Philosophical Transactions of the Royal Society of London*, vol. 314, pp. 1–340, 1986.

[34] H. Markram, "The blue brain project," *Nature Reviews Neuroscience*, vol. 7, pp. 153–160, 2006.

[35] "Blue gene," http://domino.research.ibm.com/comm/research_projects.nsf /pages/bluegene.index.html. Accessed on May 2009.

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

[37] J. B. Pawley, *Handbook of Biological Confocal Microscopy*, New York: Plenum Press, 1995.

[38] W. Denk, J. H. Strickler, and W. W. Webb, "Two-photon laser scanning fluorescence microscopy," *Science*, vol. 248, no. 4951, pp. 73–76, 1990.

[39] H. Fan, H. Fujisaki, A. Miyawaki, R. K. Tsay, R. Y. Tsien, and M. H. Ellisman, "Video-rate scanning two-photon excitation fluorescence microscopy and ratio imaging with cameleons," *Biophys*, vol. 76, no. 5, pp. 2412–2420, May 1999.

[40] P. S. Tsai, B. Friedman, A. Ifarraguerri, B. D. Thompson, V. Lev-Ram, C. B. Schaffer, Q. Xiong, R. Y. Tsien, J. A. Squier, and D. Kleinfeld, "All-optical histology using ultrashort laser pulses," *Neuron*, vol. 39, pp. 27–41, 2003.

[41] K. D. Micheva and S. J. Smith, "Array tomography: A new tool for imaging the molecular architecture and ultrastructure of neural circuits," *Neuron*, vol. 55, no. 1, pp. 25–36, 2007.

[42] W. Denk and H. Horstmann, "Serial block-face scanning electron microscopy to reconstruct three-dimensional tissue nanostructure," *PLoS Biology*, vol. 19, no. 11, pp. e329, 2004.

[43] K. J. Hayworth, N. Kasthuri, R. Schalek, and J. W. Lichtman, "Automating the collection of ultrathin serial sections for large volume tem reconstructions," *Microscopy and Microanalysis*, vol. 12, no. Suppl 2, pp. 86–87, 2006.

[44] B. H. McCormick, "System and method for imaging an object," US Patent 6744572, 2004.

[45] B. H. McCormick, "The knife-edge scanning microscope," Technical report, Department of Computer Science and Engineering, Texas A&M University, 2003.

[46] D. Mayerich and B. H. McCormick, "Three-dimensional imaging using knife-edge scanning microscopy," in *Proceedings, Microscopy and Micro-analysis Conference*, 2004, vol. 10, pp. 1466–1467.

[47] Lichtman, "Atlum project," http://www.mcb.harvard.edu/lichtman/ATLUM /ATLUM_web.htm. Accessed on May 2009.

[48] J. Kwon, D. Mayerich, J. Keyser, and Y. Choe, "Lateral sectioning for knife edge scanning microscopy," in *International Symposium on Biomedical Imaging*, Paris, France, 2008.

[49] D. Mayerich, B. H. McCormick, and J. Keyser, "Noise and artifact removal in knife-edge scanning microscopy," in *IEEE International Symposium on Biomedical Imaging: From Nano to Macro*, 2007, pp. 556–559.

[50] "Insight segmentation and registration toolkit (itk)," http://www.itk.org/. Accessed on May 2009.

[51] "The visualization toolkit," http://public.kitware.com/VTK/. Accessed on May 2009.

[52] "Qt: Cross-platform rich client development framework," http://trolltech.com /products/qt/. Accessed on May 2009.

[53] B. Shneiderman, "The eyes have it: A task by data type taxonomy for information visualizations," in *IEEE Visual Languages*, College Park, Maryland 20742, U.S.A., 1996, pp. 336–343.

[54] I. Boada, I. Navazo, and R. Scopigno, "Multiresolution volume visualization with a texture-based octree," *The Visual Computer*, vol. 17, no. 3, pp. 185–197, 2001.

[55] B. Shneiderman, "Tree visualization with tree-maps: 2-d space-filling approach," *ACM Trans. Graph.*, vol. 11, no. 1, pp. 92–99, 1996.

[56] S. Prohaska, A. Hutanu, R. Kahler, and H.-C. Hege, "Interactive exploration of large remote micro-ct scans," in *VIS '04: Proceedings of the Conference on Visualization '04*. 2004, pp. 345–352, IEEE Computer Society.

[57] "Hdf group - hdf5," http://www.hdfgroup.org/HDF5/index.html. Accessed on May 2009.

[58] "Tiff," http://partners.adobe.com/public/developer/en/tiff/TIFF6.pdf. Accessed on May 2009.

[59] "Libtiff - tiff library and utilities," http://www.libtiff.org/. Accessed on May 2009.

