KNIFE EDGE SCANNING MICROSCOPE BRAIN ATLAS INTERFACE FOR TRACING AND ANALYSIS OF VASCULATURE DATA

A Thesis

by

MANISHA SRIVASTAVA

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

MASTER OF SCIENCE

August 2015

Major Subject: Computer Engineering

ABSTRACT

The study of the neurovascular network in the brain is important to understand brain functions as well as causes of several brain dysfunctions. Many techniques have been applied to acquire neurovascular data. The Knife-Edge Scanning Microscope (KESM), developed by the Brain Network Lab at Texas A&M University, can generate whole-brain-scale data at submicrometer resolution. The specimen can be stained with different stains, and depending on the type of stain used, the KESM can image different types of microstructures in the brain. The India ink stain allows the neurovascular network in the brain to be imaged.

In order to visualize and analyze such large datasets (~ 1.5 TB per brain), a light-weight, web-based mouse brain atlas called the Knife-Edge Scanning Microscope Brain Atlas (KESMBA) was developed in the lab. The atlas serves several whole mouse brain data sets including India ink. The multi-section overlay technique used in the atlas enables 3D visualization of the structural information in the data. To solve the challenging issue of tracing micro-vessels in the brain, in this thesis a semi-automated tracing and analysis method is developed and integrated into the KESM brain atlas.

Using the KESMBA interface developed in this thesis, the user can look at the 3D structure of the vessels on the brain atlas and can guide the tracing algorithm. To analyze the vasculature network traced by the user, a data analysis component is also added. This new KESMBA interface is expected to help in quickly tracing and analyzing the vascular network of the brain with minimal manual effort.

DEDICATION

To my family

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my advisor, Dr. Yoonsuck Choe for his guidance in my research, continuous support, patience and encouragement. I would also like to thank my committee members, Dr. Ricardo Gutierrez-Osuna and Dr. Ursula Winzer-Serhan for their valuable comments and advice.

Also, I would like to thank Chul Sung, Jaewook Yoo, Ankur Singhal and Anshul Gupta for their help in this thesis. Without their help and comments, this research would not have been possible. I would also like to thank my friends, lab mates and the department faculty and staff for making my time at Texas A&M University a great experience.

Part of this research was funded by the National Science Foundation, under grants #0905041 and #1208174 (Collaborative Research in Computational Neuroscience, PI: Yoonsuck Choe). Data used in this project was enabled by NSF grants #0079874 and #1256086 and NIH/NINDS grant #1R01-NS54252.

Finally, I would like to express my gratitude to my family for their love and support.

TABLE OF CONTENTS

LIST OF FIGURES

Page

LIST OF TABLES

Page

1. INTRODUCTION

The Knife-edge scanning microscope (KESM) - developed by Brain Network Lab at Texas A&M University, can obtain high-resolution images from the entire mouse brain. It can scan an entire mouse brain (typically of volume ~1 cm^3) in less than 100 hours. The mouse brains stained with Golgi (neuronal morphology), India ink (vascular network), and Nissl (soma distribution) [1] have been successfully imaged using the KESM.

In order to visualize and analyze such large data sets (~ 1.5 TB per brain), a light-weight, web-based mouse brain atlas called the Knife-Edge Scanning Microscope Brain Atlas (KESMBA: http://kesmba.cs.tamu.edu) has been developed in the lab. The atlas serves several whole mouse brain data sets. The transparent overlay technique used in the atlas enables 3D visualization of the structural information of the data.

To understand the brain structure and the associated dysfunctions such as strokes, hemorrhage, and thrombosis it is critical to develop algorithms to trace the microvasculature network (India ink dataset). It is really challenging to come up with an algorithm that is accurate and fast enough to work on the entire brain, hence this thesis aims to tackle this challenge by developing a semi-automated tracing and analysis framework as part of the KESMBA.

1.1 Motivation

Understanding the microvasculature network in the brain is important to understand different functions of the brain and the associated brain dysfunction such as strokes, hemorrhage, and thrombosis. Many techniques have been developed to obtain high resolution images of the microvasculature structure in the brain. Whole-brain scale vascular data from mouse brain stained with India ink has been obtained using the KESM.

Presently, there are several automated methods to trace the vessels in the brain. However most of them focus either on tracing vessels at the macro level in the whole brain or tracing micro vessels in a small section of the brain. Moreover, in case of KESM dataset, there is no ground truth available to evaluate the performance of these automated tracing methods. On the other hand manual methods are limited by the time consuming nature of the process.

1.2 Goal of research

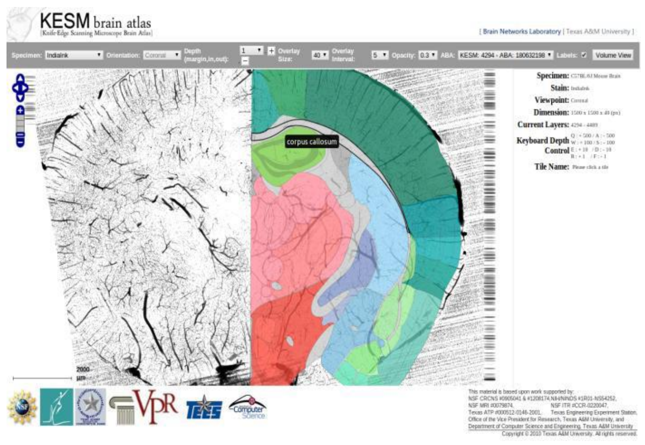
The objective of this thesis is to develop an interface for the KESMBA that will allow the users to quickly trace large volumes of vasculature data in different parts of the mouse brain. The developed interface will also aid the user in analyzing the traced vasculature network. For a user specified region of the mouse brain, it will calculate key quantitative data such as volume density of the vasculature network, number of branch points and length of different segments. The developed interface will be used by neuro scientists to trace the blood vessels in the brain and gather information about the vasculature network in different regions of the brain. Later this interface can be utilized as a crowd sourcing tool to allow quick tracing of the entire vascular network in the brain. Such information on micro vascular network in the brain is required to understand various processes in the brain and the associated dysfunctions.

1.3 Approach

The developed system is a part of the KESMBA and has a tracing component to allow the user to quickly trace large volumes of blood vessels, and an analysis component to extract and display quantitative data from a user specified region of the brain.

The tracing begins when the user clicks on a point inside a blood vessel on the brain atlas. Next a region growing algorithm runs using the clicked point as the seed point, and colors all points inside that vessel with the same color. These changes are immediately visible to the user. Once the user is satisfied and presses a key, the algorithm calculates the centroid of the colored region and propagates it to the next layer (Z slice). It then starts the region growing process to color the vessel in the next layer. Again the colored vessel in the next layer is visible to the user. This process continues till the user clicks on a new point to start a new trace.

Another extension to the KESMBA interface is a volume analysis facility, where the user can specify a region in the brain and color of the vessel, and the algorithm calculates morphometric data for the region. This means that if a user is interested in a particular vessel (that is known to be important) then he can choose the color appropriately to get quantitative data for the vessel. He can also choose to analyze all the vessels in a given region of the brain. It is important to note that for some of the Z slices, the different regions of the brain are labeled since the rich annotations from the Allen Brain Atlas [2] have been manually registered to the KESM dataset. Figure 1 shows an example of the India ink Z slice overlaid with the Allen brain atlas annotations.


KESM brain atlas
[Knife-Edge Scanning Microscope Brain Atlas]
[Brain Networks Laboratory | Texas A&M University]
Volume View
corpus callosum
VPR
TEES
Computer Science

Specimen: Indiaink
Orientation: Coronal
Depth (margin,in,out): 1
Overlay Size: 40
Overlay Interval: 5
Opacity: 0.3
ABA: KESM: 4294 - ABA: 180632198
Labels: ✓

Specimen: C57BL/6J Mouse Brain
Stain: Indiaink
Viewpoint: Coronal
Dimension: 1500 x 1500 x 40 (px)
Current Layers: 4294 - 4489
Keyboard Depth Control Q : + 500 / A : - 500
W : + 100 / S : - 100
E : + 10 / D : - 10
R : + 1 / F : - 1
Tile Name: Please click a tile

This material is based upon work supported by:
NSF CRCNS #0905041 & #1208174, NIH/NINDS #1R01-NS54252,
NSF MRI #0079874, NSF ITR #CCR-0220047,
Texas ATP #000512-0146-2001, Texas Engineering Experiment Station,
Office of the Vice President for Research, Texas A&M University, and
Department of Computer Science and Engineering, Texas A&M University
Copyright © 2010 Texas A&M University. All rights reserved.

Figure 1. One z-slice of India Ink data and the registered Allen brain atlas annotation

Once the user specifies a region in the brain, all the tiles in that region are collated to create a single volume file. Then a centerline extraction algorithm runs in the backend to extract centerlines from the vasculature network in the volume. Based on the original volume and the extracted centerline of the vasculature network, several quantitative metrics are calculated and displayed on the KESMBA.

1.4 Significance

This thesis extends the KESM brain atlas by providing an interface for quickly and efficiently tracing the microvascular network in the brain. It also creates an analysis framework for analyzing the traced vascular networks. The framework created as part of this thesis is semi-automated with limited need for human intervention and assistance. This method can trace the vascular network in a small time and provide statistics.

Since the tracing results are visible to the user at every step, the user can guide the tracing algorithm at those points where it fails. The traced vessels are stored on the

KESMBA server, and they can be used as ground truth for other supervised algorithms or in validation of other automated tracing algorithms. This thesis is expected to contribute to the ongoing research to study the vascular networks of the brain in a significant way.

1.5 Outline

This thesis is organized as follows. In chapter 2, background on the KESM system, the KESMBA web atlas and the KESM data set will be described. In addition, related work about tracing algorithms will be discussed. In chapter 3, the methodology of the thesis will be explained, including tracing and data analysis component. In chapter 4, results of the tracing and analysis component will be presented. In chapter 5, the discussion about the results, open issues and future works will be presented. In chapter 6 conclusions will be discussed.

2. BACKGROUND AND RELATED WORK

In recent years many 3D microscopy techniques have been developed to obtain high resolution images from the entire brain. One of the first such instruments to achieve whole brain-scale imaging at sub-micrometer resolution is the Knife-Edge Scanning Microscope (KESM). The entire mouse brain has been successfully imaged, using the KESM, stained with Golgi (neuronal morphology), India ink (vascular network), and Nissl (soma distribution) [1]. At the beginning of this chapter an overview of KESM is presented. The Knife Edge Scanning Microscope Brain Atlas (KESMBA) was developed to allow visualization and analysis of the KESM data. This web based atlas serving several whole mouse brain dataset is presented next. This is followed by an overview of the KESM datasets. KESM India ink dataset contains the vasculature network in the mouse brain; tracing and analysis of the vessels in the data is crucial for understanding the function of the blood vessels in different brain regions. Several methods have been attempted to trace these vessels which are discussed in the related work section.

2.1 Background

In this section the KESM (used to image the whole mouse brain), the KESMBA (for exploration and visualization of the whole mouse brain data), and the different datasets obtained using KESM are discussed.

2.1.1 Knife-edge scanning microscope

The Knife Edge Scanning Microscope [3] can image whole mouse brains at submicrometer resolution. The KESM thus fills the technology gap between imaging techniques which produce ultrahigh resolution images of a small volume of brain and techniques which produce low resolution images of whole brains. The specimen can be stained with different stains and depending on the type of stain used; the KESM can image different types of microstructures in the brain. The instrument is illustrated in Figure 2.

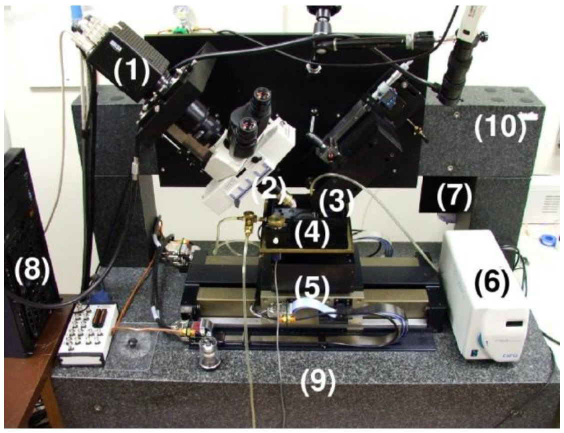


Figure 2. A photo of the Knife-Edge Scanning Microscope (KESM). (1) high speed line-scan camera, (2) microscope objective, (3) diamond knife and light collimator,(4) water immersion specimen tank, (5) air-bearing three-axes precision stage, (6)white-light illuminator, (7) water pump (in the back), (8) computer server installed with image acquisition and stage controller boards, (9) granite base, and (10) granite bridge. Reprinted with permission from [3], Copyright © 2008, John Wiley and Sons.

As shown in the figure, the air-bearing stage is controlled to provide ultra-precise mechanical movement in three axes with 20 nm precision in x and y axes, and 25 nm in z axis. One of the key features in the KESM is the custom knife-collimator assembly that serves both as a white-light collimator and a diamond microtome. The diamond knife is rigidly mounted to a massive granite bridge over the three-axis stage. On the other side, a microscope objective is held over the stage. Through the microscope, a high-speed line-scan camera can image the brain tissues at sub-micrometer resolution. A close-up of KESM is shown in Figure 3.

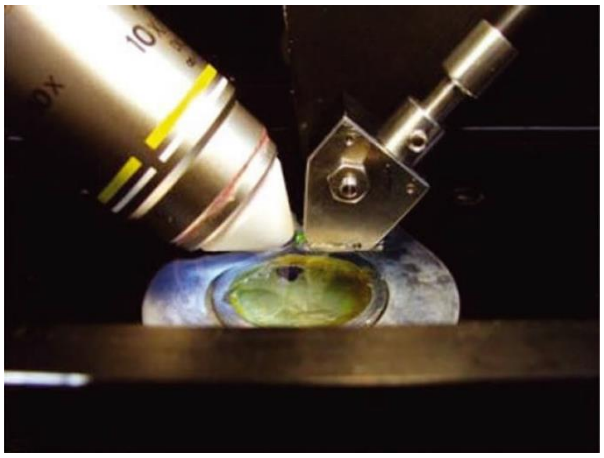


Figure 3. Close up of KESM. Reprinted with permission from [3], Copyright © 2008, John Wiley and Sons.

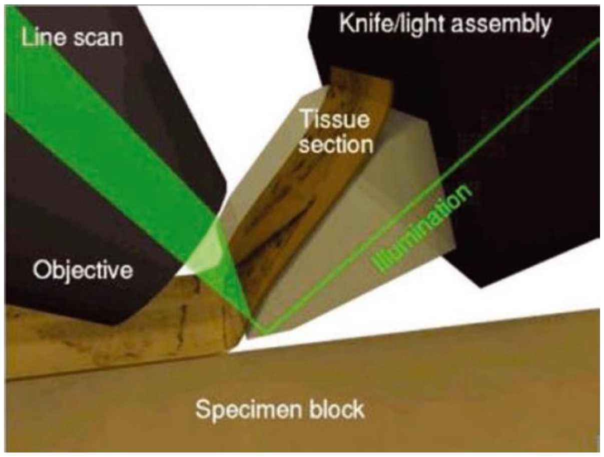


Figure 4. Imaging principle of KESM. Reprinted with permission from [3], Copyright © 2008, John Wiley and Sons.

The operational principle of the KESM is to simultaneously section and image tissue blocks. Details are shown in Figure 4. On top of the three-axis precision stage a specimen tank is kept, which contains the brain tissue embedded in hard polymer resin block immersed under water. The microscope objective and the diamond knife are aligned. The diamond knife is used to sequentially cut sections from the tissue blocks, while the collimator provides illumination. The light passes through the diamond knife and penetrates the tissue sections for imaging. The camera scans the tissue piece through the microscope objective and transfers the digital image to the computer server. In order to automatically control the stage movement and the data acquisition process, in-house control software is used [4].

2.1.2 Knife-edge scanning microscope brain atlas

Due to the multiscale nature of the KESM data, visualization and exploration are very challenging. To solve this issue, a web-based atlas was developed for efficient visualization and analysis of the dataset. The design of the framework is such that it is web based, light weight and is not dependent on custom 3D viewing applications or plug-ins.

The KESM brain atlas (KESMBA) [5] provides 3D visualization by customizing OpenLayers [6], an open source Javascript library. Similar to Google Maps, the OpenLayers API supports geographic navigation such as panning, zooming, and overlaying. The basic idea used is a transparent overlay of images with distance attenuation as shown in Figure 5. Transparency (alpha channel) was added to the stored images, so that when the client requests multiple layers to be shown simply overlaying them gives a 3D effect. This transparent overlaying technique combined with zoomable web interface provided by the OpenLayers, resulted in a powerful environment for visualization and exploration of the large 3D KESM data.

