UNIVERSITY OF CALIFORNIA
Santa Barbara

A Visual Analysis Toolkit for Microscopic Image Mosaics of Retinal Astrocytes

A Thesis submitted in partial satisfaction
of the requirements for the degree of

Master of Science

in

Computer Science

by

Panuakdet Suwannatat

Committee in Charge:

Professor Tobias Höllerer, Chair

Professor Steven Fisher

Professor B. S. Manjunath

September 2014

The Thesis of
Panuakdet Suwannatat is approved:

Professor Steven Fisher

Professor B. S. Manjunath

Professor Tobias Höllerer, Committee Chairperson

September 2014

A Visual Analysis Toolkit for Microscopic Image Mosaics of Retinal Astrocytes

by

Panuakdet Suwannatat

To Upper Ojai Search and Rescue Team

Acknowledgements

I am forever thankful to my family especially Nok, Piyatida Klumphu, also known as Aunt Bird[1] to my little nephews Pennueng and Fame. I owe my survival during the last year at UCSB, including my diet, exercise and spiritual well being to her. Nok's contribution to this project reaches almost every corner. The joy of coding while listening to her beautiful ukulele music, the sound of pounding mortar, the sizzling smell of garlic, and the sound of a chemist editing away her next paper is among the most fascinating creative experience I've ever had. Many thousand lines of Java, R, and Matlab code owe their existence to Nok for creating such a peaceful state in my mind. Nok also provided aesthetic critiques to many visual designs and served as a sounding board for my thesis defense.

This thesis and the whole project behind it would not have been possible without the extraordinary support of my advisor, Professor Tobias Höllerer. His passion and energy for the research has greatly inspired me to UCSB and to this project. The scope of this work, the scale of data, and the level of details have many times sent me into a wrong corner of the forest – trekking in a wrong direction, dwelling in irrelevant details, or simply getting stuck in quicksand. Tobias has brought me back every time without ever losing his temper or his belief in me, not even once. I am extremely fortunate to have an advisor who always keep an eye out for the big picture while knowing and caring enough about the details. Every research discussion we have is as much philosophical as it is practical. Although I have learned a lot from him about conducting research, I have learned even more from his personal style of dealing with people. Always friendly, always polite, always calm, Tobias has shown me how we can treat each other with decency while still asking tough questions, raising concerns, resolving conflicts, defending our work and being genuinely confident. I consider the lessons I learned from his examples to be the most valuable part of my education here.

I am extremely thankful to my other committee members including Professor B. S. Manjunath and Professor Steven Fisher. Professor Manjunath is the Principal Investigator for the NSF grant #0808772, "Working with Uncertain Data in Exploring Scientific Images," which has supported this project. I am very grateful to his unwavering support, his insights during our project meetings, and for connecting me with other students at the Center for Bio-image Informatics which he directs. Many of them, Brian Ruttenberg and Aruna Jammalamadaka in particular, became my close friends and collaborators who are extremely important in the completion of this project. Professor Manjunath also connected me with

[1] "Nok" in Thai means "bird." "Nok" is also the name of a chemical that she helped discover, also known as SPGS-550-M [1]. That's how cool she is!

Professor Fisher whose Retinal Cell Biology Lab at the Nueroscience Research Institute is home to a treasure trove of large-scale biological images, many of them never been seen anywhere else, that serve as input to our visualization system. Analyzing those images is an exciting and daunting endeavor, but his expertise and his exceptional team of researchers have helped me tremendously. I am also very grateful for his arrangement for me to have a desk in his lab where the center of all the actions are: data collection, annotation, programming, debugging, gathering user's feedback, paper editing, and many lively discussions. The lab is also home to many great people and I really enjoyed their friendship. I especially enjoyed listening to KDB and discussing classical music with Ken Linberg whose desk was next to mine. All the way up on the 6^{th} floor, the lab also has the nicest view of Goleta Beach where Gabe Luna and I sometimes went for a photo walk[2].

Gabe Luna is a great friend and a colleague. All the microscopic images presented in this work were collected and curated by him. As a photographer as well as a researcher, he worked the tissue samples and the microscope with expertise, precision, and passion. I am grateful to him for not only giving me a chance to work with him on the data he had spent countless hours to collect, but also for letting me in on the data collecting process itself. As a coach, he let me try mounting, cleaning, staining, and imaging a few retinas. It was a very difficult task requiring fine-motor skills, steady hands, good mental focus, and absolutely no sneezing. This has contributed to my deep appreciation of the data and the uncertainty behind it. Gabe was also actively involved in the design of the visualization system, providing me with the description of tasks, limitations of current tools, interesting ideas, and valuable feedbacks to hundreds of iterations of the program. With his biological expertise and his long experience with the astrocytes data, he was instrumental in creating almost all of the human annotations as mentioned throughout this thesis, e.g. cell center location, retina boundary, damage masks, major blood vessels masks, single cell injection domain polygons, etc. His contribution to this project has been extremely important and cannot be understated. As a friend, he and I have great passion for nature and night time photography. I enjoyed the many photo trips we made together – from aircraft spotting at SBA to rocket chasing at Vandenberg AFB.

I would like to thank all my friends at the Retina Cell Biology lab, the Bioimage Lab, and the Four-Eyes Lab. Dr Goeff Lewis has collaborated with me during the early part of this project. His insights and ideas are highly appreciated. I have also frequently consulted with him on necessary biological background. He is also an avid biker and I really enjoyed talking with him about the local biking and hiking trails. Dr Brian Ruttenberg, who was my "cube mate" during the

[2]Gabe is a Canon. I am a Nikon. But we got along just fine.

early year, gave me the first introduction to the data and research angles for this project. The first version of Retivis was developed in close collaboration with him, making extensive use of his data preparation materials. His random walk segmentation program is still being used in the current version, and he has responded promptly and helpfully to my emails asking for advice even after he had left UCSB. During my last year, Aruna Jammalamadaka, a recent PhD, has been an essential collaborator. Her statistical insights and her get-things-done work ethics pushed the analytical side of this project into a more rigorous territory. Her feedbacks on the region-based analysis tools and the scripting interface of Retivis have helped me refine and improve them. Aruna hand-annotated the veins and arteries in 7 of the 8 retinas and offered an independently discovered observation regarding the large size of cells along the veins. Aruna also collaborated with me on the single cell injection study where we screened the quality of cell images together and cross-checked our cell center locations data. During that study, Aruna implemented adaptive threshold binarization which has become very useful in the final analysis. Her expertise in computer vision, statistics, R, and Matlab, and her great friendship are invaluable to me and this project.

Many faculty members in the CS Department have helped and mentored me over the years. Some have worked with me on a previous project. I give my special thanks to Professor Matthew Turk who co-directed the Four Eyes Lab.[3] Matthew also taught me Artificial Intelligence and Computer Imaging which I enjoyed very much. His teaching style informs as well as inspires. He and his wife are also a great host during our lab party. Professor Xifeng Yan taught me Data Mining, and has given me advice regarding the network study in Chapter 3. Xifeng is also a member of my Major Area Exam committee, along with Professor Kevin Almeroth. Kevin collaborated with me when I worked on a visualization system for mobile computing network. Together with him, Professor Elizabeth Belding and Dr Amit Jardosh, we developed SCUBA, a visualization system for wireless mesh network [2]. The experience from working on that project has greatly helped me in this project and beyond. I have also sought advice from Professor Amr El Abbadi who taught me Distributed System. The course was both challenging and exciting, and the materials that I learned have proven useful when I decided to parallelize many aspects of Retivis. The Computer Network course that I took with Professor Ben Zhao was also very helpful. In that course, I experienced for the first time the true power of computing when we learned to control hundreds of computers in the PlanetLab global network at the same time during our research on peer-to-peer network. We also learned about the true power and reach of RIAA who, despite being very protective of Ms Britney Spears' music, ended

[3] 4 i's = Imaging, Interaction, and Innovative Interfaces. `http://ilab.cs.ucsb.edu`

up supporting and appreciating our research out of respect for Ben's scholarly prestige and diplomatic skills. I learned a lot from him and from many other professors here.

I would like to thank my parents for giving me my life and a loving family. In my early childhood, dad and I had daily beach walks during which we discussed science – the shape of the earth, the ocean, the tide, electricity, and even special relativity[4]. Mom raised me at her small beach-side gift shop and she taught me to add, subtract, multiply and divide during her business transactions. Dad drew me to science and Mom drew me to Math. Both of them gave me the strength of will to carry on. I only wish they could be here in person as in spirit. I love them and I am thankful for the life they gave me. I thank them and my entire family.

I consider my dear friend Nammon, the late Dr Mananya Tantiwiwat, a part of my extended family. I thank her for the many hiking trips we made together – some of them extremely risky – which gave me ample opportunity to learn from her perspective on life, the universe, and everything. During her time on earth, she is my primary source of consultation: on English, research, life, relationship, family, culture, politics, and physics, to name a few. I am also very thankful to her parents who have taken a very good care of my mother during my family leave. Without them, I could not have carried through such a difficult period and still come back to this study.