[60] "Imagemagick," http://www.imagemagick.org/script/index.php. Accessed on May 2009.

[61] C. D. Hansen and C. R. Johnson, Eds., *The Visualization Handbook*, Boston: Elsevier-Butterworth Heinemann, 2005.

[62] J. C. Fiala, "Reconstruct: a free editor for serial section microscopy," *Journal of Microscopy*, vol. 218, pp. 52–61, 2005.

[63] "Neuron_morpho plugin," http://www.personal.soton.ac.uk/dales/morpho/. Accessed on May 2009.

[64] "Imagej: Image processing and analysis in java," http://rsb.info.nih.gov/ij/. Accessed on May 2009.

[65] K. Al-Kofahi, S. Lasek, D. Szarowski, C. Pace, G. Nagy, J. Turner, and B. Roysam, "Rapid automated three-dimensional tracing of neurons from confocal image stacks," *IEEE Transactions on Information Technology in Biomedicine*, vol. 6, no. 2, pp. 171–186, 2002.

[66] "Scientific visualization tutorial," http://www.cc.gatech.edu/scivis/tutorial /linked/whatisscivis.html. Accessed on May 2009.

[67] "Loni: Laboratory of neuro imaging," http://www.loni.ucla.edu/SVG/index.php. Accessed on May 2009.

[68] Z. Melek, D. M. Mayerich, C. Yuksel, and J. Keyser, "Visualization of fibrous and thread-like data," *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 5, pp. 1165–1172, 2006.

[69] 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.

[70] D. C. Airey, L. Lu, and R. W. Williams, "Genetic control of the mouse cerebellum: Identification of quantitative trait loci modulating size and architecture," *The Journal of Neuroscience*, vol. 21, no. 14, pp. 5099–5109, July 2001.

[71] "Category: Brain," Wikimedia Commons, http://commons.wikimedia.org/wiki/ Category:Brain. Accessed on May 2009.

[72] A. F. Frangi, W. J. Niessen, R. M. Hoogeveen, T. van Walsum, and M. A. Viergever, "Model-based quantitation of 3-d magnetic resonance angiographic images," *IEEE Transactions on Medical Imaging*, vol. 18, no. 10, pp. 946–956, 1999.

[73] J. K. Donghyeop Han and Y. Choe, "A local maximum intensity projection tracing of vasculature in knife-edge scanning microscope volume data," in *IEEE International Symposium on Biomedical Imaging*, 2009.

[74] "Paraview: Scientific visualization," http://www.paraview.org/. Accessed on May 2009.

[75] J. Kwon and Y. Choe, "Enhanced facilitatory neuronal dynamics for delay compensation," *IJCNN 2007. International Joint Conference on Neural Networks, 2007*, pp. 2040–2045, 2007.

[76] J. Kwon and Y. Choe, "Internal state predictability as an evolutionary precursor of self-awareness and agency," in *Proceedings of the Seventh International*

Conference on Development and Learning. 2008, pp. 109–114, IEEE.

[77] A. Thiel, H. Schwegler, and C. W. Eurich, "Complex dynamics is abolished in delayed recurrent systems with distributed feedback times," *Complexity*, vol. 8, no. 4, pp. 102–108, 2003.

[78] R. R. Llinás, *I of the Vortex: From Neurons to Self*, Cambridge: MIT Press, 2002.

[79] J. Hawkins and S. Blakeslee, *On Intelligence*, New York: Times Books, 2004.

[80] G. A. Carpenter and S. Grossberg, "A self-organizing neural network for supervised learning,recognition, and prediction," *IEEE Communications Magazine*, vol. 30, pp. 38–49, 1992.

[81] H. Li and R. Kozma, "A dynamic neural network method for time series prediction using the kiii model," in *Proceedings of the International Joint Conference on Neural Networks*, 2003, pp. 347–352.

[82] H. Lim and Y. Choe, "Compensating for neural transmission delays using extrapolatory neural activation in evolutionary neural networks," *Neural Information Processing–Letters and Reviews*, vol. 10, pp. 147–161, 2006.

[83] H. Lim and Y. Choe, "Delay compensation through facilitating synapses and stdp: A neural basis for orientation flash-lag effect," in *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, 2006, pp. 8385–8392.

[84] H. Lim and Y. Choe, "Facilitating neural dynamics for delay compensation and prediction in evolutionary neural networks," in *Genetic and Evolutionary Computation Conference (GECCO)*, 2006, pp. 167–174, Nominated for Best

Paper Award in the Artificial Life/Evolutionary Robotics/Adaptive Behavior track.