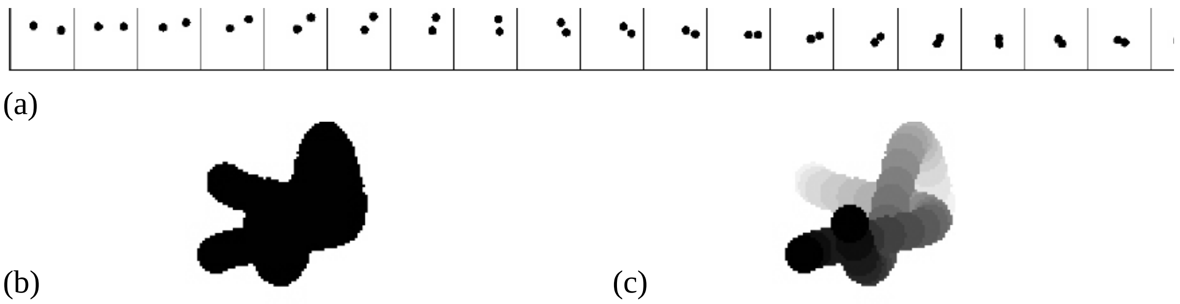


Figure 5. Transparent Overlay with Distance Attenuation (a) An image stack containing two intertwined objects is shown; (b) Simple overlay of the image stack in (a) results in loss of 3D perspective; (c) Overlay with distance attenuation helps bring out the 3D cue [5].

After preprocessing the images and adding transparency, pyramidal tiles of size 256x256 pixel were generated. The bitmap-based tiles were converted to Scalable Vector Graphics (SVG) which is a XML-based vector graphics format to reduce the file size. This was done to reduce the tile loading time on the KESMBA browser. Each tile is named to be consistent with Google Maps tile specification. For example, a tile name 1_2_3.svg denotes zoom level=1, x-coordinate=2, and y-coordinate=3. The pyramidal structure of the Openlayer tiles in different zoom levels is shown in Figure 6. The tile pyramid reduces image download time; hence generating them is useful inspite of the fact that it requires extra storage, time and effort.

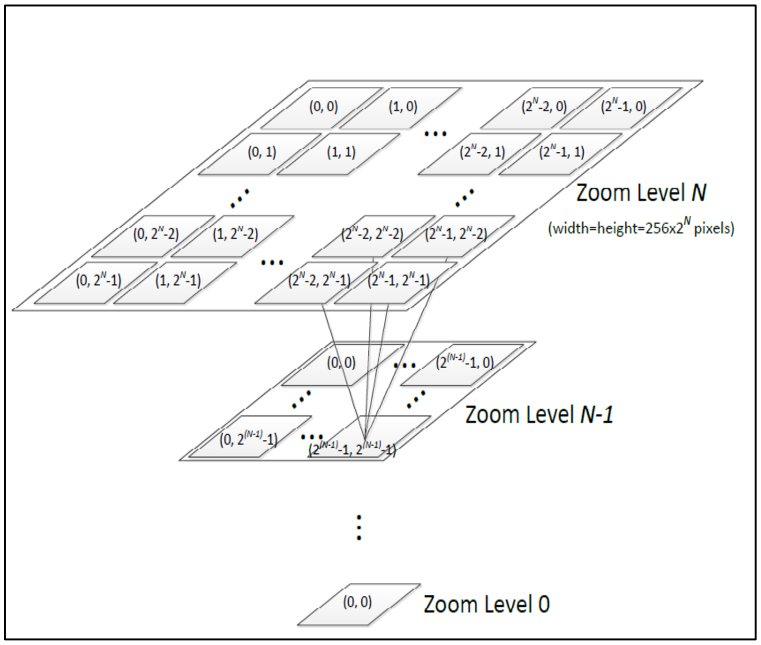


Figure 6. Tile Pyramid [5].

2.1.3 KESM dataset

Using the KESM, entire mouse brains stained with Golgi (neuronal morphology), India ink (vascular network), and Nissl (soma distribution) have been successfully imaged. In 2008, Golgi and India ink data sets were imaged from whole mouse brains. In 2010, Nissl and the second Golgi data sets were obtained using the KESM. The first Golgi dataset was partly incomplete whereas the second Golgi dataset spans the whole mouse brain. Figure 7 shows the screenshots of these data using KESMBA.

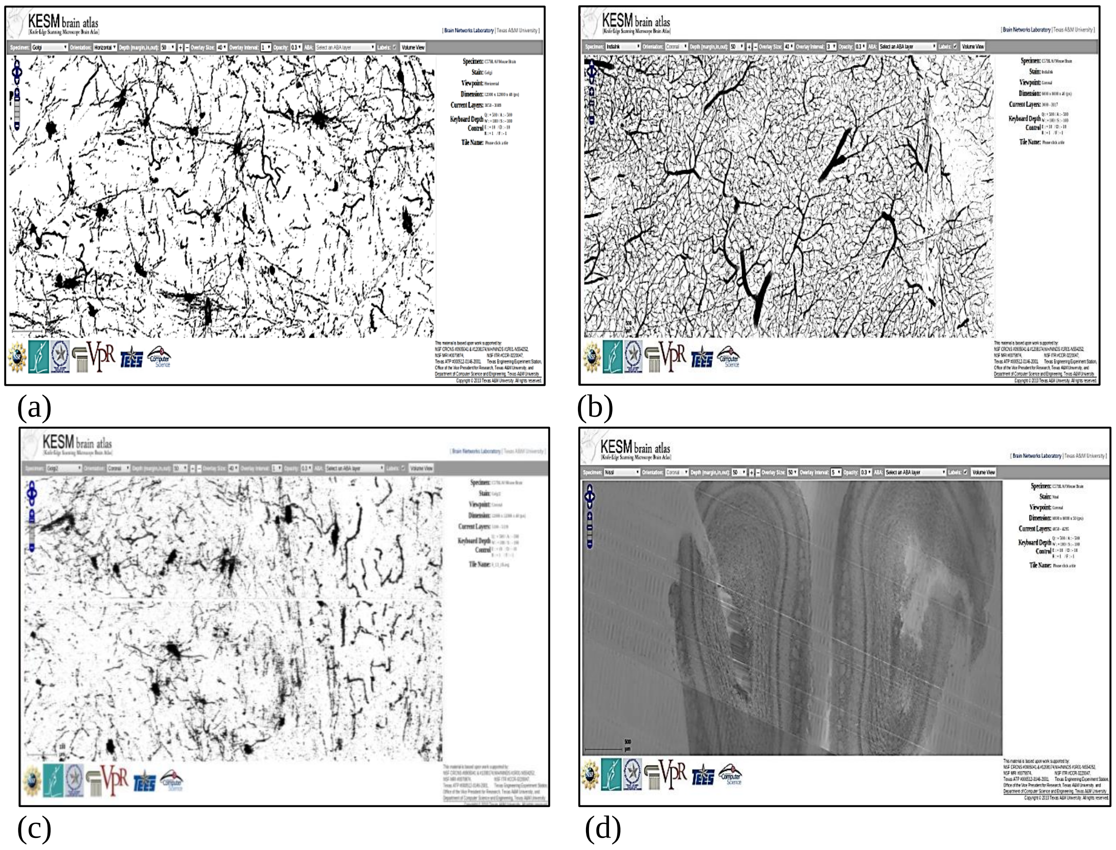


Figure 7. KESM datasets on the brain atlas (a) Golgi 1(neurons); (b) IndiaInk(vasculature); (c) Golgi 2; (d) Nissl (Soma).

The KESMBA currently serves Golgi1, Golgi2, India ink, Nissl and Array tomography data. Other datasets can be easily added after the data are processed to create multi scale tile pyramid. In this thesis, the main focus has been on the India ink data, which show the vascular network of the mouse brain. Figure 8 shows the vasculature data set, the data shown were processed to show a three-dimensional view. The developed KESMBA interface can potentially be applied to other datasets as well.

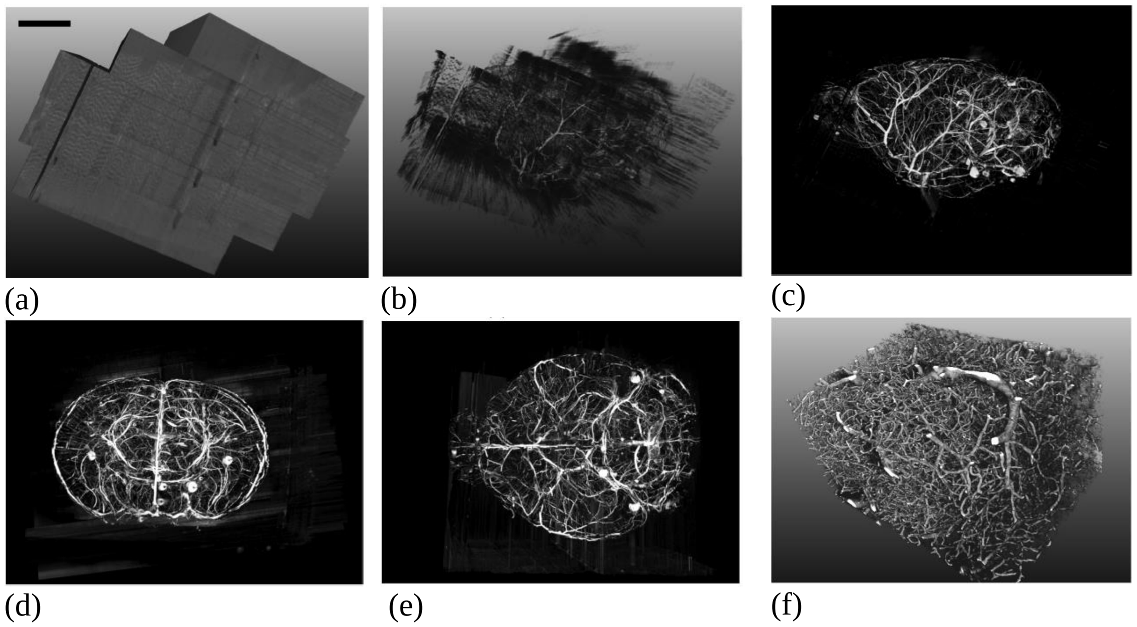


Figure 8. Different visualization of KESM vasculature data. (a) A raw data block in a sagittal view. (b) is generated from (a) where the data block was lightly thresholded. The boundary of the raw data and the content starts to appear. (c)-(e) are fully thresholded version of (a) but in different views(sagittal, coronal, and horizontal). The whole brain shape can be seen. (f) Intricate details within a 1.5 mm-wideblock. Reprinted with permission from [7], Copyright © 2011, IEEE.

2.2 Related work

After obtaining the image data from whole mouse brain using the KESM, the next step is to analyze the data. In case of India ink data, it would be really useful to trace the micro vascular network in the brain to understand more about the structure of these networks. Different techniques have been developed to trace the vasculature data. In this section, related works on vessel tracing are discussed.

Han et al. [8] developed a Maximum Intensity Projection (MIP) based 3D vector tracing algorithm for tracing the microvasculature network in the mouse brain. In their technique, first seed points were selected manually by the user to initiate the tracing. These seed points were corrected geometrically by centering the seed point along both horizontal and vertical edges of the vessel. For each seed point, the three local axis lengths were determined by boundary detection to create a local volume. This was followed by local volume evaluation and local MIP processing. Next the direction of the blood vessel was determined using 2D tracing methods, using which the 3D vessel direction was predicted. The 2D MIP processing, which is performed twice in each iteration of the algorithm, is computationally very intensive.

Mayerich et al. [9] presented a predictor-corrector algorithm to trace filament structure in volume data. They tried to extract the centerline of the vessels by predicting the path of a particle called tracer. The next tracer position was searched only within a solid angle of the previous vector. The vector whose cross-sectional image most closely matched the template was chosen as the next tracing direction. This method involves template matching process which is time consuming when tracing large vascular volume.

Yang [10] developed a flat box based tracing method for vasculature data from mouse brain using the idea of vector tracing. Seed point selection was performed on each outer surface of the vascular volume by selecting the centroids of vessel cross-section on that surface. Next vector-based vascular tracing method was applied by using a virtual flat box to observe the vessels and determine the subsequent points of the centerline. The vessel was traced by estimating the centroids of vessel cross-section while traversing the path of the flat box. Since the vascular tracing method starts from the seed points which are selected only on the surfaces of the volume, it is possible that the method misses vessels in the center of the volume if the volume is large.

Though most of these methods are fully automated, it is challenging to come up with an algorithm that can work in all parts of the brain due to the large volume of data, different types of local noise present in the data, and the complexity of the vasculature network in the brain. Hence user guided tracing algorithms looks more promising in near future.

EyeWire [11] developed by Sebastian Seung's Lab at Princeton (previously at MIT) is a game to map the connections between neurons in the retina. Eyewire was officially launched on December 10, 2012 and has since grown to over 180,000 players from 150 countries [12]. The data used in the game was generated by the Max Planck Institute for Medical Research.

The game utilizes the guidance from the user to map 3D neurons in the retina. The player is given a volume with a partially reconstructed neuron branch, and he has to complete the tracing of that branch. In order to trace (color) a neuron cross section, the player clicks on areas in the slices to add them to the trace. An AI algorithm is running at the back, automatically fills in the parts of the neuron that it detects are part of the player's clicked area. This generates volumetric reconstructions, branch by branch. Figure 9 shows a screenshot of the Eyewire game screen, on the right side of the screen there is a black and white image of the cross sections of neurons where the player has to click. The cube on the left side shows the 3D reconstruction of the traced neuron. Each volume is presented to three to five different players, and the trace chosen by the majority of the players is accepted.

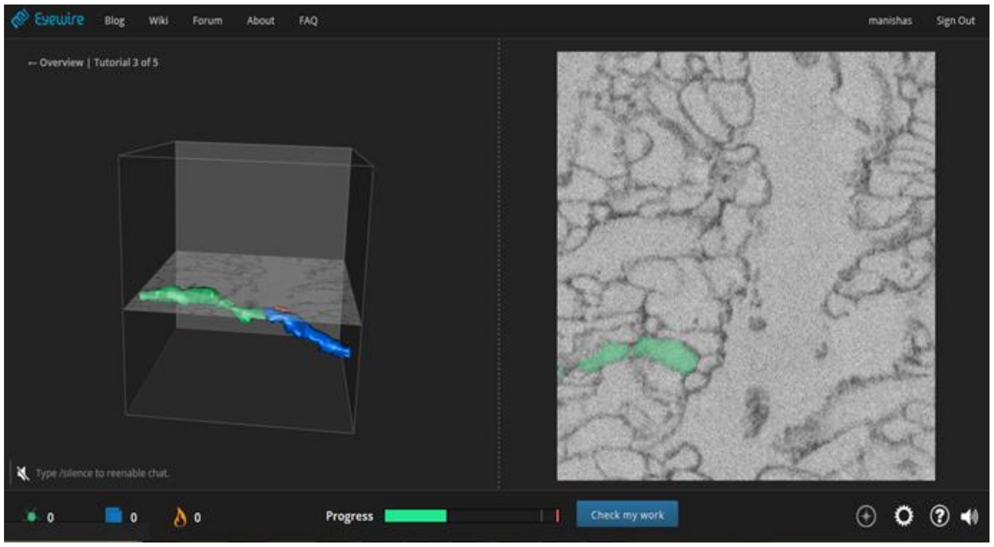


Figure 9. Screenshot of Eyewire game screen. On the right, cross-section of the neurons where the player needs to click for tracing. On the left, the 3D reconstruction of the traced neuron.

3. METHODS

This chapter describes the methodology of this thesis work in detail. In order to facilitate efficient tracing and analysis of the microvasculature network present in the India ink dataset, an interface is developed as part of the KESM brain atlas. The overall system consists of two main components: a tracing and an analysis component. These components are discussed in detail in the following sections.

3.1 Tracing component

The tracing component provides a user interface to allow a user to trace the blood vessels efficiently. Figure 10 shows the block diagram of the tracing component.