My landlord, or rather, my "host family," in Santa Barbara, has been extraordinarily supportive of me. I thank Aida and John Shellabarger for accepting me into their wonderful house. They taught me to hike and backpack, and the Santa in their house gave me my first high-quality hiking pants, among other things. They also taught me a lot about Santa Barbara, American culture, family values, and US history. Although I never quite learned how to cook, they had a good microwave[5].

Many of my Thai friends in the USA have helped me with almost everything in life. I hereby acknowledge and thank them all for their help and their friendship. Although I cannot mention all the names, I must single out Pam, Dr Ornkanya Yaoharee, and her husband Klang, Colonel Farland Citylove[6] of SBPD (Sara Buri Police Department) because of their exceptional expertise in Life, and in knowing exactly what I think/feel before I do. This has allowed them to "life-coach" me into doing the best for myself and for people around me on several occasions. I really cannot thank Pam and Klang enough.

[4]I did not understand.

[5]which they generously gave me after I left.

[6]a pseudonym, for reasons of national security

I thank Mathias Kolsch for the LaTeX template in which this thesis is made. I thank my friend Witchakorn Kamolpornwijit, of MIT and Dropbox, who gave a few extra gigabytes; they were put to good use in the development of this project. Many thanks to Isla Vista Foot Patrol officers whose quick responses to my calls and whose enforcement of the noise ordinance greatly helped the writing process. Thanks also to my flight instructors, Andrea Read and Matthew Berlo, who inspired me to always *aviate, navigate, and communicate* in Cessna-172 as well as in life.

This thesis is dedicated the Upper Ojai Search and Rescue Team, a non-profit volunteer organization dedicated to saving lives since 1951 – one of them mine. Thanks for rescuing me from the snow covered cliff at Potrero John, so I could survive to write this thesis. My appreciation to them is beyond words.

I am grateful to the support and guidance from the Office of Educational Affairs, the Royal Thai Embassy, and the many institutions of the Royal Thai Government dedicated to promoting science education including the Development and Promotion of Science and Technology (DPST), the Institute for the Promotion of Teaching Science and Technology (IPST), the Office of The Civil Service Commission (OCSC), the National Science and Technology Development Agency (NSTDA), and the Junior Science Talent Project (JSTP).

This work is funded in part by NSF grant #0808772.

Abstract

A Visual Analysis Toolkit for Microscopic Image Mosaics of Retinal Astrocytes

Panuakdet Suwannatat

Analyzing high-resolution images of astrocytes is important in understanding diseases, such as glaucoma and retinal detachment, to which astrocytes are known to become reactive. Yet, little is known about the statistics of astrocyte patterns over the area of an entire retina.

We developed an interactive visualization system supporting the visual analysis of microscopy mosaics of entire mouse retinas, and present preliminary findings from the study of 8 full retina mosaics that were imaged in UCSB's Retinal Cell Biology Lab. We present the design and implementation of this visualization system, which required several steps of data preparation and normalization. We will briefly discuss our choices and validation approaches for parameter selection in the image normalization, segmentation, and visualization steps of our pipeline. We created tools to support the generation and visual comparison of possible network structures interconnecting the astrocyte cells. We will report our initial insights from using these tools on the 8 retinas.

Together with our biology collaborators, we identified 9 properties of astrocyte cells that held most promise to shed light on underlying principles and statistics

in astrocyte distributions, and we will present the most interesting patterns that emerged from our visual analysis.

Contents

List of Figures

List of Tables

Part I

Software System

Chapter 1

Interactive Visualization Tool

Retinal astrocytes are one of two types of glial cells found in the mammalian retina. In addition to being involved in retinal vascular growth [3], formation of neuronal synapses, and the control of energy supply to neurons [4], astrocytes play an important role in diseases and injuries: glaucomatous neurodegeneration [5] and retinal detachment [6]. Studying astrocytes may elucidate their role in pathological conditions, yet there is a lack of tools for visualizing astrocytes effectively. Although biomedical imaging techniques have improved greatly since Stone and Dreher studied the distribution of astrocytes in 1987 [7], work analyzing astrocyte images such as [3] still visualizes astrocyte network as a whole without distinguishing between individual cells.

In mice, these highly planar cells are located in the innermost retinal layer termed the *nerve fiber layer* and are robustly stained using anti-glial fibrillary acidic protein (GFAP). Using laser scanning confocal microscopy, whole retinal

datasets were captured at high resolution and subsequently assembled into seamless montages. This produces very large images for quantitative and qualitative analysis[1].

As data is gathered at higher resolution, the need for an interactive visualization tool designed specifically for astrocytes is paramount. Such a tool must integrate relevant image processing techniques, visualize each piece of data in context of the whole set, and be able to communicate uncertainty in the data. In this chapter, we present an integrated system that has been developed in two major stages: a prototype and the second generation. We will introduce the data and visualization pipeline that can handle uncertainty in probabilistic segmentation. Finally, we present how our region-based analysis tool allows the users to explore and discover potential insights in large data sets.

1.1 Data pipeline

The data pipeline is shown in Figure 1.1. The steps are explained below.

[1]Parts of this chapter have previously been reported in [8] and [9].

1.1.1 Tissue staining and imaging

Eight large retinal mosaics [2] are prepared and provided to us by our biologist collaborators, Gabe Luna and Professor Steve Fisher of the Retinal Cell Biology Lab, Neuroscience Research Institute, UCSB. In each of these images, a mouse's retina tissue was fixed and stained with anti-GFAP for astrocytes and anti-collagen IV for blood vessels . As the astrocytes cytoskeleton contains the protein GFAP, all astrocytes in the retina were visible. GFAP expression in astrocytes and related issues are discussed in more details by [10]. The retina was then wholemounted and the astrocyte layer imaged at 40X magnification on a laser scanning confocal microscope Olympus FluoView 1000. Multiple overlapping sections are captured into a single large mosaic using the bio-imaging software `Imago` [11].

The image sizes range from 217.26 (GFP1) to 324.53 megapixels (GFP13) with an average size of 280.95 megapixels per retina[3]. Each pixel has a physical dimension of 0.309697 μm.[4]

[2]The full-size mosaic is kept in the file `100percent.png` under the directory `AstrocyteRoot/retName/images`.

[3]The sizes are reported from the set of 8 retinas. Any blank margins are cropped away before the sizes are calculated.

[4]0.309697 μm is the width and height of a pixel. The physical area occupied by a pixel is $0.309697^2 = 0.09591223181\ \mu m^2$.

1.1.2 Cell location identification

The centers of the astrocytes in the images were manually marked by the biologist. The red stains near the cell nuclei served as a guide for the human to confirm.

The visualization system allows the cells to be imported from an Excel file outputted from Imago. They can also be added or removed interactively by right-clicking and choosing the appropriate command from the context menu.[5] In the initial system design, cell identification is done completely in the first pass. In later iterations, it was revealed certain visualization modes such as in Figure 1.9(d) allow previously unmarked cell center to become visible[6] even after the biologist had believed that all the cells had been marked and segmented. The data pipeline was modified to allow additional cells to be added [7] even after segmentation. The segmentation processes for newly added cells are started automatically in the background.

Since the nuclei of transgenic mice express GFP, the nuclei are always located inside red blobs. Although many previous works have addressed the issue of automated nuclei detection [12] [13] [14], the results still need to be checked manually by the biologist. The existence of noises in the red channel especially near the

[5] The cell locations are kept at `allcells.txt` under the `data` directory.
[6] as a gray cell among multi-hue neighbors with 100% saturations.
[7] with a "new cell here" command of the mouse context menu

cuts is problematic for an automated approach. Hence, the biologist has decided to perform this step manually.

1.1.3 Cell segmentation

The cells are segmented using a probabilistic method, as described in [8], which is based on random walk [15]. The random walk program, implemented by Brian Ruttenberg [16], works on each individual cell independently of other cells. For each cell, the random walk agent starts from the cell center and randomly walks toward its neighbors. The probability of stepping into each neighboring square depends on its pixel intensity. To prevent agents from traveling too far into other cell's territory, the agent is occasionally reset back to the original cell center location[8]. The number of times each pixel is visited as a proportion of the total number of steps taken is recorded as a probability for that pixel. The process is repeated until the probability map starts to converge.

This process takes approximately 1 - 2 minutes per cell. Although it is computationally intensive, it is also trivially parallelizable [17]. Memory usage is also relatively small. We have built a system to launch the random walk segmentation program simultaneously with the number of instances equal to the number of CPU

[8]This parameter, called the `restart` probability, is set at 5×10^{-5}

cores [9]. For legacy reason, the random walk program is run with `OpenCV` version 2.3 under the `Ubuntu` operating system inside a `VirtualBox` virtual machine[9].