[85] H. Lim and Y. Choe, "Extrapolative delay compensation through facilitating synapses and its relation to the flash-lag effect," *IEEE Transactions on Neural Networks*, vol. 19, no. 10, pp. 1678–1688, 2008.

[86] R. Nijhawan, "Visual prediction: Psychophysics and neurophysiology of compensation for time delays," *Behavioral and Brain Sciences*, vol. 31, pp. 179–239, 2008.

[87] S. J. Thorpe and M. Fabre-Thorpe, "Seeking categories in the brain," *Science*, vol. 291, pp. 260–263, 2001.

[88] H. Lim, "Facilitatory neural dynamics for predictive extrapolation," Ph.D. dissertation, Texas A&M University, 2006.

[89] K. L. Downing, "The predictive basis of situated and embodied artificial intelligence," in *GECCO '05: Proceedings of the 2005 Conference on Genetic and Evolutionary Computation*, New York, 2005, pp. 43–50, ACM Press.

[90] B. Krekelberg and M. Lappe, "A model of the perceived relative positions of moving objects based upon a slow averaging process," *Vision Research*, vol. 40, pp. 201–215, 2000.

[91] J. Liaw and T. W. Berger, "Dynamic synapse: Harnessing the computing power of synaptic dynamics," *Neurocomputing*, vol. 26-27, pp. 199–206, 1999.

[92] F. Gomez and R. Miikkulainen, "2-D pole balancing with recurrent evolutionary networks," in *Proceedings of the International Conference on Artificial Neural Networks*, 1998, pp. 425–430.

[93] F. Gomez and R. Miikkulainen, "Active guidance for a finless rocket through neuroevolution," in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, 2003, pp. 2084–2095.

[94] F. Gomez, "Robust non-linear control through neuroevolution," Ph.D. dissertation, Department of Computer Science, The University of Texas at Austin, Austin, TX, 2003, Technical Report AI03-303.

[95] R. G. Ward and R. Ward, "Representation in dynamical agents," *Neural Networks*, vol. 22, no. 3, pp. 258–266, 2009.

[96] E. S. Fortune and G. J. Rose, "Short-term synaptic plasticity as a temporal filter," *Trends in Neurosciences*, vol. 24, pp. 381–385, 2001.

[97] H. Markram, *Elementary Principles of Nonlinear Synaptic Transmission*, London: Springer-Verlag, 2003.

[98] Y. Tan and A. V. Cauwenberghe, "Neural-network-based d-step-ahead predictors for nonlinear systems with time delay," *Engineering Applications of Artificial Intelligence*, vol. 12, pp. 21–35, 1999.

[99] R. C. Miall and D. M. Wolpert, "Forward models for physiological motor control," *Neural Networks*, vol. 9, pp. 1265–1285, 1996.

[100] X. Yao, *Evolutionary Computation Theory and Applications*, New Jersey: World Scientific, 1999.

[101] R. Ghanea-Hercock, *Applied Evolutionary Algorithms in Java*, New York: Springer, 2003.

[102] M. Buckland, *AI Techniques for Game Programming*, Boston: Premier Press, 2002.

[103] H.-M. Gross, A. Heinze, T. Seiler, and V. Stephan, "Generative character of perception: A neural architecture for sensorimotor anticipation," *Neural Networks*, vol. 12, pp. 1101–1129, 1999.

[104] B. Webb, "Neural mechanisms for prediction: Do insects have forward models?," *Trends in Neurosciences*, vol. 27, pp. 278–282, 2004.

[105] E. Oztopa, D. Wolpert, and M. Kawato, "Mental state inference using visual control parameters," *Cognitive Brain Research*, vol. 22, pp. 129–151, 2005.

[106] E. Oztopa, D. Wolpert, and M. Kawato, "Mental state inference using visual control parameters," *Cognitive Brain Research*, vol. 22, pp. 129–151, 2005.

[107] P. Werbos, "Intelligence in the brain: A theory of how it works and how to build it," *Neural Networks*, vol. 22, no. 3, pp. 200–212, 2009.

[108] H. Markram, Y. Wang, and M. Tsodyks, "Differential signaling via the same axon of neocortical pyramidal neurons," *Proceedings of the National Academy of Sciences, USA*, vol. 95, pp. 5323–5328, 1998.

[109] R. Kozma and W. J. Freeman, "The KIV model of intentional dynamics and decision making," *Neural Networks*, vol. 22, no. 3, pp. 277–285, 2009.

[110] M. S. Gazzaniga, *The Mind's Past*, Berkeley: University of California Press, 1998.

[111] L. Perlovsky, "Language and cognition," *Neural Networks*, vol. 22, no. 3, pp. 247–257, 2009.