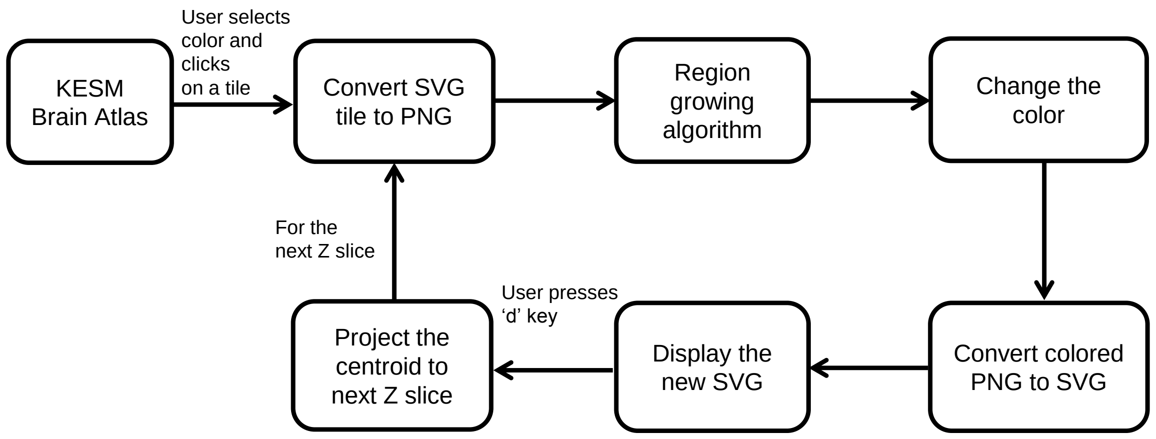


Figure 10. Block diagram for the tracing component

The idea is to use the KESM brain atlas as a platform to allow the user to efficiently trace the vasculature data. To begin tracing, the user chooses a tracing color and clicks on any point inside a vessel on the brain atlas. The KESMBA display is made up of tiled SVG images. The tile on which the user clicks is converted to PNG from SVG format. A region growing algorithm runs in the back to examine the neighboring pixels of initial clicked point (seed point) to determine whether they belong to the same vessel. The process is then repeated for the new pixels that are added to the region until the region stops growing. The color of the region that is just grown is then changed and the colored tile is converted back to SVG format and displayed on the brain atlas. The user then presses the ‘d’ key and the centroid of the region is propagated to next layer (Z slice) to repeat the process for the tiles in the next layer.

3.1.1 Region growing

The first step in the region growing algorithm is to select a seed point. The selection of seed point is critical as the region growing process begins at the chosen seed point. In this case the seed point selection is not an issue, as the user can visualize the 3D structure of the data on KESMBA and click on the appropriate seed point.

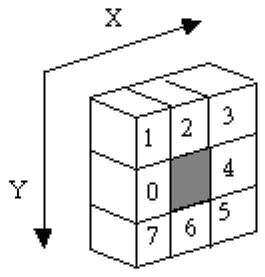


Figure 11. 8-connected neighborhood [13].

The region is then grown from the selected seed point to adjacent points depending on a region membership criterion. As in this case the region is grown only in 2D, 8-connected neighborhood is used. Figure 11 shows the 8 connected neighborhood where for a voxel (x, y, z) the 8 connected neighborhood is: (x-1,y-1,z) (x+1,y-1,z) ,(x-1,y,z) (x+1,y,z) , (x-1,y+1,z) (x+1,y+1,z).

The region membership criterion for growing the region can be based on pixel intensity values, texture, color etc. PNG image format uses RGBA color space, and for this work the alpha channel of the RGBA pixel intensity values is used to determine the region membership. The alpha channel is normally used as an opacity channel. If a pixel has a value of 0 in its alpha channel it is fully transparent , whereas a value of 1 in the alpha channel gives a fully opaque pixel. Hence in this case, the alpha channel value of a voxel being greater than a threshold means that the voxel belongs to the blood vessels as the background information is transparent and has very low alpha value. The algorithm examines the 8 neighbors of the seed points and if they have the alpha channel value greater than a threshold, they become the seeds points for later iterations. The complete region growing algorithm is shown below:

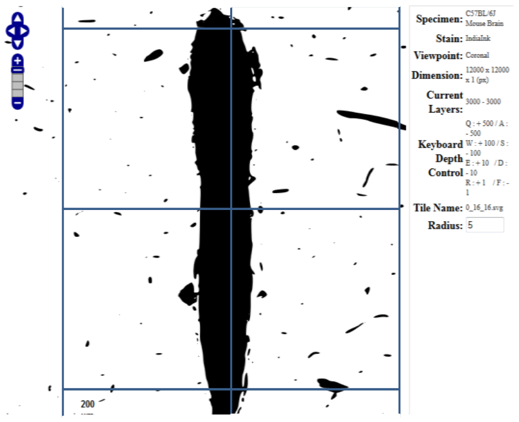
1. User clicks on a seed voxel.
2. The region is grown from the seed voxel by adding in neighboring voxels that have high alpha channel values.

   Every neighbor is tested for high alpha value; if the alpha value of the neighbor is greater than a threshold value then the neighbor is added to the current region and list of seed points. The color of the voxels in the grown region is changed to the user specified color.

3. The current seed is removed from the list of seeds.
4. Repeat 2-3 until the growth of the region stops which means that the list of seed point is empty.

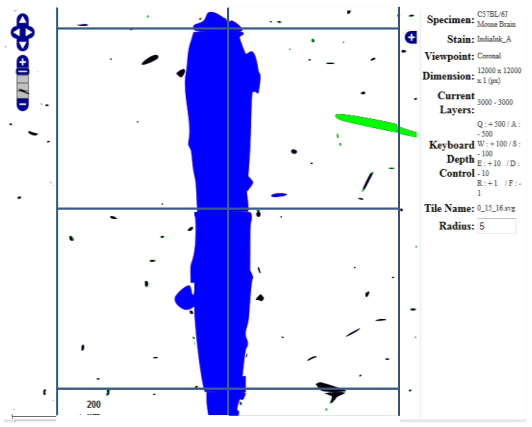
3.1.2 Extended region growing across tiles

The KESM Brain Atlas is made up of multi scale pyramidal of tiles, each of which is 256×256 pixel in size. In a single layer a blood vessel can cross the tile boundaries and extend across multiple tiles. Hence the region growing algorithm is adapted to take this into account. If the current seed voxel happens to be on the boundary of the tile, then to find its 8 connected neighbors, the voxels from the neighboring tiles are read. The way this is implemented is that for the seed point that lies on the boundary of a tile, the name of the tile adjacent to that boundary and the seed point (x, y) location on this neighboring tile is saved into a text file. Once the processing of the current tile is done, the new tile listed in the text file is read along with the seed point. A new region growing algorithm runs for this new tile.



Specimen: C57BL/6J Mouse Brain
Stain: IndiaInk
Viewpoint: Coronal
Dimension: 12000 x 12000 x 1 (px)
Current Layers: 3000 - 3000
Keyboard Depth Control: Q : + 500 / A : - 500 W : + 100 / S : - 100 E : + 10 / D : - 10 R : + 1 / F : - 1
Tile Name: 0_16_16.svg
Radius: 5

(a)



Specimen: C57BL/6J Mouse Brain
Stain: IndiaInk_A
Viewpoint: Coronal
Dimension: 12000 x 12000 x 1 (px)
Current Layers: 3000 - 3000
Keyboard Depth Control: Q : + 500 / A : - 500 W : + 100 / S : - 100 E : + 10 / D : - 10 R : + 1 / F : - 1
Tile Name: 0_15_16.svg
Radius: 5

(b)

Figure 12. Region growing across tiles (a) Vessel that extends across tiles (a blue colored box represent a single tile); (b) Region growing across tiles to color the entire vessel in blue.

Figure 12a shows an example of a blood vessel that extends across multiple tiles. Figure 12b shows the result of region growing algorithm for the blood vessel.

3.1.3 PNG-SVG conversion

All the tiles displayed on the KESM brain atlas is in XML-based vector graphics format, Scalable Vector Graphics (SVG). Since the conversion to SVG vectorizes the image, to access the individual pixel values the tile is converted to PNG (raster) format. This conversion of the vector to raster format is done using ImageMagick. ImageMagick [14] is a free and open-source software suite for displaying, converting, and editing raster image and vector image files. The tiles are first converted to PNG format using ImageMagick and the output raster image is used for region growing. Once the region is grown and the color of the region is changed to the user specified color, the raster scan format needs to be converted back to SVG to be displayed on the brain atlas.

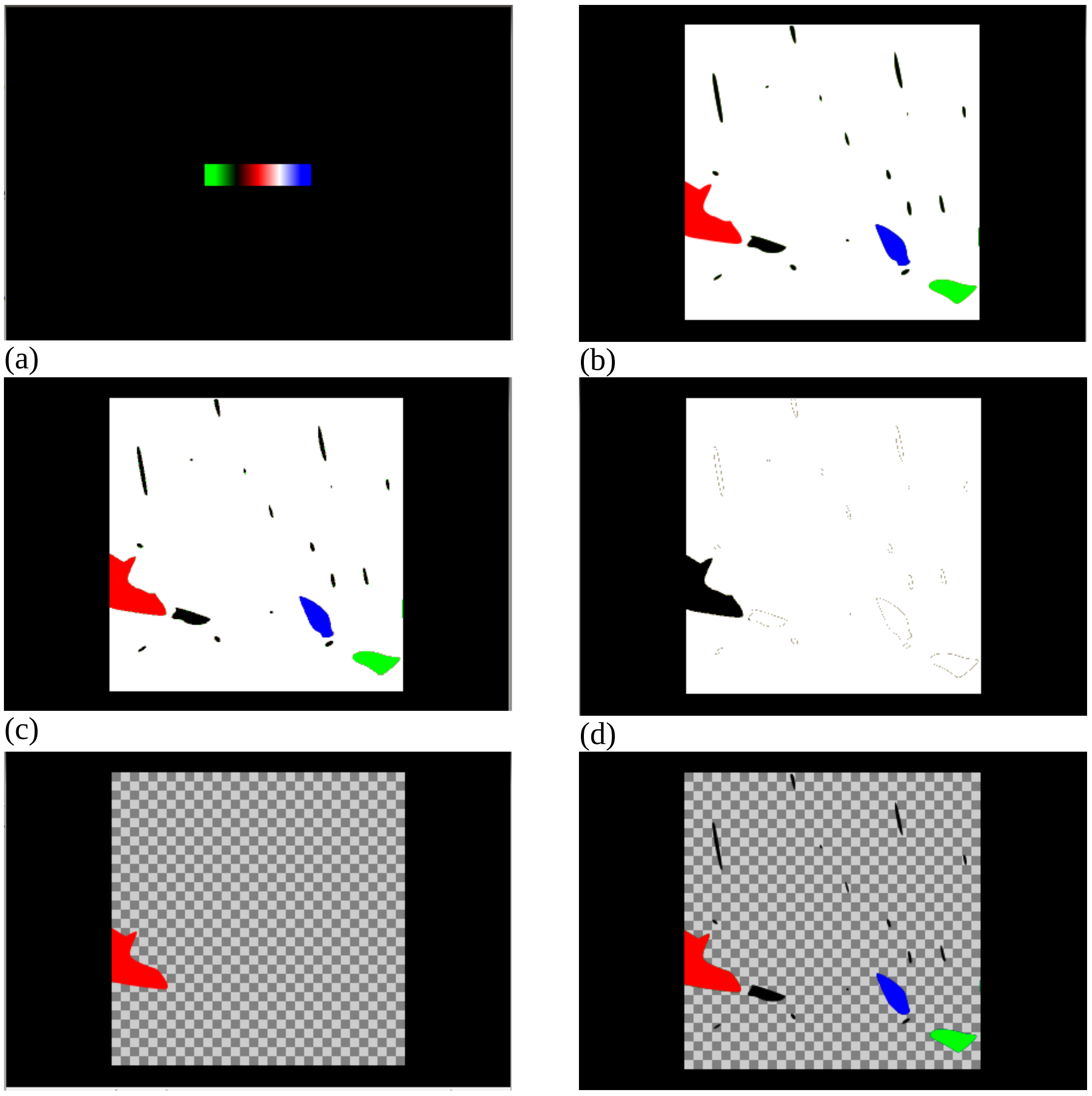

(a)
(b)
(c)
(d)
(e)
(f)

Figure 13. Conversion of colored PNG tile to SVG (a) Color palette; (b)Colored PNG tile; (c) Color Quantized tile; (d) Tile where red color has been isolated; (e) SVG tile containing only red colored vessel; (f) Converted SVG tile

Although ImageMagick can convert from vector format to raster, it cannot convert the raster back to the vector format. Potrace [15] can be used to generate SVG vector image data from the raster scan. Potrace is an open source tool which transforms bitmaps into vector graphics. But Potrace can only vectorize black and white images as it just finds edges and makes paths.

It is really challenging to effectively vectorize a colored raster image. One solution to this is to use ImageMagick to do color-separation first, and then edit the SVG output from Potrace as appropriate. With this approach images with a few constant color regions can be effectively vectorized to SVG format.

Figure 13 shows the steps to convert a colored PNG image to SVG format. The first step in the process is to specify a color palette (Figure 13a). The palette should be provided in the form of a PNG image where each palette entry has exactly one pixel of that color. Then the PNG image in RGBA format is quantized with ImageMagick 'pngnq' function. The algorithm uses a neural network to optimize the color map selection. An example of quant image is shown in Figure 13c; the original PNG image is shown in Figure 13b.

The next step is to run an 'isolate color' algorithm for each color specified in the palette. This algorithm takes as input a source color and an image, and returns an image where the objects in the source color are filled with black and all the rest with white. This means that in the output image only the objects colored with the source color are shown in black; all other objects are merged with the background and are shown in white color. 'Fill' ImageMagick command is used to do this. An example is shown in Figure 13d where the source color is red.

The result of the previous steps is a binary image with some objects in black and rest everything in white, Potrace can be used to convert this binary image to SVG. An example trace image is shown in Figure 13e. Next these separate SVG traces (one for each color) are added to create a composite SVG with appropriate color (Figure 13f).

3.1.4 Displaying the new tile

As explained in the previous sections, after region growing the new colored tiles are converted to SVG format and stored on the KESMBA server. But the tiles were not being displayed on the KESMBA web browser even after refreshing the page. This was happening because of caching used by the browser.

A web cache is an information technology where the web documents, such as HTML pages and images are stored temporarily. This is usually done to reduce bandwidth usage, server load and perceived lag. Caches are used to speed up a process so that data does not have to be recomputed or fetched from its original location and, therefore save time. Internet browsers use caching to store HTML web pages by storing a copy of visited pages and then using that copy to render when the page is re-visited/refreshed. Hence in this thesis, the browser was caching the tiles and the new colored tiles were not being displayed.

The solution is to introduce some random number in the URL of the tiles, used by the browser to fetch tiles from the KESMBA server. In the KESMBA server each layer (Z slice) has a folder which contains all the tiles for that layer. To solve the caching problem, the layer folder names are renamed as layer#_random#. Hence before the URL of a layer is created, the layer folder is renamed with a new random number and hence new URL is generated every time. Since the browser has not seen this URL before (because of the random number component) it does not use the cached tile, rather it gets the new processed tile from the server and displays it.

3.1.5 Centroid propagation to next layer

Once the region is grown on the tiles of the current layer, the user presses‘d’ key and the region growing process is repeated for the next layer. But this time the seed point is selected automatically without the user having to click on a new point. To select the seed point for the next layer, a minimum rectangular window ((Xmin; Ymin) and (Xmax; Ymax)) bounding the region is used. For the circular cross sectional regions (Figure 14a), the seed point is selected as the center of the minimum rectangular window, i.e. ((Xmin + Xmax)/2; (Ymin + Ymax)/2). However, for irregular shaped cross sectional regions (Figure 14b), this method can select a point outside the vessel boundary as the seed point. Therefore, the coordinate of the seed point is selected as follows:

1. The x axis coordinate of the seed point is selected as the center of the minimum rectangle window, i.e. Xseedpoint = ((Xmin +Xmax)/2).
2. Given the x axis coordinate of seed point (Xseedpoint) the voxels are scanned from top to bottom in the rectangular window along the same x coordinate. Then the connected pixels that belong to objects are grouped into vertical line segments. The middle point of the longest connected line segment is selected as the seed point.

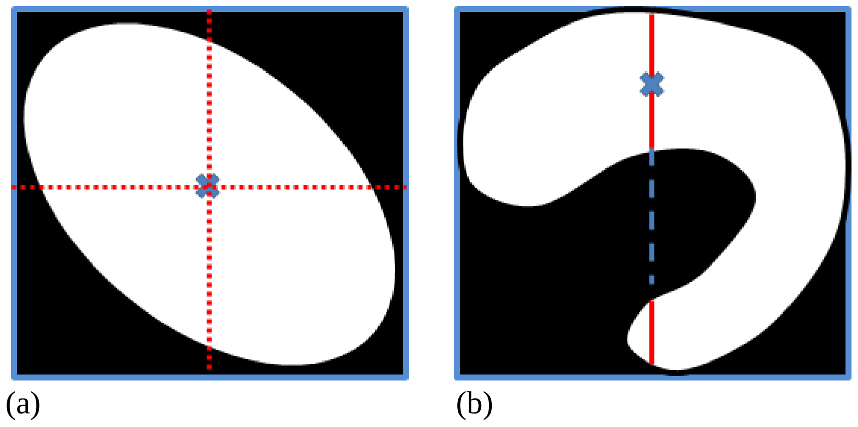


Figure 14. Seed Point selection for next layer; (a) Centroid selection in oval shaped object; (b) Centroid selection in an irregular shaped object [10]

3.2 Data analysis component

Once a blood vessel in the India ink data is traced using the tracing component, the data analysis component can be used to analyze the traced vasculature network. This component analyzes the network and displays some basic quantitative data. The main steps in the data analysis component are shown in the block diagram in Figure 15.