In the latest version, the cells can be added or deleted interactively in the visualization software. Accordingly, the segmentation is modified so that each cell is segmented on demand, still taking full advantage of multiple CPU cores. The visualization client communicates with the segmentation server (which can be on the same or on a different machine) via a shared disk space. Multiple servers can work together to segment the retina. It has been argued in [18] that, for a distributed system to work smoothly, a real-time performance monitoring is required. While performance debugging techniques for distributed system of black boxes have been previously discussed in [19], we can customize and integrate our own progress monitoring with a simple visualization. The progress of segmentation, the cells being actively segmented, and the overall number and locations of segmented cells are continuously monitored and visualized so the user does not have to wait until the process is finished before exploring the retina. The time-lapse video at [20] demonstrates this monitoring process.

Another segmentation method based on adaptive thresholding has recently been developed by Aruna Jammalamadaka [21]. It is also fully integrated and

[9]The random walk program does not run on OpenCV version 2.4.7 or later.

compatible with our software system.[10] The file structure of the segmentation results allow for future extensions by external tool to support a different segmentation algorithm such as a watershed approach [22] [23] [24].

1.1.4 Segmentation results re-assembly

As shown in Figure 1.1, the segmentation results for individual cells are assigned random colors (hues) and put back together. The background color and transparency is interactively configurable while rendering to screen. Certain segmentation parameters have to be chosen at this step. This includes the cut-off threshold and a boolean flag indicating whether to use continuous gradual coloring to reflect the probability map or to use a solid color.

1.2 Early prototype

A prototype version is initially developed to test the viability of our visualization techniques. To reduce complexity, only a quarter of the retina – one of the four wings – is considered. Furthermore, the resolution is reduced by 50% in each dimension, which reduces the number of pixels to only 25%. This allows us to focus on designing the visualization and interactions to convey data and

[10]The segmentation results are kept under the `rwalk` directory with `stack.raw` being the raw output from Random Walk Segmentation and `ImageBW.png` being the grayscale image of cell as converted from the raw output or as produced directly by another segmentation method.

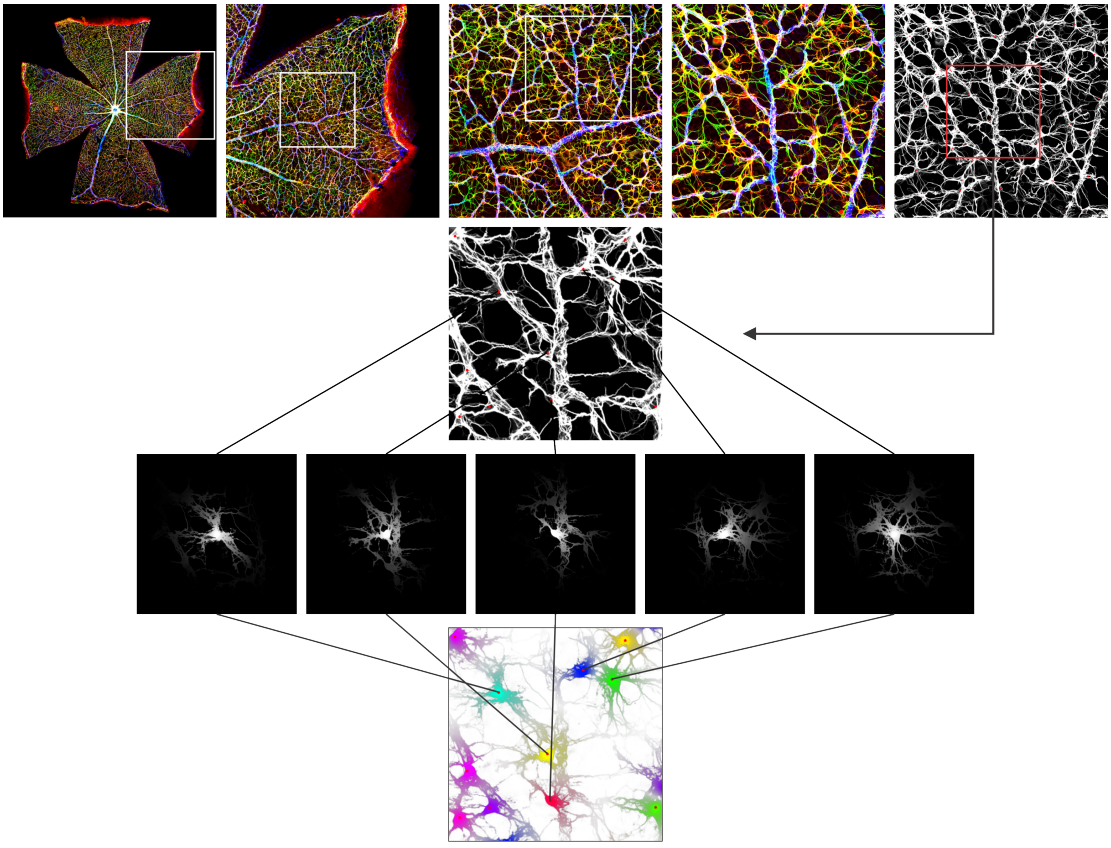


Figure 1.1: The visualization pipeline

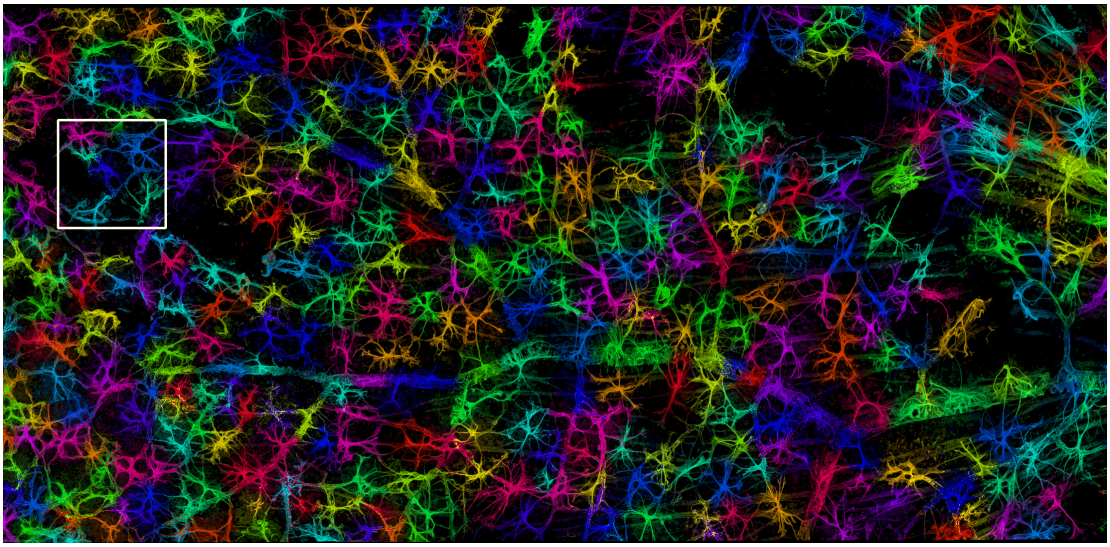


Figure 1.2: Visualization of astrocytes in the early version

its uncertainty. We then address potential performance problems with increased resolution in Section 1.3.

1.2.1 Visualization techniques

The visualization's primary goal is an in-context presentation of data so underlying patterns can emerge. This includes patterns of cell shapes, wrapping patterns around blood vessels, and patterns of cell density. These patterns are brought to the surface in Figure 1.2.

To show the cell shapes and density patterns in context, the segmentation results are put together in a single large image. The image in Figure 1.2 is rendered in accordance with design decisions made in iterative prototyping between

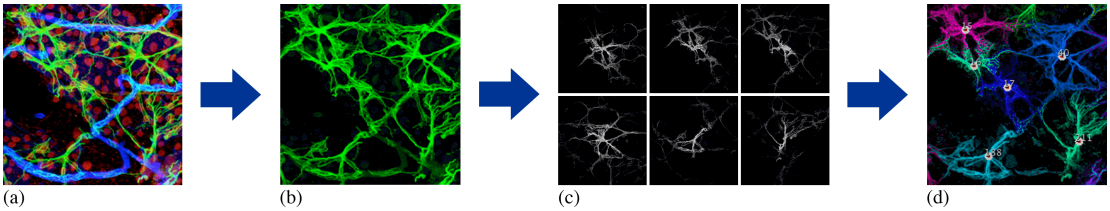
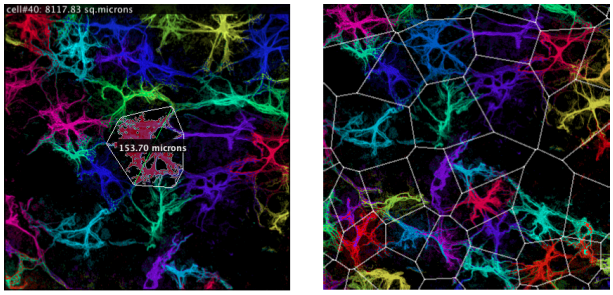


Figure 1.3: Data pipeline in the prototype version

computer scientists and biologists [8]. Referring to the data pipeline in Figure 1.3, we describe how we construct 1.3(d) from 1.3(c).