[112] D. S. Levine, "Brain pathways for cognitive-emotional decision making in the human animal," *Neural Networks*, vol. 22, no. 3, pp. 286–293, 2009.

[113] B. Scassellati, "Theory of mind for a humanoid robot," *Autonomous Robots*, vol. 12, pp. 13–24, 2002.

[114] T. E. Feinberg and J. P. Keenan, "Where in the brain is the self?," *Consciousness and Cognition*, vol. 14, no. 4, pp. 661–678, 2005.

[115] K. Gold and B. Scassellati, "A Bayesian robot that distinguishes 'self' from 'other.'," in *Proceedings of the 29th Annual Meeting of the Cognitive Science Society*, Nashville, Tennessee, 2007, Posted at http://www.cs.yale.edu/homes/scaz/papers/Gold-CogSci-07.pdf. Accessed on May 2009.

[116] J. Bongard, V. Zykov, and H. Lipson, "Resilient machines through continuous self-modeling," *Science*, vol. 314, pp. 1118–1121, 2006.

[117] J. Weng, J. L. McClelland, A. Pentland, O. Sporns, I. Stockman, M. Sur, and E. Thelen, "Autonomous mental development by robots and animals," *Science*, vol. 291, no. 5504, pp. 599–600, 2001.

[118] J. D. Han, S. Q. Zeng, K. Y. Tham, M. Badgero, and J. Y. Weng, "Dav: A humanoid robot platform for autonomous mental development," in *Proceedings of the 2nd International Conference on Development and Learning*, Cambridge, Massachusetts, 2002, pp. 73–81.

[119] N. Block, "On a confusion about a function of consciousness," in *Behavoral and Brain Sciences*, 1995, pp. 227–247.

[120] C. Menant, "Proposal for an approach to artificial consciousness based on self-consciousness," Tech. Rep., November 2007, http://cogprints.org/3715/ Accessed on May 2009.

[121] J. Taylor, "CODAM: A neural network model of consciousness," *Neural Networks*, vol. 20, no. 9, pp. 983–992, 2007.

[122] S. Nolfi, J. L. Elman, and D. Parisi, "Learning and evolution in neural networks," *Adaptive Behavior*, vol. 3, pp. 5–28, 1994.

[123] D. M. Wolpert, R. C. Miall, and M. Kawato, "Internal models in the cerebellum," *Trends in Cognitive Science*, vol. 2, pp. 338–347, 1998.

[124] K. Kawamura, W. Dodd, P. Ratanaswasd, and R. Gutierrez, "Development of a robot with a sense of self," *Computational Intelligence in Robotics and Automation, 2005. CIRA 2005. Proceedings. 2005 IEEE International Symposium on*, pp. 211–217, 2005.

[125] B. Bakker and M. de Jong, "The epsilon state count," in *From Animals to Animats 6: Proceedings of The Sixth International Conference on Simulation of Adaptive Behavior*, 2000, pp. 51–60.

[126] E. T. Rolls, "A computational neuroscience approach to consciousness," *Neural Networks*, vol. 20, no. 9, pp. 962–982, 2007.

[127] V. Itskov and C. Curto, "From spikes to space: reconstructing features of the environment from spikes alone," *BMC Neuroscience*, vol. 8, no. Suppl 2, pp. P158, 2007.

[128] C. W. Anderson, "Learning to control an inverted pendulum using neural networks," *IEEE Control Systems Magazine*, vol. 9, pp. 31–37, 1989.

[129] B. Mehta and S. Schaal, "Forward models in visuomotor control," *Journal of Neurophysiology*, vol. 88, pp. 942–953, 2002.

[130] H. Lim and Y. Choe, "Facilitating neural dynamics for delay compensation and prediction in evolutionary neural networks," in *GECCO '06: Proceedings of the 2006 Conference on Genetic and Evolutionary Computation*, 2006, pp. 167–174.

[131] R. J. Frank, N. Davey, and S. P. Hunt, "Time series prediction and neural networks," *Journal of Intelligent and Robotic Systems*, vol. 31, no. 1-3, pp. 91–103, 2001.

VITA

Jae-rock Kwon was born in Yeongyang, Kyeongsangbukdo, Korea, the son of Gijeong Kwon and Jongsik Yun. He received his B.S. and M.S. degrees in electronic communication engineering from Hanyang University in 1992 and 1994 respectively. After graduation, he worked at LG Electronics, SK Teletech, and Qualcomm in Seoul, Korea. He entered into the Computer Engineering program in Texas A&M University for his doctoral studies in the fall of 2004 and graduated with his Ph.D. in August 2009.

Permanent Address:

Department of Computer Science and Engineering

Texas A&M University

3112 TAMU

College Station, TX 77843-3112