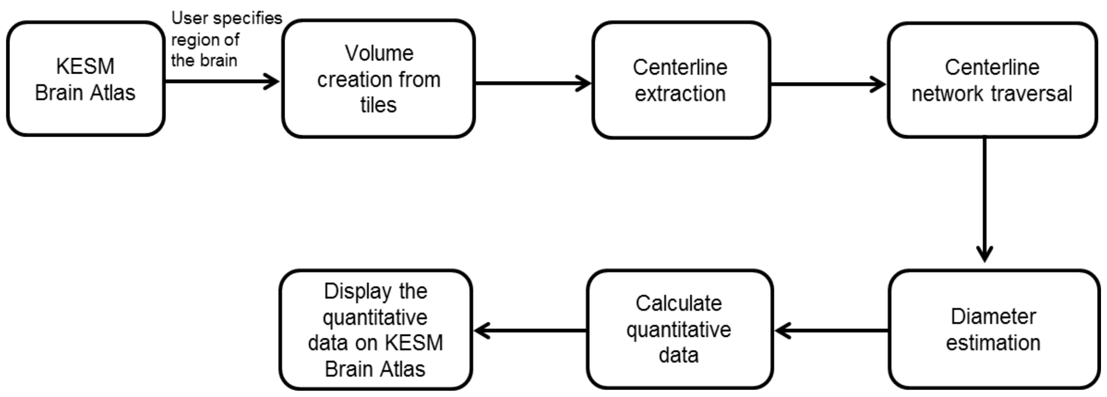

KESM Brain Atlas
User specifies region of the brain
Volume creation from tiles
Centerline extraction
Centerline network traversal
Diameter estimation
Calculate quantitative data
Display the quantitative data on KESM Brain Atlas

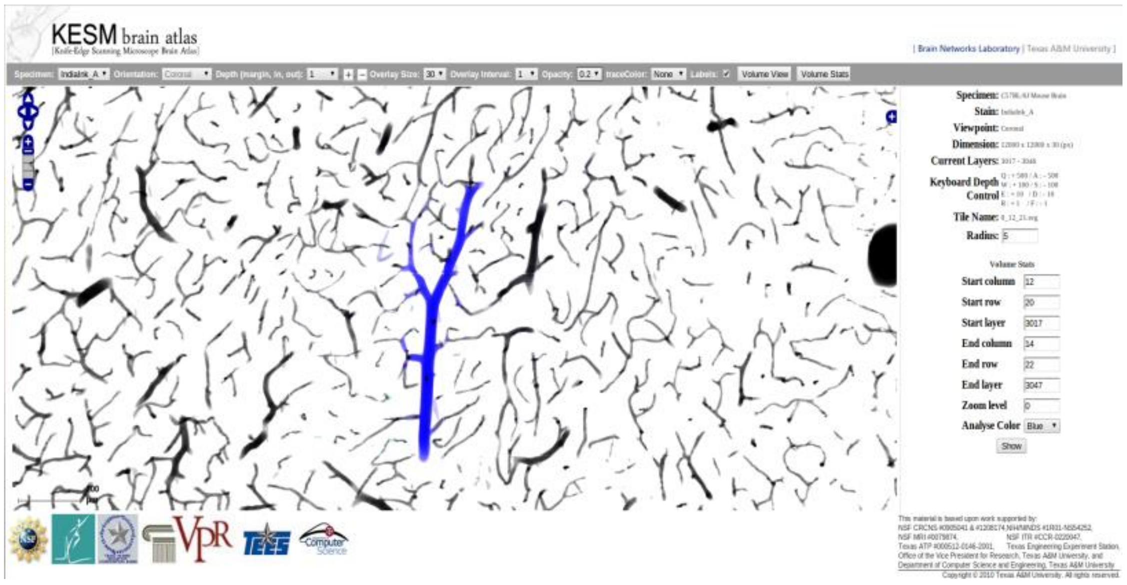
Figure 15. Block diagram for the data analysis component

To begin the data analysis, the user specifies the region in the mouse brain and the color of the vessel that he wants to analyze. The next step in the process is to create a volume from the SVG tiles in the selected region of the brain. This volume contains only the vessels that are of the same color as the color specified by the user. Next centerline extraction algorithm is used to extract the centerline from the vessels in the volume. The extracted centerline network is then traversed using depth first search traversal to label each vessel segment and count the number of branch points. The next step is to estimate diameter at each centerline voxel, and different quantitative data are calculated from the vasculature network in the volume. The extracted quantitative data are displayed on the KESM brain atlas.

3.2.1 Volume creation using the tiles

As the first step in the analysis pipeline, the user specifies a region of the mouse brain that he would like to explore. The way this is done is by specifying the starting row, column and layer number. The user also specifies the ending row, column and layer number along with the zoom level at which he wants to do the exploration. He also chooses a specific color for exploration, and only the blood vessels traced with that color are analyzed.

Then a Python script runs at the back that collates all the tiles in the user specified region of the brain. It then picks the vessels of the specified color in the tiles and creates a 3D volume. The way this code works is that it reads the tiles and then creates a Numpy array [10] with the pixel values in the tile. It then stack the Numpy arrays horizontally, vertically and depth wise to create the 3D volume. It only makes those pixels in the volume as white that are of the specified color in the tiles. An example volume is shown in Figure 16.


KESM brain atlas
[Knife-Edge Scanning Microscope Brain Atlas]

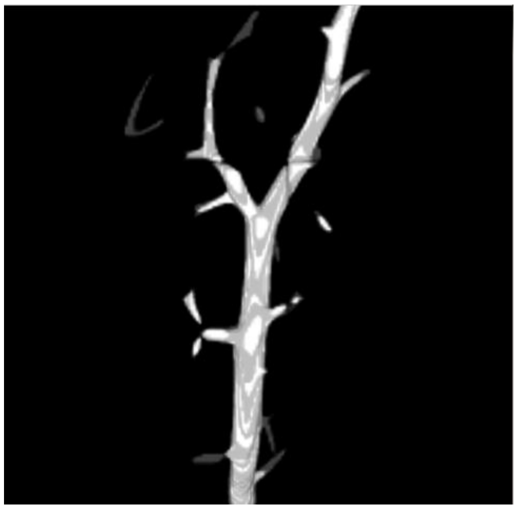
| Brain Networks Laboratory | Texas A&M University]

Specimen: Indiaink_A Orientation: Coronal Depth (margin, in, out): 1 + − Overlay Size: 30 Overlay Interval: 1 Opacity: 0.2 traceColor: None Labels: ✓ Volume View Volume Stats

Specimen: C57BL/6J Mouse Brain
Stain: Indiaink_A
Viewpoint: Coronal
Dimension: 12000 x 12000 x 30 (px)
Current Layers: 3017 - 3046
Keyboard Depth Control: Q : + 500 / A : - 500, W : + 100 / S : - 100, E : + 10 / D : - 10, R : + 1 / F : - 1
Tile Name: 0_12_21.png
Radius: 5

Volume Stats
Start column: 12
Start row: 20
Start layer: 3017
End column: 14
End row: 22
End layer: 3047
Zoom level: 0
Analyse Color: Blue
Show

(a)



(b)

Figure 16. (a) Vasculature network traced with blue color; (b) 3D volume reconstructed from the tiles.

3.2.2 Centerline extraction

The next step in the process is to extract the centerline of the vasculature network in the 3D volume. To extract the centerline of the vessels, Palágyi’s algorithm [16] is used which is a fast 3D curve-thinning algorithm that preserves line-end points and extracts both geometrically and topologically correct centerlines. Curve-thinning that is an iterative object reduction process that preserves line-end points, can directly produce one voxel wide centerlines.

The Palágyi's algorithm first classifies each voxel as border point, line-end point, or simple point based on their local neighborhood. In this case the 26-connected neighbor which is the $3 \times 3 \times 3$ neighborhood of each voxel is used (Figure 17). A simple point is a point whose removal does not alter the topology of the object. A voxel is called line-end point if it has exactly one neighbor. A voxel is called U-border point if its 6-neighbor in direction U (up) is “0”.Similarly N (North), E (East), S (South), W (West), and D (down) -border points can be defined.

Border points of a binary object that satisfy certain topological and geometric constraints are deleted in successive iterations. Each sequential object reduction process consists of six sub-iterations based on the six surfaces (Up, Down, North, South, East, and West) in 3D. This algorithm repeats the sequential process until there is no remaining border points. The detailed thinning steps are as follows:

1. Repeat for each direction U, N, E, S, W, and D.
2. Mark all border points according to the actual direction that are simple points and not line-end points.

3. For each marked point p, delete p if it remains simple and is not a line-end point after the deletion of some previously visited marked points.
4. Repeat 1-3 until no changes occur.

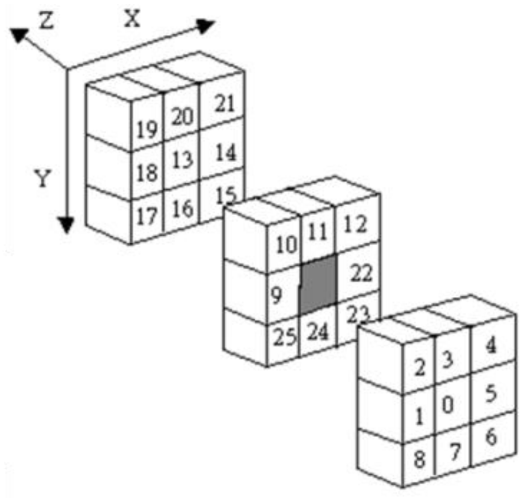


Figure 17. 26 connected neighborhood of a voxel. Adapted from [13].

3.2.3 Centerline network traversal

After extracting the centerline from the vasculature volume, the next step is to analyze this network. At first the centerlines are converted into a graph structure, where each voxel corresponds to a graph node/vertex and there is an edge between two nodes if the corresponding voxels are 26-adjacent. To analyze the network a depth first search (DFS) traversal is done on the extracted centerline network. While doing the depth first search traversal of the network, each segment of the vessel is assigned a different label.

The way this is done is as to start with a random point (v) that is not yet labeled and initialize it a label of ‘1’. Next the neighbors of the point are inserted in a stack with label of ‘1’ and the chosen point v as their parent. The following steps are repeated for all points in the stack. Pop a new point from the stack and if it has just one unlabeled neighbor (excluding its parent), then assign it the same label as the parent and pop the next point. If the popped point has more than 1 neighbors (excluding its parent), then push each of its neighbors in the stack with a different label and the current point as their parent. Mark all the neighbors of the current point as labeled. The current point is also marked as a branch point as all the normal line points have only two neighbors (including parents) and the branch points have three (Figure 18). If the stack is not empty then pop a new point from the stack and repeat the above process.

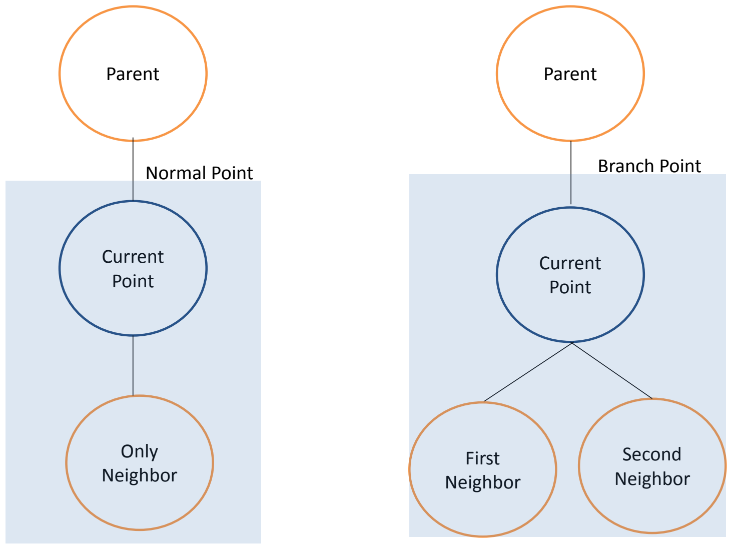


Figure 18. Line point and Branch point

The above method would traverse all the points in the network that are connected to the initial randomly chosen point. This would have been fine if the analysis was done on the entire vasculature network in the mouse brain, but since just a part of the network is being analyzed it is possible that the there are multiple connected components in the network. Hence once the depth first search traversal ends, the methods checks if there are any voxels left in the centerline network that are not yet labeled. If there is a voxel left that is unlabeled, then it starts a new depth first search from that unlabeled point. The entire process is repeated till all the points in the centerline network are labeled. The main steps of the algorithm are summarized below:

1. While there are unlabeled voxels left in the centerline network
    a. Start with a random seed point (v) that is not yet labeled and is not a branch point (has only 2 neighbors).
    b. Insert its neighbors with the same label as the initial seed point in a stack. The seed point becomes the parent of its neighbors. Mark the original seed and all its neighbors as labeled.
    c. while stack is not empty
        i. Pop a new point (v) from the stack.
        ii. If v has only one neighbor (excluding its parent) then assign it the same label as its parent. Mark the point as labeled and continue.
        iii. If v has more than 1 neighbor (excluding its parent) then assign all its children a new label and insert them in the stack.

Mark the point as a branch point. Mark all its children as labeled.

3.2.4 Diameter estimation

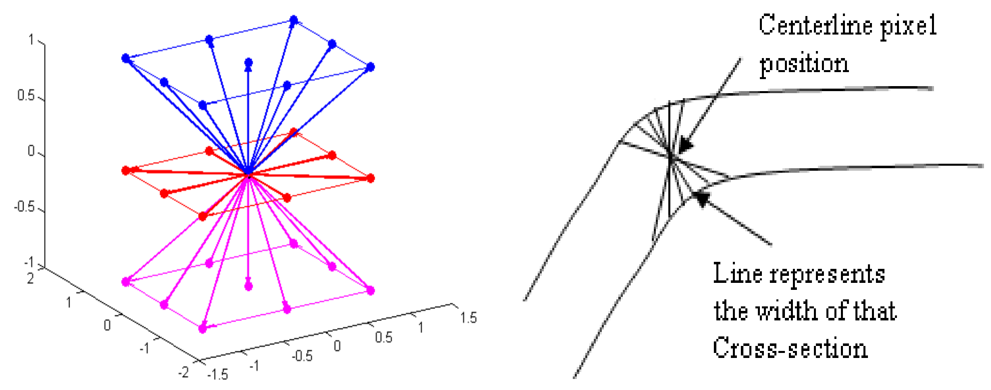
The original vasculature volume and the extracted centerline volume are used to estimate the radius of the vessel at each voxel of the centerline. To do this a voxel is selected from the centerline network and then 26 direction vectors are created around that point (Figure 19a). For any one of these direction vectors, a point is picked at a chosen radius r to see if it lies inside the vessel. The location of a point in the direction vector d, for the centerline point p at radius r is calculated as:

x = p.x + round (d.x*r);

y = p.y + round (d.y*r);

z = p.z + round (d.z*r);

If the point p lies inside the vessel then the radius r is increased till the point starts to lie outside the vascular cross-section. The radius r, at which the points start to lie outside, is the estimated radius in the direction d. The same process is repeated for all the other 25 direction vectors. In the next step the radii in the diametrically opposite directions are averaged to find the mean radius in that direction. As a final step the minimum of these average radii is picked as the radius at the centerline voxel p (Figure 19b).


1
0.5
0
-0.5
-1
2
1
0
-1
-2
-1.5
-1
-0.5
0
0.5
1
1.5
Centerline pixel position
Line represents the width of that Cross-section

(a) (b)

Figure 19. (a) 26 directional vectors; (b) Diameter at a voxel is the minimum of the diameters in different direction vectors.

3.2.5 Displaying the quantitative data

After the centerline network is traversed to label each segment and mark the branch points, and the diameter is estimated at every centerline voxel, the next step is to calculate the following quantitative data for the network and display it on the brain atlas:

1. Volume Density: The ratio of the number of voxels that belong to blood vessels to the total number of voxels in the volume. To calculate this number, the total number of non-zero voxels in the volume is divided by the total number of voxels in the volume.
2. Histogram of segment lengths: The histogram of length of segments in micrometers. After the depth first search traversal is done, the length of the different segments can be found by calculating the Euclidean distance along the voxels with a particular label. To convert this segment length which is in units of number of voxels to micrometer, the following scale is used for India ink data:

- In zoom level 0: 1 unit = 2 μm in x-axis; 2.33 μm in y-axis; 3.33 μm in z-axis.
- In zoom level 1: 1 unit = 5 μm in x-axis; 5.83 μm in y-axis; 8.33 μm in z-axis.
- In zoom level 2: 1 unit = 10 μm in x-axis; 11.67 μm in y-axis; 16.67 μm in z-axis.
- In zoom level 3: 1 unit = 20 μm in x-axis; 23.33 μm in y-axis; 33.33 μm in z-axis.