The combined image (Figure 1.3(d)) has the same dimension as the original image (Figure 1.3(b)). Each pixel, instead of containing a single color denotation, contains a linked list to pairs of (id, p) where p is the probability of cell_{id} occupying that pixel.

In rendering the final image as in Figure 1.3(d) (inset of Figure 1.2), the colors (hues) of different cells are initially randomly assigned. Since each pixel can be assigned to different cells, the color of the pixel is chosen to be the hue value of the cell whose probability is the greatest (called the winning cell). The brightness of the color is then proportional to such score. These design decisions conform with the principle that hue is suitable for distinguishing category while brightness is suitable for representing continuous quantity [25].



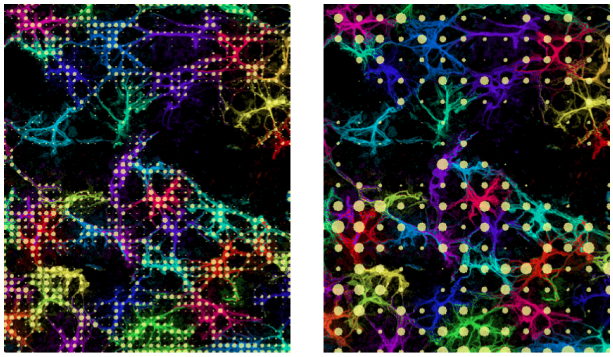
(a) detail of cell 40 (b) Voronoi diagram

Figure 1.4: Other visualization features

1.2.2 Interactions

A variety of simple interaction techniques help the user explore the data both in detail and in context. Layers of information can be turned on by control panel switches, keyboard shortcuts, or mouse gestures.

In the default mode, the user sees a simple view as in Figure 1.2, but the user can also bring up the cell center locations as shown in Figure 1.3(d). As the user hovers the mouse over the image, the active cell becomes highlighted. The pixel boundary of the segmented cell is highlighted and its extent outlined by drawing its convex hull. The cell area is calculated by counting the number of pixels belonging to the cell, then converted to μm^2 and shown in a box. In Figure 1.4(a), the user is viewing the detail of cell 40 (also the cell in the upper-right corner of the inset in Figure 1.2), as well as performing a distance measurement of the diameter.



(a) with diameter = 10px (b) with diameter = 30px

Figure 1.5: Large glyphs appear where segmentation is uncertain.

A selection of multiple cells can be toggled by double clicking at the cell centers. The user can also compare how the cell boundary is related to a Voronoi diagram of the cell centers in Figure 1.4(b). When the feature is turned on, as the mouse hovers, the centers of the cells to which this pixel belong are highlighted with lines linked to the cells and the probability from the segmentation algorithm annotated.

1.2.3 Uncertainty Visualization

A substantial part of our analysis depends on the segmentation results. It is important to communicate uncertainty to the user. Not only can the visualization of uncertainty contribute to the confidence in the results, it can also be used to guide the segmentation itself as discussed in [26] and [27]. Previous work on uncertainty visualization of segmentation results include an approach based on both computed uncertainty and user-annotated uncertainty [28]. In this section,

we focus on computed uncertainty. The user can already annotate the damage area, which poses uncertainty, as explained in Section 6.3.

The uncertainty of our data comes from the fact that we assign colors to pixels based on the result of a probabilistic segmentation method. Cells overlap more heavily in some areas than others; therefore, the confidence in our decision to assign colors to a region varies across the image. Since the validity of any scientific conclusion depends on the correct interpretation of this image, it is important that the amount of uncertainty be communicated to the user. We utilize the concept of entropy to quantify uncertainty. For each pixel, the entropy is defined as the random walk score of the winning cell divided by the sum of the random walk scores of all cells with respect to this pixel. The entropy is high when it is clear which cell occupies the pixel. To visualize, the image is divided into $k \times k$-pixel grids (k adjustable by the user), each cell displaying a glyph (a solid disc) whose size is inversely proportional to the average entropy. Figure 1.5 overlays the image with glyphs of varying sizes. Wherever a glyph is big, the uncertainty is high – there are contentions between nearby cells so the cell color assignments should not be trusted completely.

1.3 Improvements in the second generation

Continued use of the system and users' feedback revealed a critical need to create an updated system in which retinal datasets are viewed at full resolution (0.31 μm/pixel). Additionally, a need for interactive segmentation parameter choices and more comprehensive visual analysis tools was identified. The challenges for these improvements range from the size of the data to the speed of the algorithms involved.

To handle significantly larger images, we employed two techniques: image pyramid and multithread processing.

1.3.1 Image pyramid cache

A major hurdle in visualizing retinal data sets consists in the sheer image dimensions of the microscopy output. Resulting montages are as large as 300 megapixels. We designed and implemented an image pyramid system, as introduced in [29], to be flexible without any size constraints. We store our patches in separate small files organized in a directory structure according to scale factors. The system supports arbitrarily large images and is tolerant against individual file corruptions, using a simple redundancy scheme.

To support fast switching among different pre-rendered segmentation results, the system caches recently-viewed patches in memory until they are 30 seconds [11] older than the most recently-loaded patch. For smooth navigation, low resolution patches are always ready for display while the high resolution version is being loaded.

The image pyramid, technically called `LargeMap` in the system, for the original image is generated automatically when imported [12]. Approximately seven levels of zoom factors, depending on the size of the original image, are each kept in a sub-directory [13]. Files in the 100% zoom factor directory are potentially useful for a quick test of image processing algorithms on small patches. The patches are at most 300×300 pixels in dimension in the current implementation, although the sizes are configurable in the `LargeMap` package.

The rendered segmentation visualizations according to different parameters (cut off threshold and gradual coloring option) are also kept as `LargeMaps`.[14] Because of image pyramid, it is possible to instantly switch back and forth between multiple views of the retina as shown in Figure 1.6. Important special effects such as channel splitting (showing only the GFAP or the blood vessel channel in the original image as shown in Figure 1.8(c, d)) or color mapping (showing the

[11] value empirically determined in viewing experiments although this is configurable.
[12] It is kept at the directory `data/LargeMap/origPic`
[13] under the `patches-by-zoom-factors` directory
[14] They are located at `data/preRendered` directory.

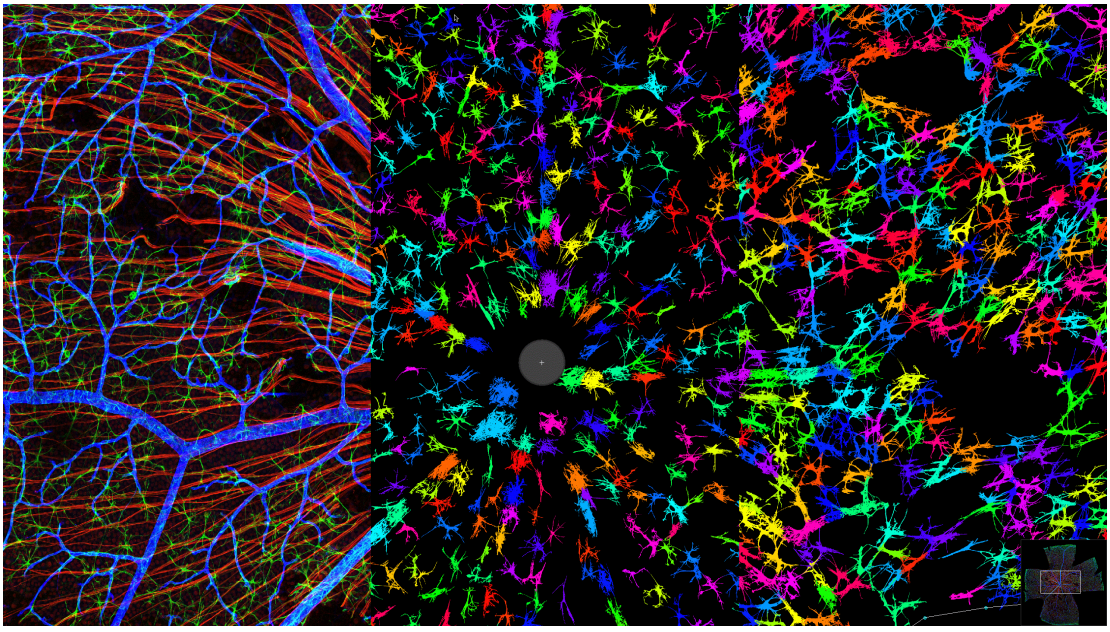


Figure 1.6: A composite image showing three views of a retina

combined segmentation result as translucent or on white instead of on black as shown in Figure 1.9(c, d, e)) can be implemented by applying the filters on the small patches at appropriate zoom levels instead of the entire image.