Note that this is only a rough approximation since the true voxel resolution is 0.6 μm x 0.7 μm x 1.0 μm (at the highest resolution).

3. Histogram of vessel radii: The histogram of radii of vessels calculated at every point in the centerline network. As explained above the radius of the vessel at each centerline voxel is calculated using the 26 direction vector. Then a histogram of theses radii is calculated after converting the radius from the unit of voxels to micrometers. For the calculation of radius, the resolutions in y and z axis are assumed to be same as the resolution in x–axis.
4. Number of branch points: The number of voxels where branching happened in the vasculature network. While doing depth first search traversal, the points that have more than two neighbors are marked as branch point. A count of number of such points in the network gives the number of branch points in the volume.
5. Number of voxels in original volume: The count of number of non-zero voxels in the volume.

6. Number of voxels in centerline volume: The count of number of non-zero voxels in the centerline network volume.

3.3 KESM brain atlas

The previous sections described the tracing and the analysis component in detail. This section explains how these functionalities are implemented in the KESM brain atlas. It specifically explains some of the key components in development of the interface for tracing and analysis, and its integration into the brain atlas.

3.3.1 User click coordinates

To begin the tracing process, the user clicks at a point on KESM brain atlas tile. In order to perform region growing for that tile, the coordinates at which the user clicked needs to be converted to xy coordinates relative to the tile origin.

The KESM brain atlas is built using PHP and JavaScript. JavaScript within the browser is event driven which means that it responds to interactions by generating events, and expects a program to listen to interesting events (such as "click" mouse events). Some objects within the Openlayers API [6] are designed to respond to user events such as mouse or keyboard events. To register for event notifications, addListener() event handler is added which takes an object, an event to listen for, and a function to call when the specified event occurs. These events also typically pass the UI state (such as the mouse position) as arguments to the event listener. The UI 'click' event passes the pixel location of the event (relative to the map viewport). Below is the code to register the mouse click event to OpenLayer map (g_map):

```
g_map.events.register("click", g_map, function(e) )
```

The returned pixel location relative to the map needs to be changed to pixel location relative to the tile in order to do region growing. ‘getLonLatFromPixel’ function takes in a global pixel location and returns the corresponding latitude-longitude. ‘GetileData’ function takes as input latitude-longitude and returns the corresponding tile and the pixel offset within that tile. If there is not an existing tile in the grid that covers the given location, null is returned otherwise an object is returned with the following properties: tile, i (x-pixel offset from top left of the tile), and j (y-pixel offset from top left of the tile). The script to convert a pixel location returned by the mouse click event listener to xy coordinates relative to tile is shown below:

```
function getGridCoord(evt, lyr)
{
        var ctr = g_map.getLonLatFromPixel(evt.xy);
        return lyr.getTileData(ctr);
}
var center = getGridCoord(e, b_layer);
var x_coord =  center.i;
var y_coord = center.j;
```

3.3.2 Key press listener

Another task is to add a listener to the OpenLayer map to detect the‘d’ key press event. This is needed because once region growing is done for a layer and the new tiles are displayed on the brain atlas, the next step is to wait for the user to press the‘d’ key.

Once the key press event is detected the centroid of the vessel in the current layer is taken as the seed point for region growing in the next layer.

To be able to detect the‘d’ key press event, ‘keydown’ or ‘keypress’ event listener can be used. But since both of them repeats while the user keeps the key depressed, neither of them can be used directly. Hence ‘keydown’ event listener is combined with the ‘keyup’ event listener to achieve the desired functionality of triggering the event only once even if the key is kept depressed. The idea is to use an ‘allowed’ flag that is set in the ‘keyup’ event. The ‘keydown’ event listener calls the centroid propagation function for the next layer only if the ‘allowed’ flag is set and it also resets the flag. The script to do that is shown below:

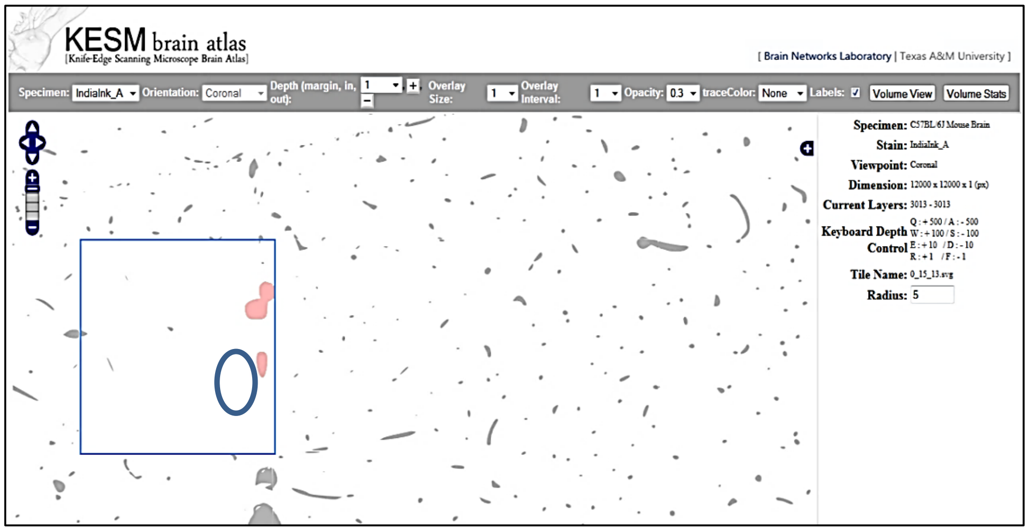
```
var callbacks = { keydown: function(evt) {
        if (!allowed){
                return;
        }
        allowed = false;
        if(evt.keyCode == 68){
                doSomething(evt.keyCode);
        }
}       };
var callbacks1 = { keyup: function(evt) {
        allowed = true;
}       };
```

3.3.3 Editing facility

Sometimes the user can make mistakes like clicking on another vessel while tracing one. Hence, the user is provided with an editing facility to correct the tracing of vessels. The way it works is that the user specifies a radius and ‘right clicks’ on a point in the traced vessel on the brain atlas. The algorithm then takes the clicked point as a seed point and performs region growing from there within the specified radius. It decolorizes all the traced voxels in that radius that are connected to the initial seed point. Figure 20 shows an example of the editing facility, where the red colored vessel is not completely decolored because the radius of 15 is chosen, if the radius is increased and the process is repeated the entire vessel will be decolored.

3.3.4 Displaying the data analysis results

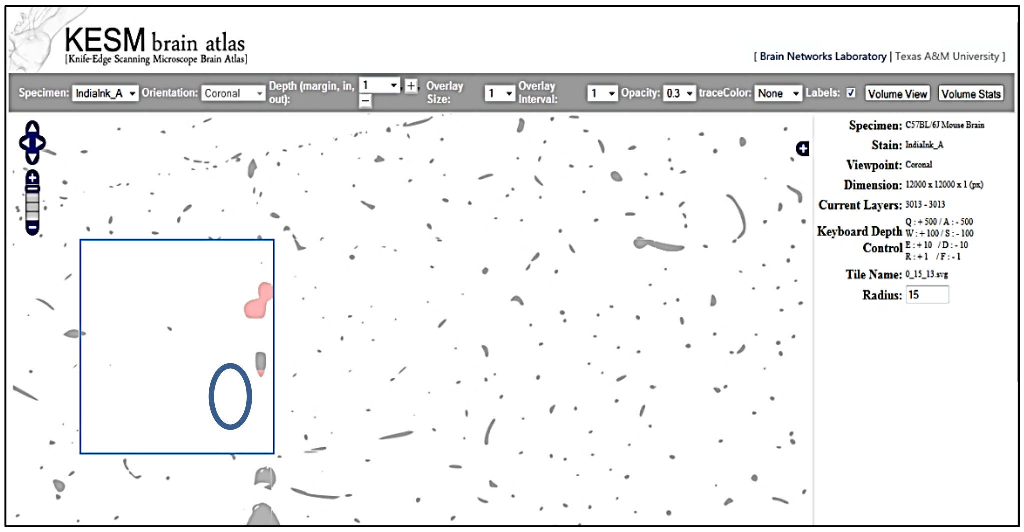
After analyzing the vasculature network in the 3D volume specified by the user, the quantitative data for the network need to be displayed on the brain atlas. For calculating the quantitative data, a Python script is written that calculates all the required statistics and writes them on a text file. It also saves the histograms plots as PNG images. Then a PHP file is created that utilizes a canvas element to display the images, and a table element to display other quantitative data read from the text file.


KESM brain atlas
[Knife-Edge Scanning Microscope Brain Atlas]
[Brain Networks Laboratory | Texas A&M University]
Volume View
Volume Stats

Specimen: IndiaInk_A Orientation: Coronal Depth (margin, in, out): 1 Overlay Size: 1 Overlay Interval: 1 Opacity: 0.3 traceColor: None Labels: ✓

Specimen: C57BL/6J Mouse Brain
Stain: IndiaInk_A
Viewpoint: Coronal
Dimension: 12000 x 12000 x 1 (px)
Current Layers: 3013 - 3013
Keyboard Depth Control Q : + 500 / A : - 500 W : + 100 / S : - 100 E : + 10 / D : - 10 R : + 1 / F : - 1
Tile Name: 0_15_13.svg
Radius: 5

(a)


KESM brain atlas
[Knife-Edge Scanning Microscope Brain Atlas]
[Brain Networks Laboratory | Texas A&M University]
Volume View
Volume Stats

Specimen: IndiaInk_A Orientation: Coronal Depth (margin, in, out): 1 Overlay Size: 1 Overlay Interval: 1 Opacity: 0.3 traceColor: None Labels: ✓

Specimen: C57BL/6J Mouse Brain
Stain: IndiaInk_A
Viewpoint: Coronal
Dimension: 12000 x 12000 x 1 (px)
Current Layers: 3013 - 3013
Keyboard Depth Control Q : + 500 / A : - 500 W : + 100 / S : - 100 E : + 10 / D : - 10 R : + 1 / F : - 1
Tile Name: 0_15_13.svg
Radius: 15

(b)

Figure 20. (a) Vessel traced in red color; (b) Traced vessel within radius 15 is decolored using editing facility.

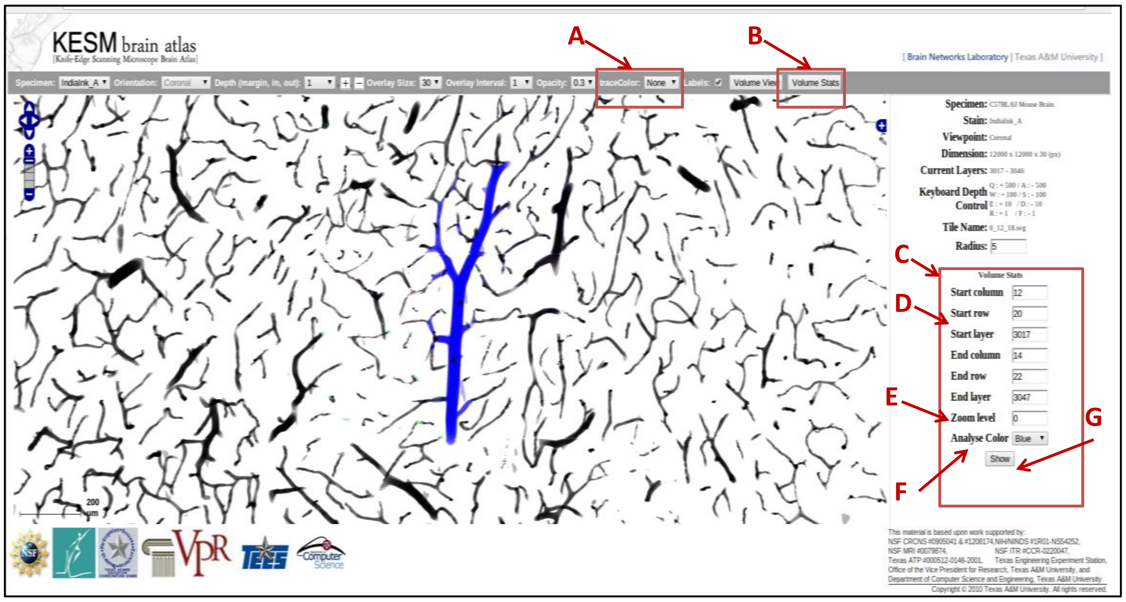
4. RESULTS AND ANALYSIS

In this chapter, results and analysis of the developed KESMBA interface will be discussed. First results of the tracing and analysis component are presented on a sub network of the India ink dataset. Next, evaluation of the data analysis component is presented. This is performed by reconstructing a new volume using the centerline and the calculated quantitative data. The accuracy is calculated by comparing the original volume with the reconstructed volume. Finally quantitative data are presented for some volumes containing vasculature network from the India ink dataset.

4.1 Tracing and analysis component

This section shows the results of the developed vascular tracing and analysis interface. In order to start the tracing, the user needs to select the color (Figure 21A) and click on a point inside the vessel that he wants to trace. Then the region growing algorithm colors the vessel as shown in Figure 21 that shows a screenshot of the KESMBA interface.

In order to get the quantitative data from the traced vessel, the user needs to click on Volume Stats button (Figure 21B). Then a new menu panel opens on the right side of the KESMBA interface (Figure 21C). Next the user needs to input the range of rows, columns, layers and zoom level information (Figure 21D, E) in the KESMBA browser window. As the user clicks on a tile on the view panel of the KESMBA, the information panel shows the location and the zoom level of the selected tile. The user also needs to choose which colored vessels (red, blue, green or all) he wishes to analyze (Figure 21F). The current implementation of the interface allows tracing in only three colors but more colors can be added easily. After the user types in the parameters and clicks the ‘Show’ button (Figure 21G), the volume statistics are calculated and displayed in a new tab.


KESM brain atlas
[Knife-Edge Scanning Microscope Brain Atlas]
[Brain Networks Laboratory | Texas A&M University]
A
B
C
D
E
F
G
Volume View
Volume Stats
Show
200 µm

Specimen: IndiaInk_A Orientation: Coronal Depth (margin, in, out): 1 Overlay Size: 30 Overlay Interval: 1 Opacity: 0.3 traceColor: None Labels: ✓

Specimen: C57BL/6J Mouse Brain
Stain: IndiaInk_A
Viewpoint: Coronal
Dimension: 12000 x 12000 x 30 (px)
Current Layers: 3017 - 3046
Keyboard Depth Control Q : + 500 / A : - 500 W : + 100 / S : - 100 E : + 10 / D : - 10 R : + 1 / F : - 1
Tile Name: 0_12_18.svg
Radius: 5

Volume Stats

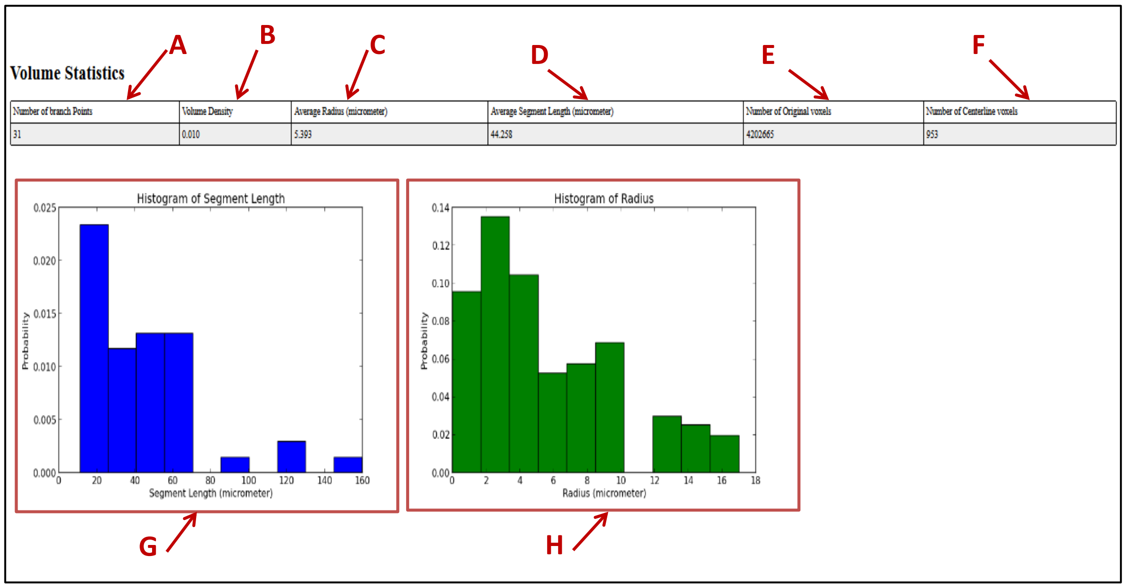
Start column	12
Start row	20
Start layer	3017
End column	14
End row	22
End layer	3047
Zoom level	0
Analyse Color	Blue

This material is based upon work supported by:
NSF CRCNS #0905041 & #1208174,NIH/NINDS #1R01-NS54252,
NSF MRI #0079874, NSF ITR #CCR-0220047,
Texas ATP #000512-0146-2001, Texas Engineering Experiment Station,
Office of the Vice President for Research, Texas A&M University, and
Department of Computer Science and Engineering, Texas A&M University
Copyright © 2010 Texas A&M University. All rights reserved.

Figure 21. Screenshot of KESMBA, Red markers and text are added on top for the purpose of explanation, A. Trace Color: Choose the color of tracing and None for no tracing; B. Volume Stats button to start the analysis of the traced vessels; C. The panel to enter the input parameters for the data analysis component and the Show button to request displaying the quantitative data; D. Start Row: The starting row of the tiles inside the 3D volume to be analyzed, similarly other information about the volume is provided in the Panel; E. Zoom level of the data at which the analysis needs to be done; F. Analyze Color: color of the vessel that needs to be analyzed, the options are Red, Blue, Green or All; G. Show button that requests the start of the data analysis algorithm

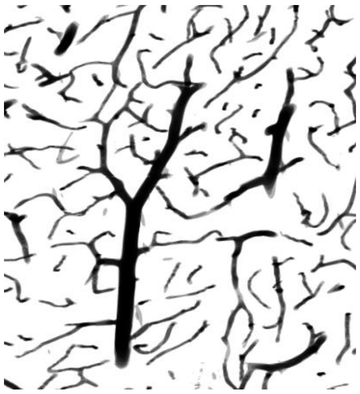
Based on the information entered by the user, all the tiles in the specified volume are collated to create a 3D volume file. Next the centerline is extracted from that volume. The radius is estimated at each voxel of the centerline, and a depth first search traversal is performed to label each segment of the centerline network. Based on these, quantitative data are calculated for the volume and displayed on the KESMBA.

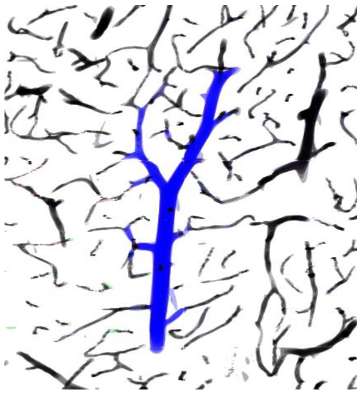
Figure 22 shows the screenshot of the web-based interface for displaying the quantitative data. The figure shows two histograms (Figure 22G and H) that displays the segment length and radius in micrometer units. In the top of the figure there is a table to display other volume statistics like number of branch points (A), volume density (B), average radius in μm (C), average segment length in μm (D), Number of object voxels in original volume (E) and number of object voxels in the centerline volume (F).

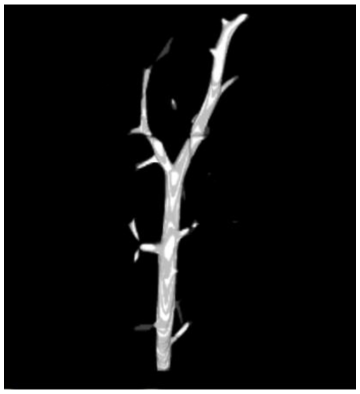

Volume Statistics

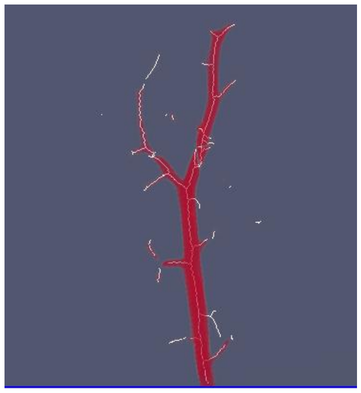
Number of branch Points	Volume Density	Average Radius (micrometer)	Average Segment Length (micrometer)	Number of Original voxels	Number of Centerline voxels
31	0.010	5.393	44.258	4202665	953

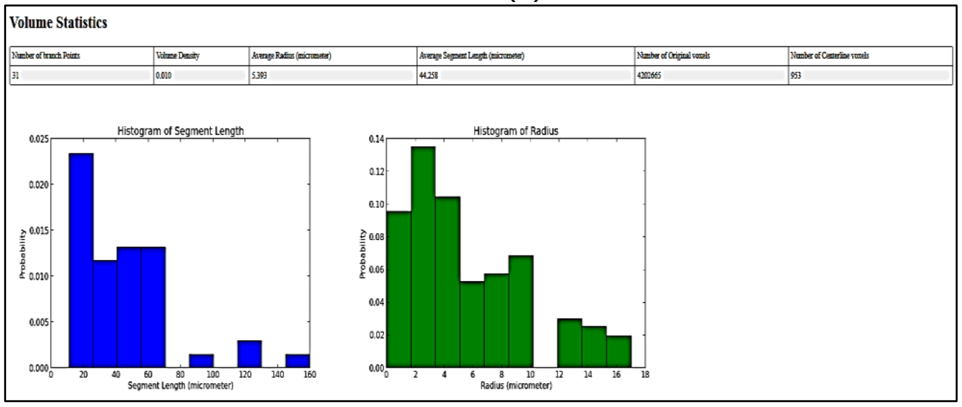
Figure 22. The KESMBA web interface displaying the quantitative data for a volume; A. Number of branch points in the volume; B. Volume density; C. Average radius of the vessels in the volume in μm; D. Average length of segments in μm; E. Number of voxels that belong to the vessel in the original volume; F. Number of object voxels in the centerline volume.


(a)


(b)


(c)


(d)



Volume Statistics

Number of branch Points	Volume Density	Average Radius (micrometer)	Average Segment Length (micrometer)	Number of Original voxels	Number of Centerline voxels
31	0.010	5.393	44.258	4202665	953

(e)

Figure 23. (a) Original volume; (b) Vessel traced with blue color; (c) 3D volume generated for data analysis; (d) extracted centerline overlaid on the original volume; (e) quantitative data of the vasculature network in the volume, displayed on KESM brain atlas.

Figure 23 and Figure 24 show the tracing results and quantitative data for two vessels traced using the KESMBA interface. In these figures (a) shows the original untraced volume, (b) shows the same volume traced with blue and green color respectively, (c) shows the 3D volume generated using the tiles from the user specified region and color (d) shows the extracted centerline overlaid on the original volume and (e) shows the KESMBA interface that displays the quantitative data for that volume.

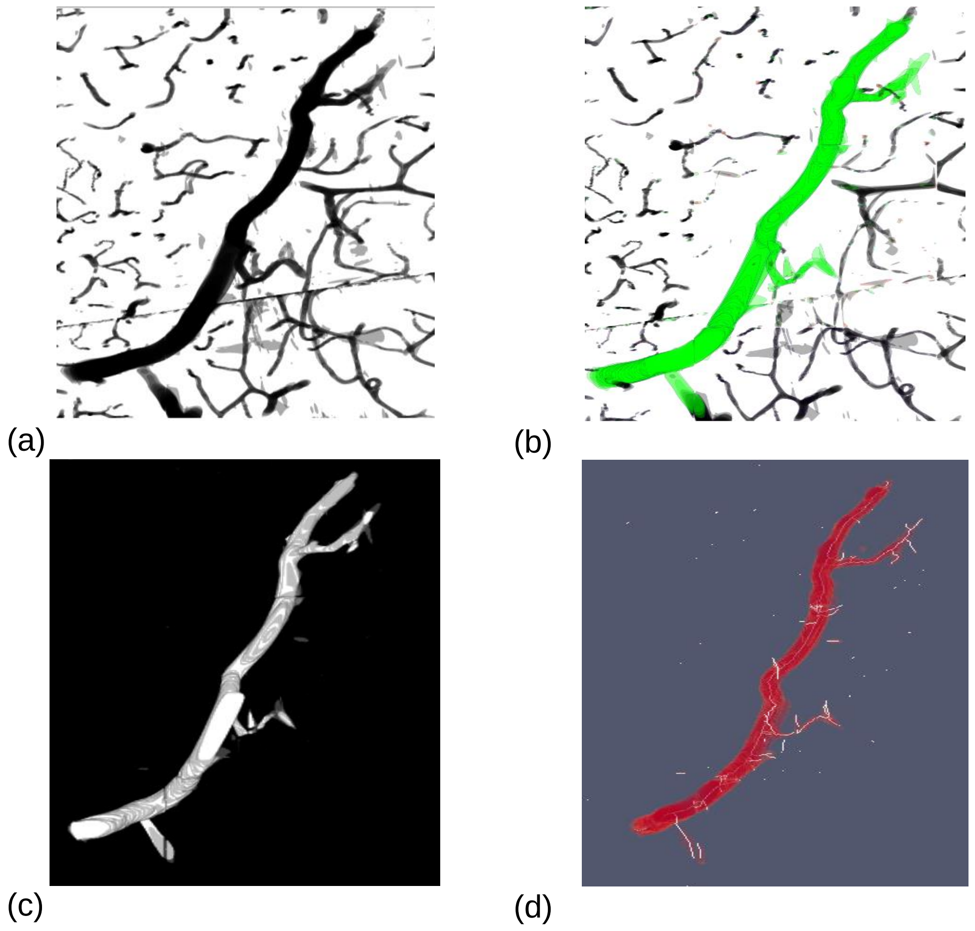
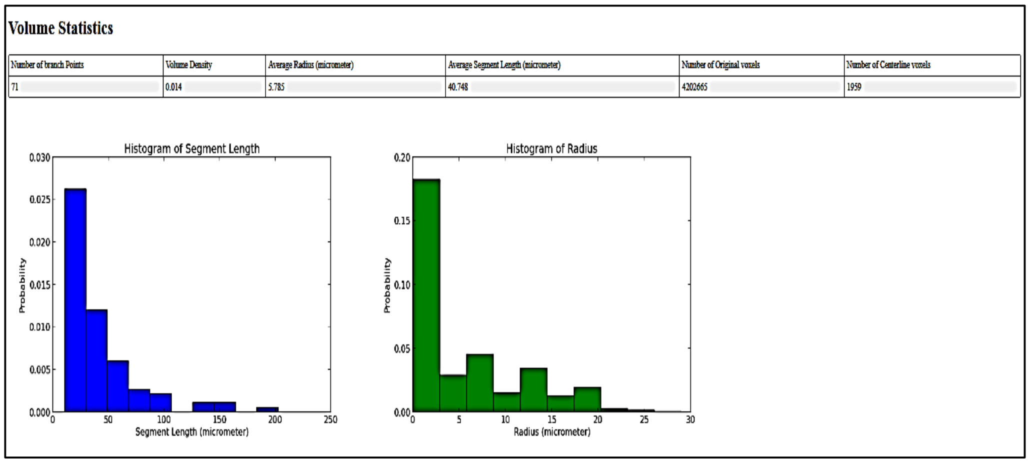


Figure 24. (a) Original volume; (b) Vessel traced with green color; (c) 3D volume generated for data analysis; (d) extracted centerline overlaid on the original volume; (e) quantitative data of the vasculature network in the volume, displayed on KESM brain atlas.



Volume Statistics

Number of branch Points	Volume Density	Average Radius (micrometer)	Average Segment Length (micrometer)	Number of Original voxels	Number of Centerline voxels
71	0.014	5.785	40.748	4202665	1959

(e)

Figure 24 continued.

4.2 Evaluation of the analysis component

To evaluate the accuracy of the data analysis component, a volume is reconstructed using the extracted centerline and the radius calculated at each voxel of the centerline. To reconstruct the volume, a sphere of radius equal to the estimated radius is placed at each centerline voxel. Figure 25 shows the original volume and reconstructed volume of sizes 50×50×50 and 186×186×186 voxels. As is evident from the figures, the reconstructed volume looks structurally very similar to the original volume but the reconstruction is not perfect.

From Figure 25a and Figure 25b, it appears that the reconstruction has overgrown the vessels. In Figure 25c and Figure 25d, the reconstructed volume shows many gaps that are missed out when compared with the original volume.

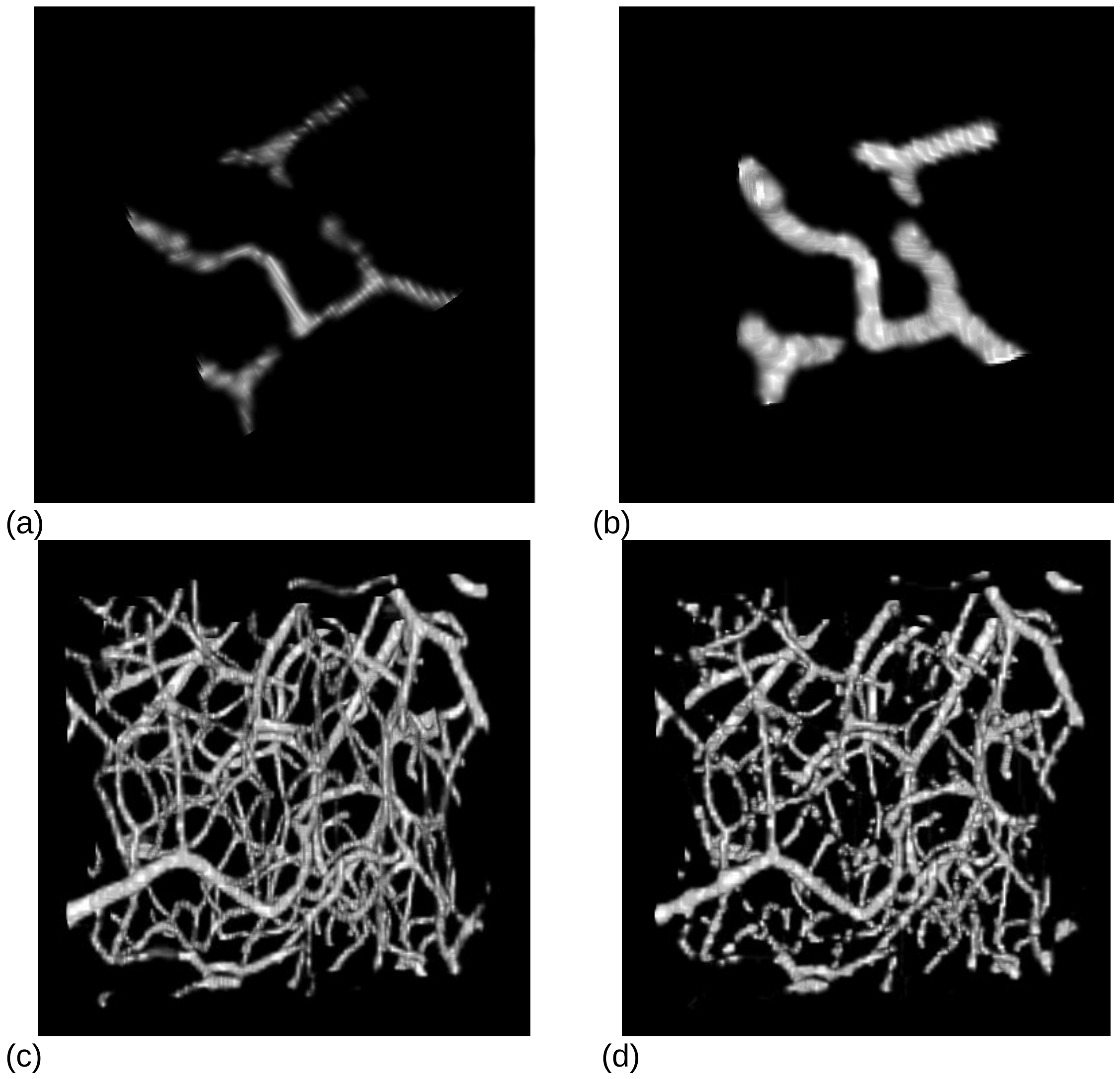


Figure 25. Volume reconstructed using the centerline and the estimated diameter at each voxel; (a) Original volume of size 50×50×50 voxels; (b) Reconstructed volume of size 50×50×50; (c) Original volume of size 186×186×186 voxels; (d) Reconstructed volume of size 186×186×186

Although visually the original volume and the reconstructed volume appear to be structurally similar, in order to quantify the performance of the analysis component, precision and recall metrics are calculated by comparing the reconstructed volume to the original volume as:

$$Precision = \frac{\text{Number of voxels where both original and reconstructed volume had a non zero value.}}{\text{Number of voxels where reconstructed volume had a non zero value.}}$$

$$Recall = \frac{\textit{Number of voxels where both original and reconstructed volume had a non zero value}}{\text{Number of voxels where original volume had a non zero value.}}$$