1.3.2 Multithread processing

To improve system performance from our previous version, we parallelized our computation using Java thread pool [30]. In most cases, a large amount of computation occurred in independent repetitive tasks across multiple cells. Typically, our system spawns a pool of N threads where $N = 1.5 \times$ (# of CPU cores). A number of threads higher than the number of cores produces a faster result be-

cause threads can become intermittently idle while waiting for resource, yielding execution time to more active tasks. Although more analytic approaches in determining thread pool sizes exist [31], it has been informally observed that our heuristic approach has yielded a CPU utilization near 100% while not exceeding physical memory capacity. [15]

This change greatly improved the performance in the following areas: precomputed segmentation of cells in an entire data set (4,500 cells) is now completed in 2 hours compared to 2-3 days in the previous version. Pre-rendering of a full-resolution whole retina using new segmentation parameters and constructing its image pyramid is accomplished in approximately 4 minutes. Performing a basic region-based analysis requires less than 10 seconds when analyzing up to 50,000 regions[16].

1.4 System components

The system diagram is shown in Figure 1.7. The central component is the graphic user interface (GUI) consisting of three seamlessly integrated components: the data importer/editor, the visualization viewer, and the region-based analyzer.

[15] Our system was developed on an Apple iMac with 16 GB of RAM and 4 CPU cores, and tested on an IBM PC running Windows with 42 GB of RAM and 24 CPU cores.

[16] A typical random-region analysis requires at most 10,000 regions.

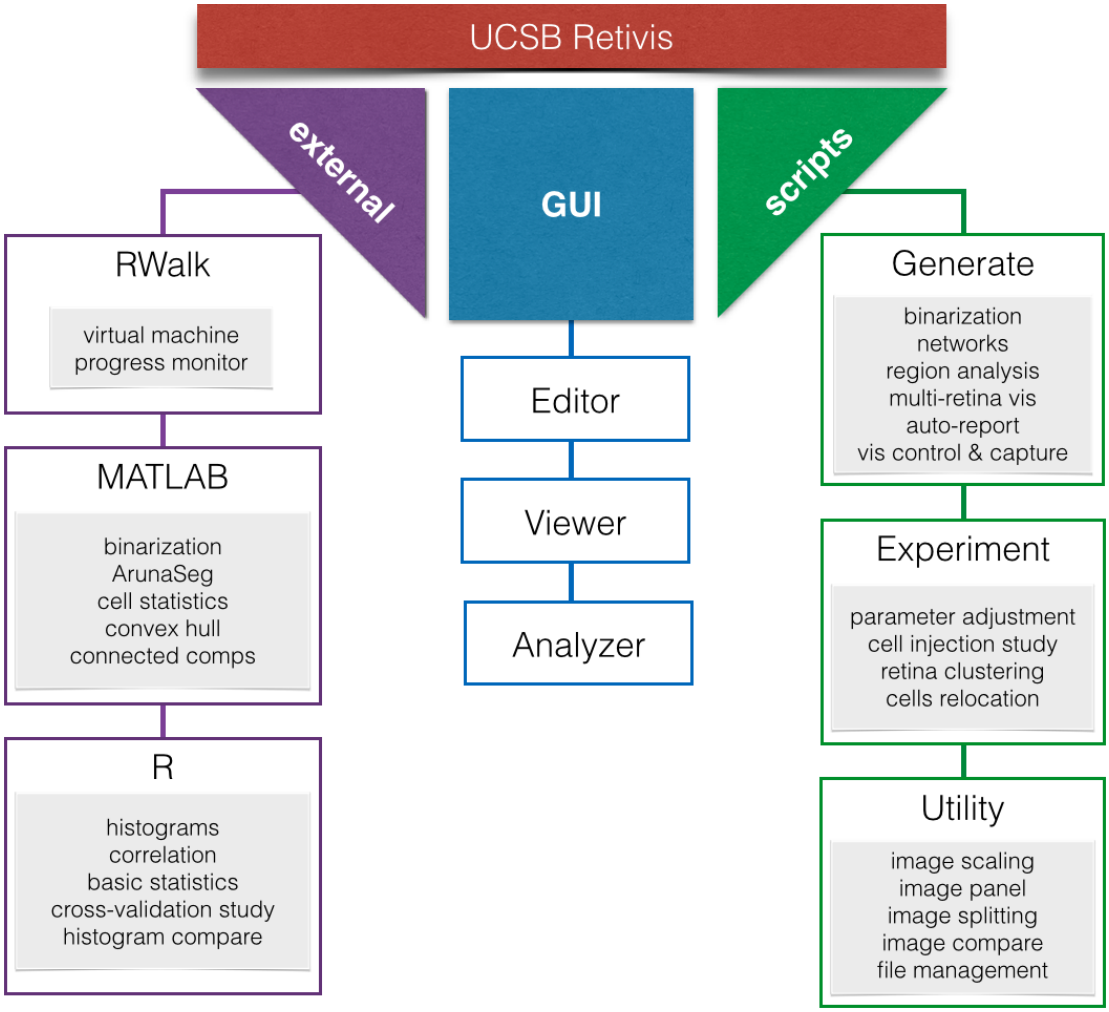


Figure 1.7: UCSB Retivis system diagram

The GUI is supported by integrated external tools. The random walk program runs on a Linux virtual machine and communicates with `Retivis` via a shared disk space[17] and is monitored continuously in the GUI. The MATLAB technical computing software is used for calculating adaptive thresholding binarization, alternative segmentation, binary image region property, connected component statistics

[17]the `segment-me` directory under each cell's directory at `rwalk/cell#`

and charts, etc. It is integrated with the program via the `MatlabControl` Java library [32] which allows simultaneous connections to multiple running instances of MATLAB. This greatly facilitates parallel processing of per-cell computations such as segmentation and cell statistics.[18] Finally the `R` Statistics Software is used for calculating histograms, basic statistics, and advanced histogram comparisons. It is used especially extensively to run the single cell injection cross validation study (Section 2.7). The `Rserve` package [33] and the `REngine` Java library [34] allows `R` commands to be issued from `Retivis`.

To reduce user's workload and to ensure consistency in treatments and analyses of multiple retinas, a scripting interface is provided. In the scripting mode, the user treats `Retivis` as a scripting language by calling the appropriate macro name and parameters from the command line [19]. The script controls the GUI, which in turns control the backend, to generate derived information and visualization such as cell binarization, overlap networks, region-based analysis, multiple-retina visualization, automated web-based insight reports, and miscellaneous screen captures. The scripts are the primary tool for running all experiments such as the parameter adjustments in the single cell injection study, retina clustering, and cells relocation[20]. The scripting interface is also used as generic utilities for scal-

[18] The parallel connections are managed by the custom `MatlabPool` library under the `mutils` package.

[19] by calling `./retivis macro=macroName ret=retName <params>`, for example

[20] a pilot experiment in which we place cells either at random locations or in a regular grid to observe the difference in distributions.

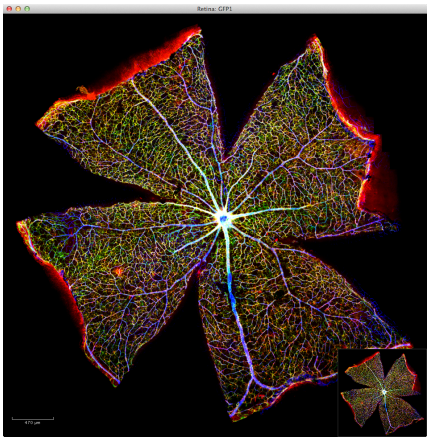
ing, combining, and comparing images or other files. For a large and complex set of data, a custom utility that understands the basic file structure has proven crucial in data management, gaining quick insights of the overview, and preparing for the presentation of the data. While the GUI is used for an interactive survey of data and the screening of relevant variables, the scripting interface is used to create all the final visualizations presented in Appendix A.

All the information from the user's input, the external tools, and automatic processes are combined by the central GUI component and presented to the user as layers of information.

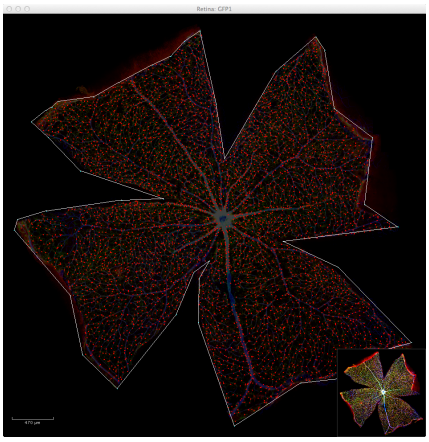
1.5 Information layers

1.5.1 Basic information and interactions

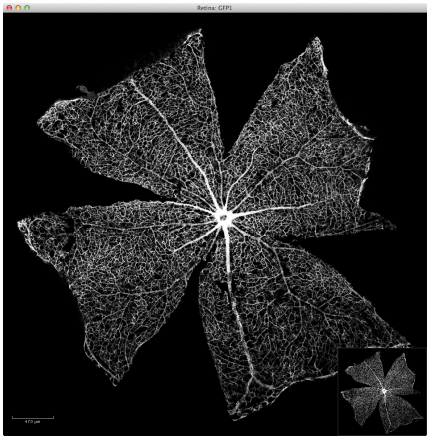
Our visualization system, called `UCSB Retivis`, is capable of displaying basic information such as the original image, the cell centers, the optic nerve head, the retina border, the Voronoi diagram, and the segmentation result. The information is displayed in layers and each layer can be turned on or off in the control panel GUI. For many layers, basic attributes such as the color, the background transparency, and the level of details can be configured. Most information is al-



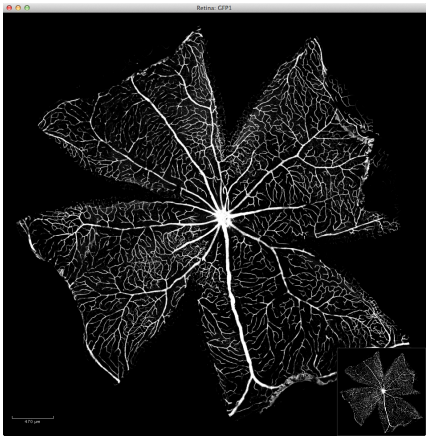
(a) Original image



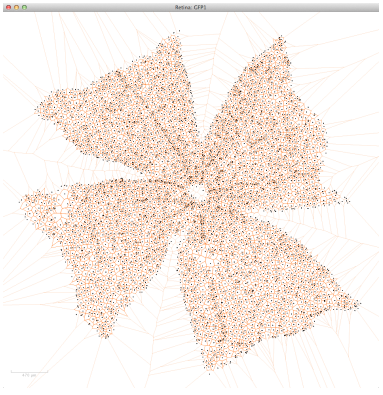
(b) cell centers, border, ONH



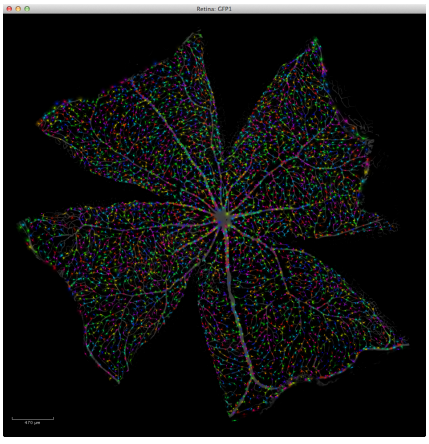
(c) GFAP (astrocytes)



(d) blood vessels

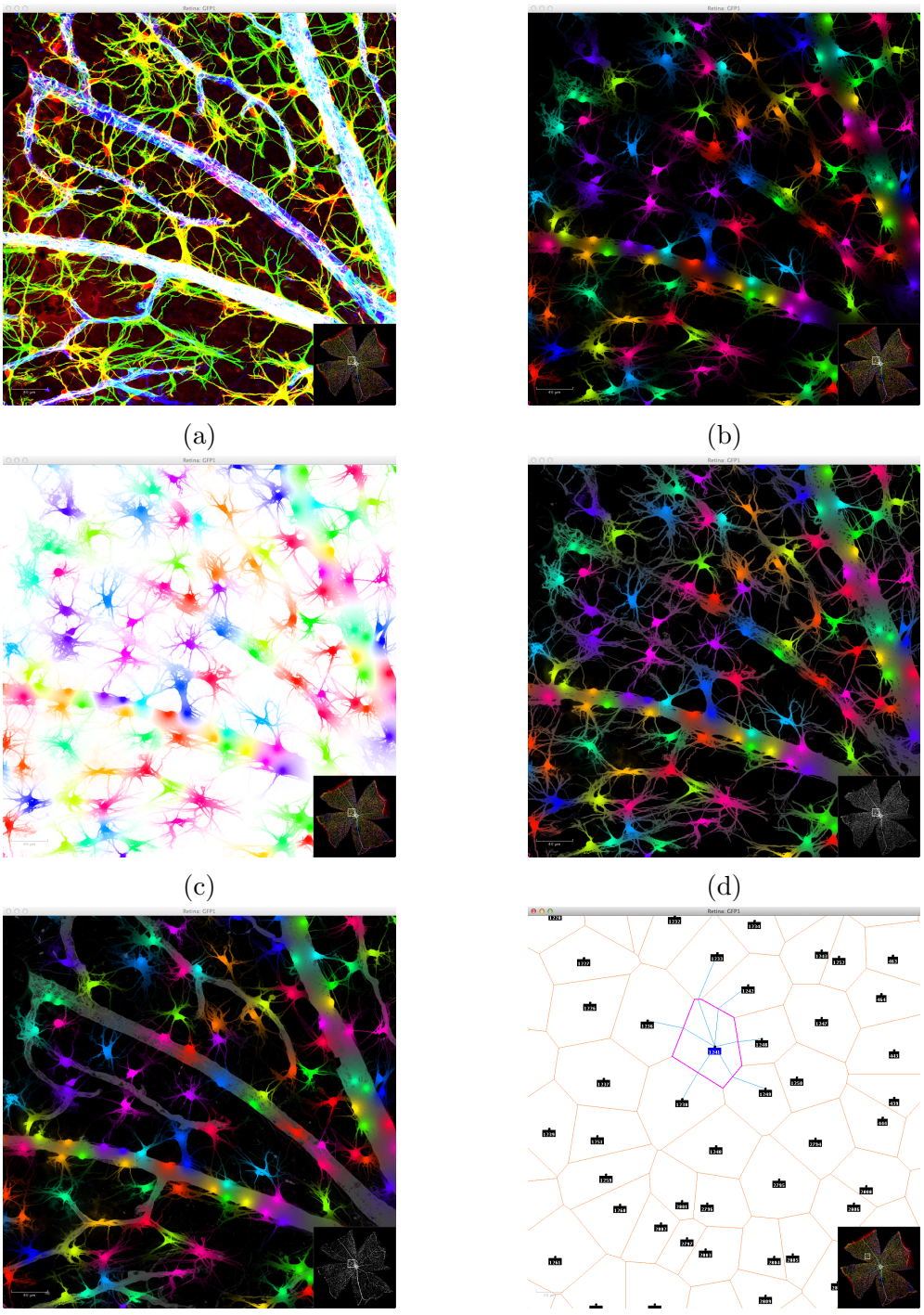


(e) Voronoi diagram



(f) segmentation result

Figure 1.8: Basic information layers (full retina)



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

Figure 1.9: Basic information layers (zoomed to 100%)

ready stored or precomputed on the disk, but some information can be rendered on demand using the multi-resolution image pyramid patches.

Figure 1.8 and Figure 1.9 show some examples of the information layers in full retina overview mode and in the 100% zoom mode, respectively. The users can double click at any part of the image to zoom in to 100% and center the view at the cursor. The mouse wheel can be used to zoom in or out. Panning is done by dragging while pressing the left mouse button. From any zoom level, the user can press `[R]` on the keyboard to reset the view back to the overview mode where the zoom level is automatically set such that the entire retina is displayed in the window.

At the start, the user is presented with the original image in Figure 1.8(a), whose 100% zoom level is shown in Figure 1.9(a). This original image can be dimmed or turned off by adjusting a slider in the control panel or repeatedly pressing `[O]`. Figure 1.8(b) shows an example when the original image is dimmed to 25% and two other information layers are turned on: the cell centers (in red) and the retina border (in white with draggable cyan anchor points). The cell center can be added or deleted using a context menu by right clicking on the location of the center. The retina border can be edited by checking the "allow edit" option under

the `analysis - define regions` tab, after which the instruction will appear on screen.[21]

Two additional options are provided for controlling the active channel of the original image. The user may wish to see only the astrocytes in GFAP (green) channel by checking "only astrocytes" as shown in Figure 1.8(c), or the blood vessels (blue) channel by checking "only blood" as shown in Figure 1.8(d). The channel extraction is done on-the-fly at the time of rendering using the patches from the image pyramid.

When the original image layer is turned off completely, the background can be set to a specific color such as white as shown in Figure 1.8(e). Here, the cell centers are also set to a different color (black) and the Vorornoi diagram is shown in orange. The Voronoi diagram is computed with the JTS Topology Suite [35] as needed[22]. Figure 1.9(f) shows the Voronoi diagram at 100% when the "extra" option is checked and the user clicks on a cell. The links are drawn to the Voronoi neighbors through the Voronoi edges that they share. This visualization ensures that all statistics based on Voronoi cell and neighbors are computed correctly.