Next F1 score is calculated as:

$$F1\ Score = \frac{2*Precision*Recall}{Precision+Recall}$$

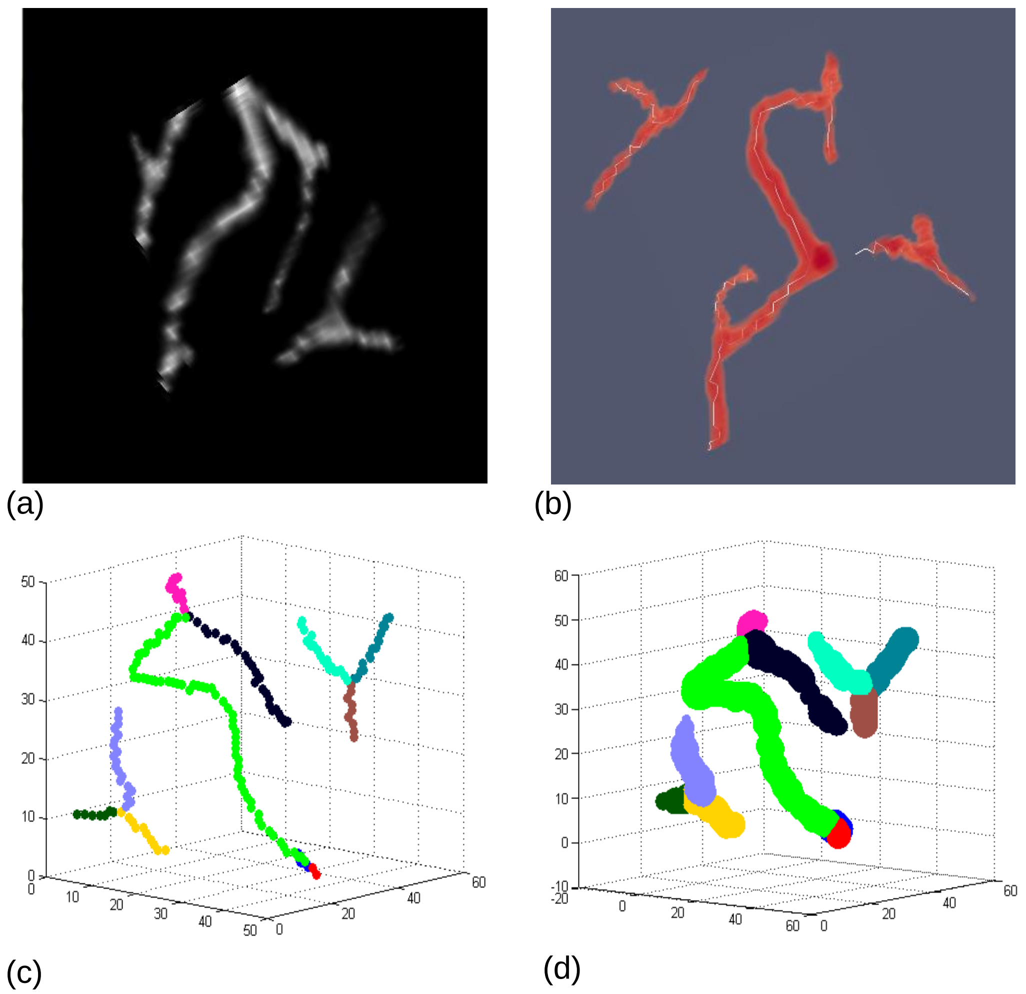
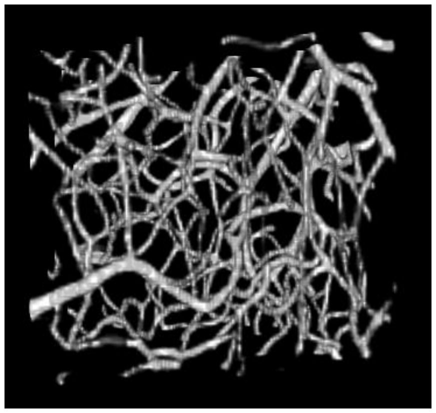
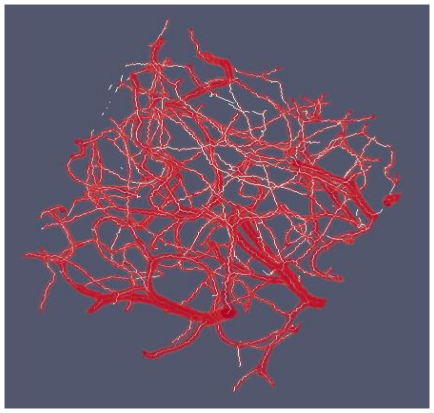
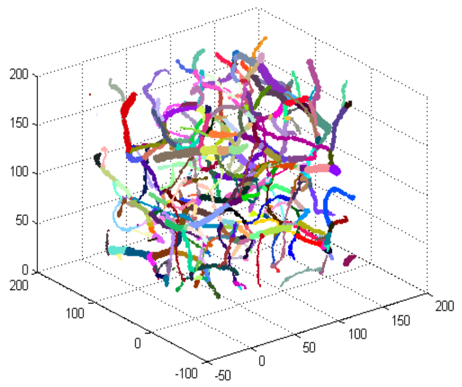


Figure 26. Volume reconstruction (a) Original volume of size 50×50×50 voxels; (b) Extracted centerline overlaid on the original volume (original vessel: red color; centerline: white color); (c) Extracted centerline where each segment is labeled with different color; (d) Reconstructed volume where each segment is labeled with different color

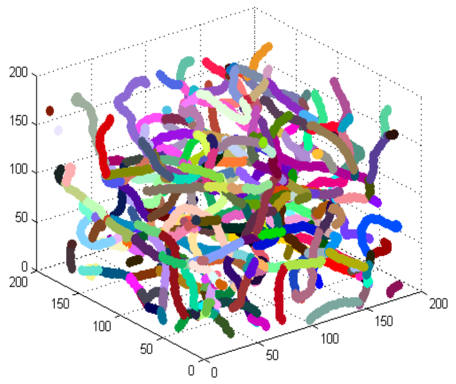
To evaluate the data analysis pipeline, 8 volumes are created from the KESM India ink data at zoom level 0. Next the centerline is extracted for these volumes. Radius is estimated at every centerline voxel using the extracted centerline and the original volume. A new volume is reconstructed using the centerline and the estimated radius by placing a sphere at each centerline voxel. Figure 26 and Figure 27 shows the process of reconstruction for volumes of size 50×50×50 and 186×186×186.


(a)


(b)



(c)



(d)

Figure 27. Volume reconstruction (a) Original volume of size 186×186×186 voxels; (b) Extracted centerline overlaid on the original volume (original vessel: red color; centerline: white color); (c) Extracted centerline where each segment is labeled with different color; (d) Reconstructed volume where each segment is labeled with different color

In order to reconstruct the volume, first the centerline is extracted. Figure 26b & Figure 27b show the extracted centerline in white color overlaid on the original vessels (red color). Then each vessel segment is labeled by performing depth first search traversal over the centerline network (Figure 26c and Figure 27c). Finally, the calculated radii of the vessels at each centerline voxels are used to reconstruct the volume (Figure 26d and Figure 27d).

The precision, recall and F1 scores are calculated by comparing the original volume with the reconstructed volume. The scores for the 8 volumes are shown in Table 1. The volumes are chosen to vary from 40×40×40 to 512×512×512.

Table 1. Precision, Recall and F1 score for the volumes reconstructed using the centerline and radius calculated by the data analysis component.

X	Y	Z	Recall	Precision	F1
40	40	40	0.8092	0.8966	0.8506
50	50	50	0.8256	0.8816	0.8527
180	180	180	0.7738	0.7531	0.7633
186	186	186	0.7168	0.8305	0.7695
256	256	100	0.7095	0.8556	0.7757
256	256	256	0.5692	0.8596	0.6849
512	512	100	0.5403	0.8575	0.6629
700	700	186	0.7266	0.8324	0.7759
512	512	512	0.7261	0.8527	0.7843

Another experiment is performed where the radius at the centerline voxel is allowed to be fractional. In this case, to calculate the radius at a centerline voxel, the location of a point in the direction vector d for the centerline point p at radius r is calculated using:

x = p.x + round (d.x*r);

y = p.y + round (d.y*r);

z = p.z + round (d.z*r);

where the radius r varies in steps of 0.1 compared with 1 in previous case. Table 2 shows the precision, recall and F1 scores for 8 volumes when the radius is allowed to be fractional.

Table 2. Precision, Recall and F1 score for the volumes reconstructed using the centerline and radius (which is fractional) calculated by the data analysis component.

X	Y	Z	Recall	Precision	F1
40	40	40	0.7278	0.9437	0.8218
50	50	50	0.7408	0.9391	0.8283
186	186	186	0.6412	0.9059	0.7509
256	256	100	0.6648	0.8988	0.7643
256	256	256	0.5237	0.9236	0.6684
512	512	100	0.4987	0.9174	0.6462
700	700	186	0.6822	0.8662	0.7632
512	512	512	0.6677	0.9087	0.7698

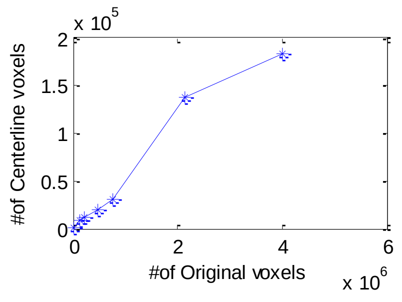
From Table 1 and Table 2, it is clear that for both the integer and the fractional radius precision is better than recall. This observation aligns with the results shown in Figure 25 where the reconstructed volume appears to be broken and has gaps when compared with the original volume. The possible reason could be that the minimum radius among all the direction vectors is chosen as the radius at the centerline voxel. The other possible reason could be that instead of approximating the vessel as cylinder it is assumed that it is made up of individual spheres kept at each centerline voxel. This could be causing the gaps in the reconstructed volume and hence low recall numbers. But the integer radius numbers look better that the fractional radius number for all the 8 volumes, hence integer radius is used for further analysis.

Table 3. Quantitative data extracted from the vasculature network present in the 8 volumes.

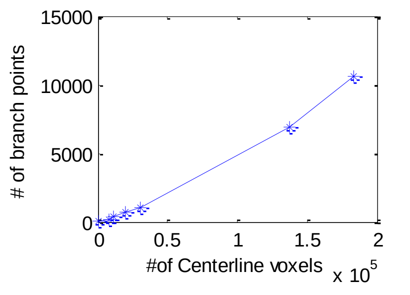
X(μm)	Y(μm)	Z(μm)	Volume density	Average Radius (μm)	Average Segment length (μm)	# Original Voxels	# Centerline Voxels	# branch points
80	93.3	133.3	0.046	6.33	70.11	2924	66	1
100	116.7	166.7	0.053	5.57	59.80	6663	208	4
372	434	620	0.019	3.36	78.44	119797	8284	197
512	597.3	333.3	0.033	3.13	44.03	216853	11540	429
512	597.3	853.3	0.028	3.59	57.01	476504	19545	683
1024	1194.7	333.3	0.029	3.56	63.07	754798	30842	1084
1400	1633.3	620	0.024	3.97	65.97	2153021	137818	6858
1024	1194.7	1706.7	0.030	3.35	74.70	4002010	183617	10558

4.3 Volume statistics

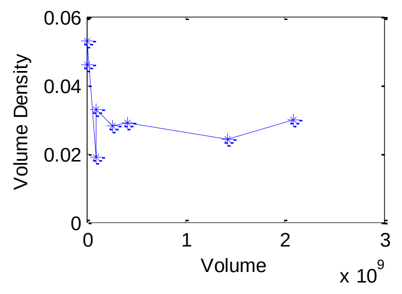
For the 8 volumes, the following quantitative data is calculated: volume density, average segment length in μm (after removing segments which are smaller than 5 voxels in length), average radius in μm (after removing the top and bottom 10 percentile), number of voxels in the original volume that belongs to the blood vessel, number of voxels in the centerline volume and number of branch points. Table 3 shows the statistics for the 8 volumes.



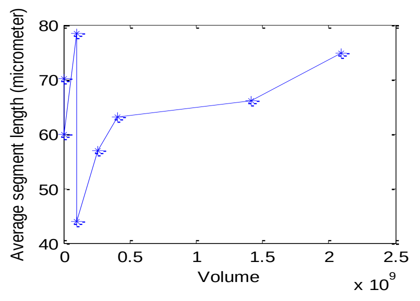
(a)



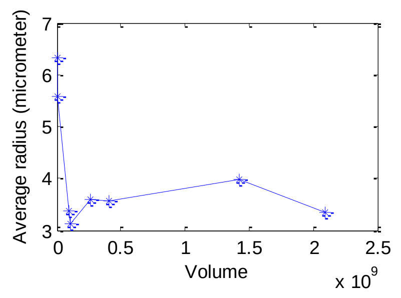
(b)



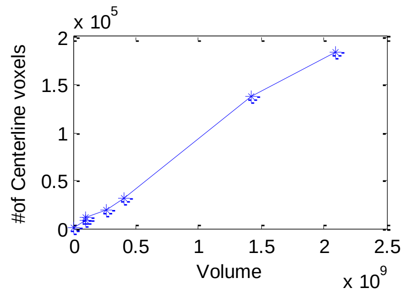
(c)



(d)



(e)



(f)

Figure 28. Plots of the quantitative data for the 8 volumes; (a) Plot of number of centerline voxels vs number of voxels in the original image; (b) Plot of number of branch points vs number of centerline voxels; (c) Plot of volume density vs size of the input volume in voxels; (d) Plot of average segment length vs size of input volume; (e) Plot of average radius vs size of input volume; (f) Plot of number of centerline voxels vs size of input volume

Figure 28 shows plots of the extracted quantitative data for the 8 volumes. Few interesting observations can be made from the figure. The plot in Figure 28a shows that initially the number of centerline voxels increases quickly as the number of voxels in original volume is increased but later the increase in the number of centerline voxels is less. Figure 28b shows that the number of branch point increases sublinearly with number of voxels in the centerline volume. Figure 28c, d, and e show that as the input volume size is increased the volume density, average segment length and average radius remains more or less constant. The mean radius is slightly larger for the smaller volumes; this could be due to a violation of the assumption that the resolutions in y and z axis are same as the resolution in x–axis. Hence depending on the orientation of the vessels in the given volume, the method can overestimate/underestimate the mean radius. Figure 28f shows that as the volume size is increased the number of centerline voxels increases quickly initially but the increase is less towards the end.

5. DISCUSSION

This thesis presented a vasculature tracing and analysis framework to allow fast and efficient tracing of blood vessels in the India ink dataset. The analysis component helps in analyzing the traced vessels in a user specified region of the brain and get some basic quantitative data. In this chapter contribution, open issues, and future work are discussed.

5.1 Contribution

The developed KESMBA interface allows a user to quickly and efficiently trace large volumes of vascular network in the brain. Since the user can visualize the 3D structure of the data on the brain atlas, he can guide the algorithm in correctly tracing the vessels. The user guided tracing method requires limited human intervention. The user just needs to click on a point in the vessel to start tracing, then he can just continue to press a key to trace the vessel in the subsequent Z slices. The traced vessels are stored on the KESMBA server and can be used as ground truth for training supervised algorithms. The traced vessels can also be used for validation of other automated tracing algorithms. The analysis component extracts key quantitative data from the user specified volume of the brain and displays it on the KESMBA. The evaluation of the analysis component on the volumes of the India ink data shows that it performs well.

5.2 Open issues

In this section some open issues related to the developed KESMBA interface are discussed.

5.2.1 Spurious branches in the centerline

The centerline of the vessels extracted using Palágyi’s algorithm [16] contains several false segments. These side branches are present in results of all skeletonization algorithms because these algorithms are sensitive to coarse object boundaries or surfaces. An example is shown in Figure 29 where the red circles show some of the side branches.

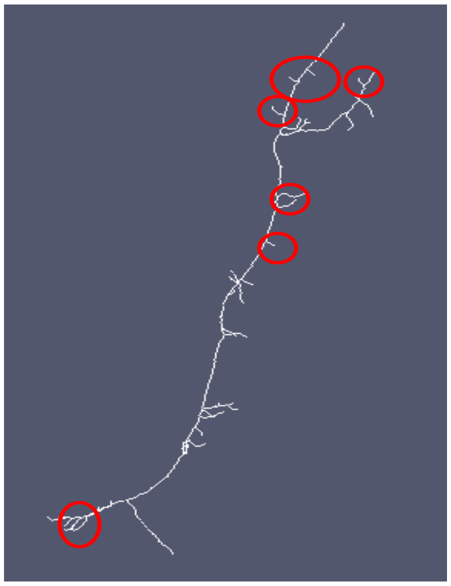


Figure 29. Side branches in the extracted centerline. The red circles show some of the false segments.

A reliable centerline that is free of the side branches is critical for the data analysis component. Unwanted branches cause false segment labeling and consequently lead to false measurements of statistics like segment length, radii, number of centerline voxels, branch points etc.