The segmentation result, rendered with a cut-off threshold of 0.0005, is shown in Figure 1.8(f) and Figure 1.9(b). The segmentation can be rendered on top

[21]The boundary polygon and its related information (e.g. size, perimeter) are kept under `data/retina-boundary`. The entire edit history is also kept.

[22]It is kept at `data/voronoi-diagram`

of the original image layer to provide context as shown in Figure 1.9(d) and (e) when the GFAP and the blood vessel channels, respectively, are also displayed as a gray background. Figure 1.9(d) has helped the biologist verify the accuracy and completeness of cell center markings.[23] Figure 1.9(e) is useful in surveying the relationship between blood vessels and astrocytes.

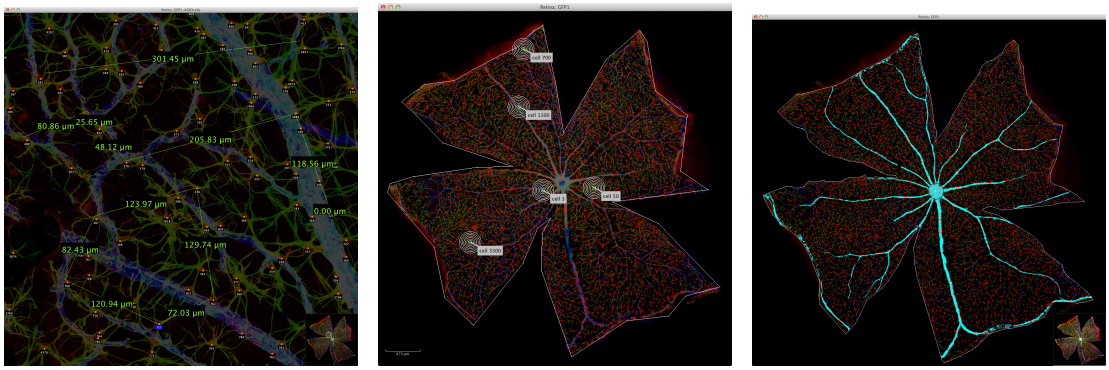
1.5.2 Annotations

The user can toggle the scale bar on or off by opening the tab μ and selecting "show scale reference." The size of the scale bar adjusts automatically based on zoom level but can be fixed at a certain physical length. On this tab, the `line`, `polyline`, or `polygon` measurement tools can be selected. The user can put virtual measurement tapes on the image and they will stay on as annotations, as shown in Figure 1.10(a), until the end of the current session.

When there is a question regarding a specific cell, the user can quickly jump to any cell in the retina by selecting "Go to cell ..." in the context menu. However, there has been a situation where multiple cells need to be located and their relative locations need to be discovered quickly. To select and highlight multiple cells, a list of cell numbers, separated by space, can be provided by the user [24]. Figure

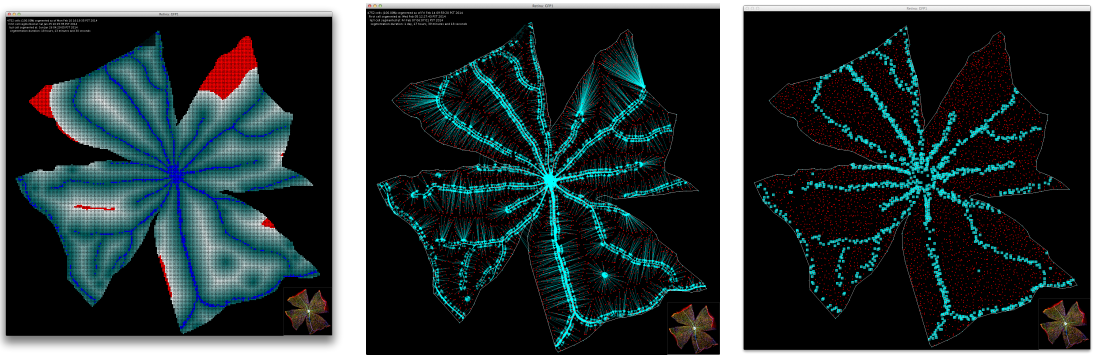
[23] approximately 100 additional cells per retina have been found after the fact because they appear to be in gray instead of another color.

[24] on the `multi-select` text box under the `etc` → `others` tab



(a) measurements (b) multiple selections (c) major blood vessels

Figure 1.10: Annotation layers



(a) the distance map (b) nearest vector (c) nearby cells

Figure 1.11: Major blood vessels

1.10(b) shows an output when "3 50 700 1100 3300" are provided. The five cells are clearly highlighted. This feature has become useful when it was detected that a number of cells did not have a corresponding Voronoi region. After a careful inspection, it was discovered that those cells were accidentally duplicated from another set of cells, and they were later deleted.

Major blood vessels masks are prepared by our collaborating biologist, Gabe Luna, using Adobe Photoshop. Starting from the blue channel which shows all blood vessels, the non-major vessels and noises were manually deleted. The resulting image is binarized and imported into the program. It is displayed in Figure 1.10(c). From this binary mask, a distance map is calculated for every point inside the retina as shown in Figure 1.11(a). The distance map was calculated by running a Dijkstra's single-source shortest path algorithm [36] from every edge point of the major blood vessels. A more efficient $O(n)$ algorithm is presented in [37].

Based on the distance map, from every cell, a vector can be drawn toward the nearest major blood vessel as visualized in Figure 1.11(b). This vector view is useful in verifying the calculation and in detecting noises in the mask, which is found in the small island near the lower right hand corner – it has since been removed. Cells within the vicinity of a major blood vessel can be highlighted as shown in Figure 1.11(c).

(a) min score = 0 (b) min score = 3 (c) variable thickness

Figure 1.12: Astrocyte network visualization

1.5.3 Astrocyte network and other information

Multiple plausible approaches exist for the definition of astrocyte networks. Different options are considered and discussed in Chapter 3. Once a network has been generated, it is written to disk and can be loaded for visualization. The network shown in Figure 1.12(a) is calculated from the overlap of grayscale images of cells which are Voronoi neighbors. When a filter of a minimum score is imposed, the network becomes more sparse; e.g. edges in Figure 1.12(b) have a minimum connectivity score of 3. Another filter may be imposed to enforce a maximum connection length. However, filtering may not be needed if the scores are visually encoded into the thickness and opacity of the edges as shown in Figure 1.12(c). In this view, the low-score connections almost disappear. Also, it is apparent that strong overlaps usually appear along some blood vessels – specifically the veins. There is, however, a single connection of unusual strength near the top left corner

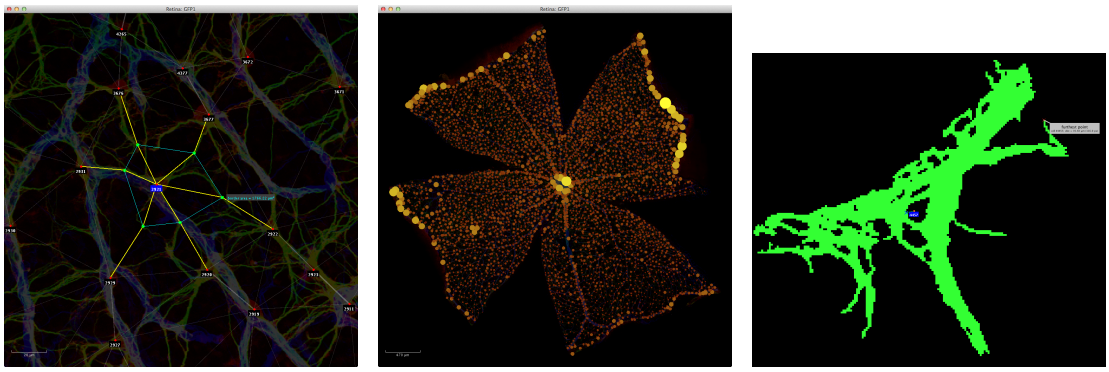
of the retina. Those two cell centers were later found to be located on the same cell. It was in fact a small error in the cell center location data that was detected with this visualization.

Another secondary information that can be visualized is a detailed view on local networks. Figure 1.13(a) shows a local network of cell 2921. Yellow line segments connect to its six neighbors, each connection bending and passing through a green point which is the point at which the overlap between two cells are the strongest. Cyan lines connect through the strongest connection points, forming polygon which can be considered the "domain" of the cell.

Finally, cell sizes can be visualized with circle glyphs as shown in 1.13(b); and the distance to the furthest geodesic point of each cell can be visualized as shown in 1.13(c). The cell size visualization suggests that the cells along the veins are the biggest. Cell sizes near the ONH and the boundary seem to be an anomaly. The furthest point visualization is useful in debugging the calculation and in explaining why small but complex cells may have a high furthest point distance.