There are pruning approaches (e.g., morphological pruning [17]) capable of removing all side branches that are shorter than a predefined threshold [18]. Palágyi, Tschirren, Hoffman and Sonka [16] presented a pruning technique for tubular segments of varying thickness that is capable of removing long parasitic branches from thick parts and preserving short correctly determined branches in thin segments. Spurious branches are identified by assessing the distance-from-surface and the branch length.

5.2.2 False outlines after SVG conversion

Sometimes while converting the colored PNG tile to SVG, certain small object gets a different boundary color. In the step when color separation is done using ImageMagick, certain object's outline get a different color. Figure 30 shows a tile where the small object boundaries are colored green. This causes issues while doing the analysis of vessels of a specific color. Some of the calculated quantitative data might be incorrect. More research needs to be done to find out the cause of this issue.

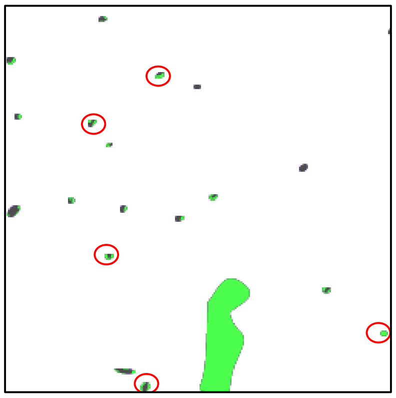


Figure 30. SVG tile where boundary of certain small objects is colored. The red circles show some of the objects whose outlines are colored.

5.2.3 Tracing where the vessel splits

At certain places in the vasculature network, the vessel cross-section in a layer branches into two or more segments in the next layer. The current implementation just traces one of the vessel branches in the next layer. An example is shown in Figure 31, where the green colored vessel splits into two branches in the next layers and with the current implementation only one of the branches is traced further. This needs to be modified in a way that if the vessel splits, all the segments of the branched vessel can be traced in parallel.

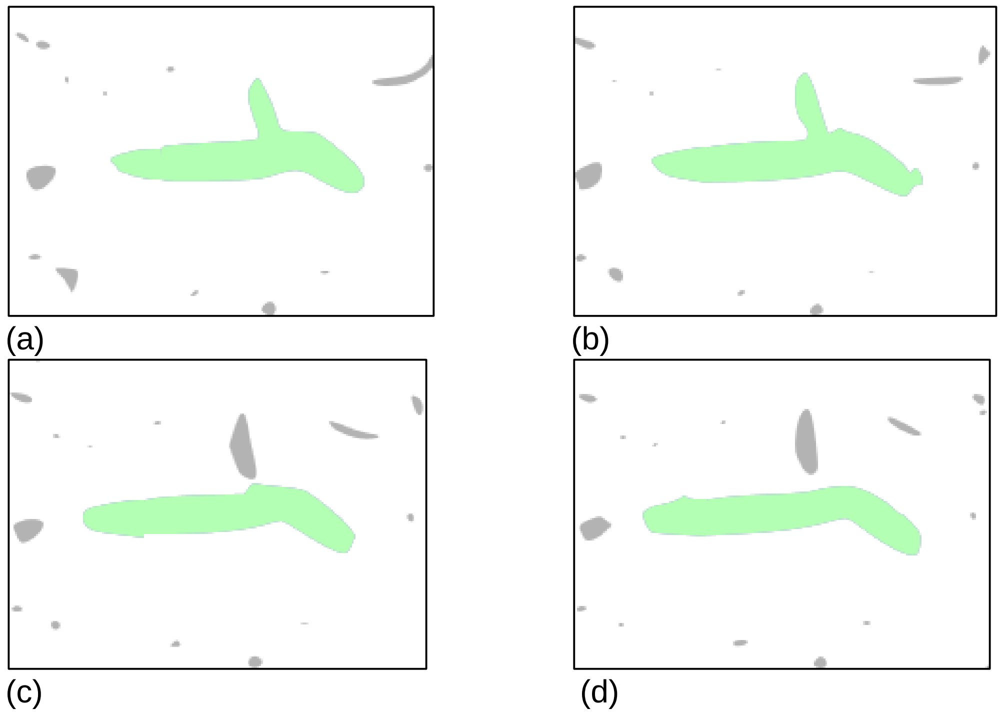


Figure 31. Branching vessel (a) Cross section of a vessel as seen in a layer; (b) Cross section of the same vessel in the next layer; (c) The same vessel branches into two segments in the next layer and only one of them is traced; (d) The two segments (one traced and one untraced) move further away from each other in the next layer

5.3 Future work

In this section some directions for future work are discussed.

5.3.1 User registration

User login can be implemented for the KESMBA tracing and analysis interface. With this facility the tracing done by a user can be restricted to be available only to the user, to a group or to everyone. With user registration, a ranking system can be developed for users to maintain integrity, where each user’s rank could be based on the tracing done by the user. User ranking can be based on a trust-based method where if the user’s tracing matches the majority then his ranking is boosted. The other option can be to recognize certain users as experts, and those users whose traced results agree with the experts get higher ranks. Similarly if a vessel is traced by multiple users it will be associated with a high quality score. This can be used to implement a filter capable of dynamically selecting more reliable tracings.

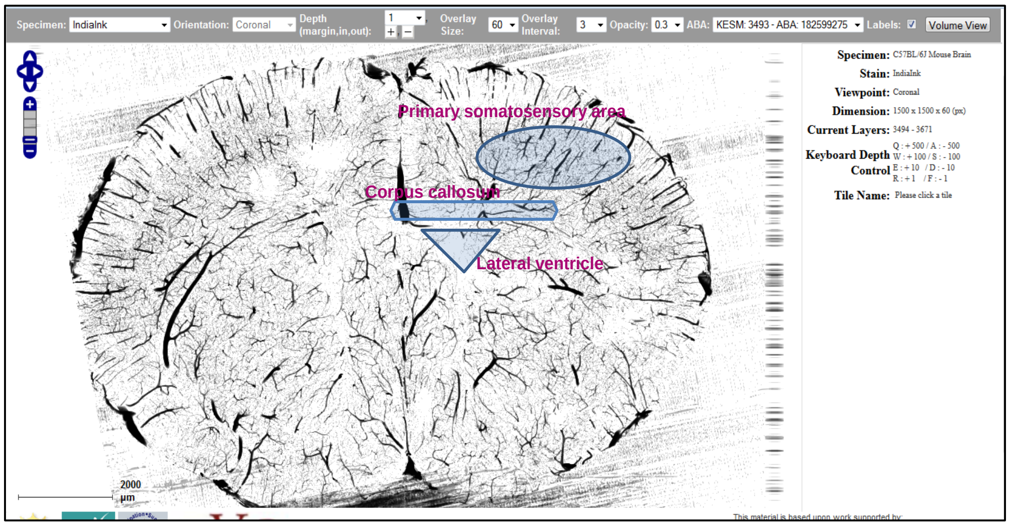
5.3.2 More quantitative data for the analysis component

More volume statistics can be calculated for the traced vasculature network. Some of the other quantities that might be interesting would be total number of loops in the network and number of paths between a pair of points in the network. These quantities will provide more information about the complexity of the network. Other quantities like diameter ratio at the branch points, branching angles, vessel orientation can also be added. Some sort of tortuosity measure for a segment can also be an interesting quantity to calculate. It can be approximated as the ratio of segment length (in pixels) to the Euclidean distance between the end points (in pixels).

5.3.3 Richer annotation

To better serve the purpose of information sharing, rich annotation is necessary to create and store interpretations of the data. Annotations can be in form of user-specified content (textual, graphical) attached to a region on the KESM brain atlas. Few examples of annotations can be anatomical names, physiological characteristics, related literature citations, etc.

The intrinsic features of the OpenLayers support various types of overlay objects which are tied to the map tiles with latitude and longitude coordinates. The types of overlay objects include icons, polygons, polylines, and markers. Geometric primitives can be implemented so that regions of interest can be marked by pointing, clicking, and dragging the mouse on the zoomable web-based display. Depending on the primitive (point, line, oval, rectangle, polygon, or free-form closed contour), control points can be recorded and stored in an XML file associated with each annotation. Figure 32 shows a mockup of annotation drawings on the KESMBA.


Specimen: IndiaInk Orientation: Coronal Depth (margin,in,out): 1 Overlay Size: 60 Overlay Interval: 3 Opacity: 0.3 ABA: KESM: 3493 - ABA: 182599275 Labels: ✓

Specimen: C57BL/6J Mouse Brain
Stain: IndiaInk
Viewpoint: Coronal
Dimension: 1500 x 1500 x 60 (px)
Current Layers: 3494 - 3671
Keyboard Depth Control: Q : + 500 / A : - 500; W : + 100 / S : - 100; E : + 10 / D : - 10; R : + 1 / F : - 1
Tile Name: Please click a tile

Figure 32. Mockup of annotation on the KESMBA

5.3.4 Interactive display of the traced vessels in 3D

Sometimes the same vessel gets split into two or more branches in the next layer. It is visually difficult to find out that which branch segment is part of which vessel unless the entire context is known.

It would be very helpful for the user if the currently traced vessel can be shown in 3D in a separate window. This would help the user in visually analyzing the traced vessel as it grows, and correct the mistakes that are made while tracing.

A 3D-unit-volume viewer [5], capable of providing an interactive view and analysis of the volume data, is already implemented in the KESMBA. It allows users to interactively access the different angles of view of an image volume. The unit volume viewer can be adapted to show the currently traced vessel in 3D. The other option would be to integrate ImageJ , Amira or MeVisLab with the KESM brain atlas and use them to show the traced volume in 3D (Figure 33).

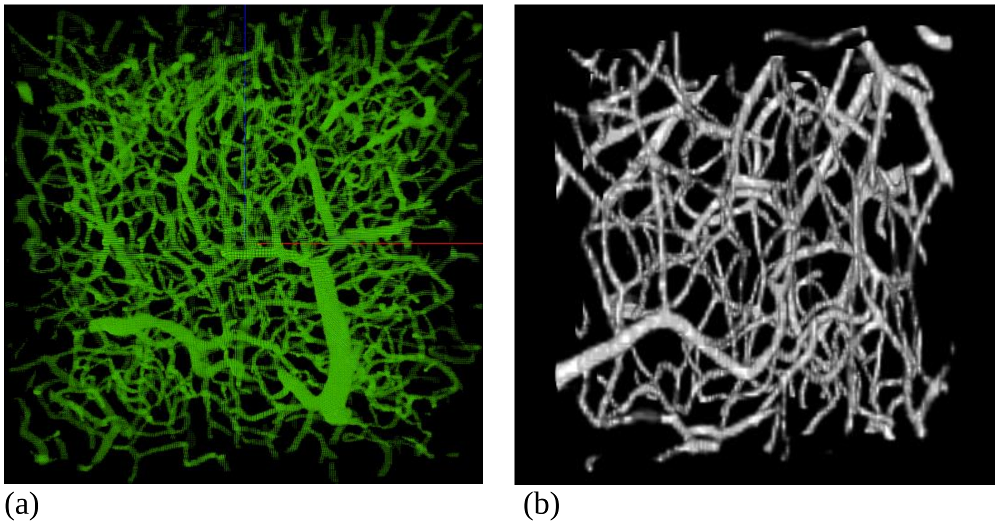

(a) (b)

Figure 33. 3D Visualization of the volume (a) The KESMBA unit volume viewer [5]; (b) ImageJ

5.3.5 Increase the speed

The developed interface takes few seconds to trace a vessel cross section in a Z slice. This latency is mainly due to the fact that all the tiles in a layer are fetched from the server every time. This happens because of the random number component in the URL of the tiles. The URL of a tile is host_name/ Indiaink/ Zslice#/ tile#. In the current implementation, the new URLs are generated as host_name/ Indiaink/ Zslice#_Random# / tile#. The random number is added to the name of the Z slice and hence all the tiles in that layer needs to be re-fetched from the server. This completely disables the browser's caching.

A solution to this issue would be to add random number to each tile name rather than the Z slice number. The new URL can be created as host_name/ Indiaink/ Zslice#/ tile#_Random#. After processing a tile, its random number component would be changed while for other tiles the random number component would remain unchanged. This new implementation would then fetch only the newly processed tile from the server and get the other tiles from the browser's cache.

6. CONCLUSION

In this thesis, an interface is developed as part of the KESM brain atlas to allow quick and efficient tracing of vasculature network in the mouse brain KESM India ink data. To trace a vessel a user needs to click on any point inside the vessel once, and then he can just continue to press a key to trace the vessel in the next layers. The traced vessels are immediately visible to the user on the atlas and an editing facility is also provided to correct any mistakes made by the algorithm. The user can trace different vessels with different colors to separate them for analysis.

Furthermore, a data analysis component is also developed to analyze the traced vessel. The analysis component extracts the centerline for the vasculature volume and displays some useful quantitative data for the volume.

Evaluation of the data analysis pipeline is presented by reconstructing a new volume using the extracted centerline and estimated radius. The original volume is compared with this reconstructed volume to find metrics like precision, recall and F1 scores. The method performs well on all the test volumes taken from the India ink dataset. Other quantitative data like number of voxels in the centerline network, number of branch points, average radius, average segment length, volume density etc. are calculated and plotted for these test volumes to observe interesting relationship among these quantities. In summary, the developed interface enables efficient tracing and analysis of the KESM India ink data and possibly other similar data.

REFERENCES

[1] Y. Choe, D. Han, P.-S. Huang, J. Keyser, J. Kwon, D. Mayerich, and L. C. Abbott, "Complete submicrometer scans of mouse brain microstructure: Neurons and vasculatures.," *Neuroscience Meeting Planner Chicago, IL: Society for Neuroscience. Program No. 389.10.*, 2009.

[2] H. W. Dong. (1999-2015, 03/22). *The Allen reference atlas: A digital color brain atlas of the C57Bl/6J male mouse*.

[3] Y. Choe, L. Abbott, D. Han, P. Huang, J. Keyser, J. Kwon, D. Mayerich, Z. Melek, B. McCormick, and A. Rao, "Knife-edge scanning microscopy: high-throughput imaging and analysis of massive volumes of biological microstructures," *High-Throughput Image Reconstruction and Analysis: Intelligent Microscopy Applications*, pp. 11-37, 2008.

[4] J. Kwon, D. Mayerich, Y. Choe, and B. H. McCormick, "Automated lateral sectioning for knife-edge scanning microscopy," in *5th IEEE International Symposium on Biomedical Imaging: From Nano to Macro, 2008. ISBI 2008.* , 2008, pp. 1371-1374.

[5] C. Sung, "Exploration, registration, and analysis of high - throughput 3D microscopy data from the Knife-Edge Scanning Microscope," Texas A&M University, College Station 2013.

[6] O. Org. (2012). *OpenLayers 2*. Available: http://openlayers.org/two/

[7] Y. Choe, D. Mayerich, J. Kwon, D. E. Miller, J. R. Chung, C. Sung, J. Keyser, and L. C. Abbott, "Knife-edge scanning microscopy for connectomics research," in *The 2011 International Joint Conference on Neural Networks (IJCNN)* 2011, pp. 2258-2265.

[8] D. Han, "Rapid 3D tracing of the mouse brain neurovasculature with local maximum intensity projection and moving windows.," Texas A&M University, College Station 2009.

[9] D. M. Mayerich and J. Keyser, "Filament tracking and encoding for complex biological networks," in *Proceedings of the 2008 ACM symposium on Solid and physical modeling*, 2008, pp. 353-358.

[10] W. Yang, "Automated Neurovascular tracing and analysis of the Knife-Edge Scanning Microscope India Ink Dataset," Texas A&M University, College Station 2014.

[11] V. Marx, "Neuroscience waves to the crowd," *Nature methods,* vol. 10, pp. 1069-1074, 2013.

[12] E. team. (2013). *EyeWire*. Available: http://eyewire.org/

[13] I. T. G. Laboratory. (2014). *Accessing Image Pixel*. Available: https://clouard.users.greyc.fr/Pandore/c++doc/html/neighbours_page.html

[14] I. S. LLC. (1999-2015). *ImageMagick*. Available: http://www.imagemagick.org/script/index.php

[15] P. Selinger. (2001-2015). *Potrace*. Available: http://potrace.sourceforge.net/

[16] K. Palágyi, J. Tschirren, E. A. Hoffman, and M. Sonka, "Quantitative analysis of pulmonary airway tree structures," *Computers in biology and medicine,* vol. 36, pp. 974-996, 2006.

[17] R. C. Gonzalez and R. E. Woods, "Digital image processing," ed: Prentice hall Upper Saddle River, NJ, 2002.

[18] S.-Y. Wan, A. P. Kiraly, E. L. Ritman, and W. E. Higgins, "Extraction of the hepatic vasculature in rats using 3-D micro-CT images," *IEEE Transactions on Medical Imaging* vol. 19, pp. 964-971, 2000.