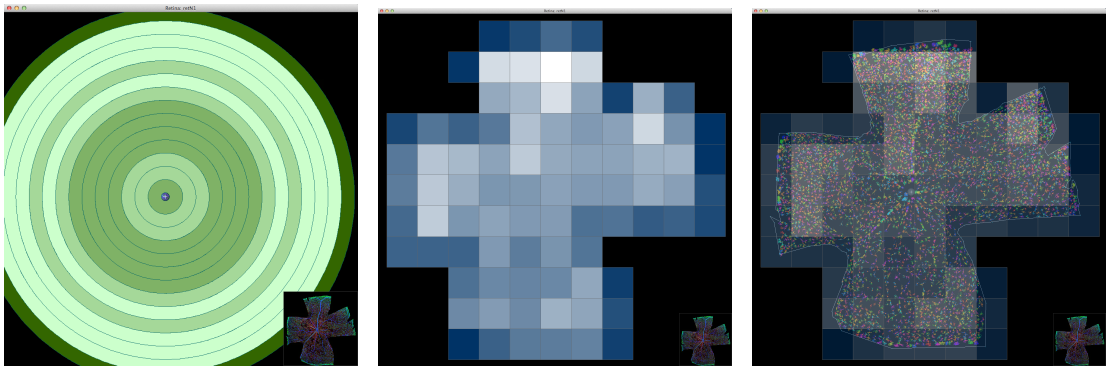
1.6 The region-based analysis tool

Users can divide areas into square regions (Figure 1.14(b, c)), concentric circles around the optic nerve head (Figure 1.14(a)) or by any other options as shown in



(a) local connectivity (b) cell sizes (c) furthest point

Figure 1.13: Other secondary information



(a) concentric map (b) opaque squares (c) translucent squares

Figure 1.14: Variations of region-based heat maps

in Figure 1.15. Sizes of the squares or the circles are under user control. After the regions are defined and placed, the relevant statistics (Section 1.6.2) for each region are computed. Some variables which are slow to compute can be excluded by the user[25].

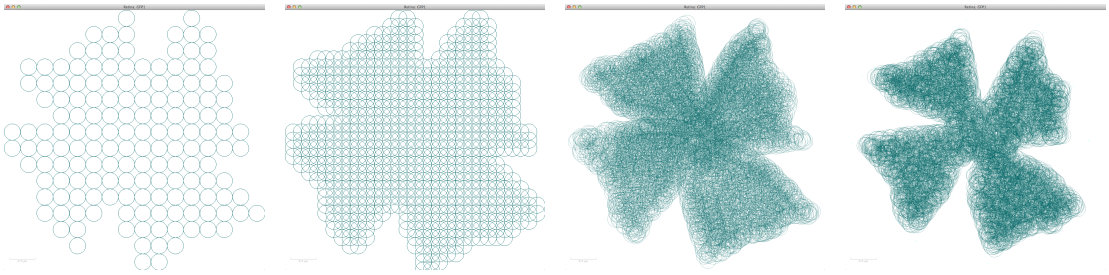
A central visualization option for analysis results consists in overlaid heat maps using custom color coding. Heat maps can be displayed in solid color or blended with user-selected transparency over the retina image to show context, as shown in Figure 1.14(b, c). The heat map color transfer function can be displayed as a continuous or customizable step function. To remove or highlight outliers, a certain top and bottom percentiles of the data, as directed by the user, may be excluded from the heat map color scale. The extreme values are shown as red or blue for the top and bottom ranks, respectively. For a quantitative analysis, the user can export all values to a spreadsheet in Microsoft Excel `.xls` or a generic `.csv` table format.[26]

1.6.1 Region shapes and placements

The shape of the regions can be a square, a circle, or a ring (donut). The ring shape can only be placed in concentric circles around the optic nerve head

[25]by unchecking the option `do it slowly` under the `analysis` tab

[26]The analysis output is located at `data/analysis-output`. The default filename is `output` but this can be specified from the GUI or the `outputName` parameter of the `analyze` script.



(a) on a grid (b) overlapping (c) on cells (d) random, inside

Figure 1.15: Region placement options

(ONH) as shown in Figure 1.14(a). Square and circle shapes can be placed on a grid (see Figure 1.15(a) where the circles are of diameter 300 μm) with an option to overlap the regions (see Figure 1.15(b) where the circles are overlapped with an offset of 150 μm). They can also be placed around the cell centers in which case the number of regions equal the number of cells (Figure 1.15(c)). Finally, to avoid any sampling biases toward densely populated areas, the regions can be placed randomly. In Figure 1.15(d), we randomly placed 5,000 circles around the retina. Every circle is guaranteed to intersect the body of the retina as defined by the hand-drawn boundary, but it may not be entirely inside the boundary. However, in the case of Figure 1.15(d), all the circles are forced to be entirely inside the boundary. The number of randomly placed regions is controlled by the user.

1.6.2 Computed statistics

Several statistics are calculated for each region. The names of the variables, the brief descriptions, and the units (if applicable) are included on a separate sheet in the output Excel file[27].

Nine variables that we use for the final analysis are listed and explained in Chapter 5 although many more variables have been computed.

The numerical statistics for each region are listed on Table 1.1. While a correlation between some of these variables may potentially be interesting, some variable pairs are obviously correlated by definition (e.g. `bin area` and `bin equivdiameter`). A few variables are omitted because they are for debugging purpose (e.g. whether the region id is odd or even) or almost duplicates of other variables (e.g. average amount of green, a duplicate of `sum chan0 g`). Variables whose names start win `bin` are a result of binarization. Most of them were calculated with the function `regionprops` in Matlab [38].

Table 1.2 lists variables that represent non-numerical entities such as a set of cells contained in the region or the X, Y coordinate of a certain point. For some analysis, it is important to know the location of each region; in which case the `region center x` and `y` variables are helpful. In other cases, it may only be

[27] under `data/analysis-output/output.xls`. Another file, `output_numberFields.csv`, is also created to support additional analytic tasks by an external statistics tool such as `R`

required to determine if the region is between the boundary and the optic nerve head (ONH). The boolean variables for such purpose are listed in Table 1.3.

Some variables do not describe any particular region. The variables in Table 1.4 describe either common characteristics of all the regions (shape and size) or the retina (name and number) under consideration.

Variable	**Description**
`region id`	id#
`bin area`	Binarized segmented cell area
`bin convex hull area`	Area of convex hull of binarized segmented cell
`bin convex hull holes`	Number of holes between the filled-in binarized image and the convex hull.
`bin eccentricity`	Binarized segmented cell Eccentricity
`bin equivdiameter`	Binarized segmented cell EquivDiameter
`bin eulernumber`	Binarized segmented cell EulerNumber
`bin extent`	Area of binarized segmented cell divided by area of its bounding box (0 to 1)
`bin filledarea`	Area of binarized segmented cell with all holes filled in
`bin fraction of convex hull`	Area of binarized segmented cell divided by area of its convex hull, AKA solidity (0 to 1)

`bin furthest geodesic point dist`	Distance along cell body (NOT Euclidean) from cell center to the furthest point of binarized segmented cell.
`bin majoraxislength`	Length of major axis of the ellipse over the binarized image (same second-moments).
`bin minoraxislength`	Length of minor axis of the ellipse over the binarized image (same second-moments).
`bin num holes`	Binarized segmented cell's number of holes (= 1-EulerNumber)
`bin orientation`	Binarized segmented cell Orientation (-90 to 90, with 0 being horizontal)
`bin perimeter`	Binarized segmented cell perimeter
`cellcenters count`	number of cell centers in this region
`cell near center`	Cell number of the cell nearest to the center of this region (-1 if none)
`density numcells per area`	density (number of cells per area of retina within region)
`dist 2 nearestneighbor of center cell`	distance to nearest neighbors of the cell at the center of region

`dist center 2 label`	distance from the center of optic nerve head to the ID label position (usually at the center)
`dist nearest major blood`	distance from center of region to the nearest major blood vessel
`dist of center cell to nearest major bv`	distance from the cell at the center of region to the nearest major blood vessels
`mindist from border`	minimum distance from retina's border
`mindist from onh`	distance from optic nerve head to the edge of the region
`region on retina area`	area of the retina within this region
`sum chan0 r`	sum of red channel values
`sum chan1 g`	sum of green channel values
`sum chan2 b`	sum of blue channel values
`voro area of center cell`	size of Voronoi region of the cell at the center of region
`weighted cell area of center cell`	weighted cell area of the cell at the center of region

Table 1.1: Numerical variables

Variable	**Description**
`cellcenters set`	set of cell centers in this region
`bin centroid x`	Binarized segmented cell Centroid in region of interest (X)
`bin centroid y`	Binarized segmented cell Centroid in region of interest (Y)
`region center x`	center of the bounding box of the region - X
`region center y`	center of the bounding box of the region - Y

Table 1.2: Non-numerical variables

Variable	**Description**
`center inside boundary`	Is the center of region inside the retina boundary? (0 or 1)
`center is between onh and boundary`	Is the center of region outside of ONH and inside the retina boundary? (0 or 1)
`center outside onh`	Is the center of region outside the optic nerve head? (0 or 1)
`totally inside boundary`	Does this region reside completely inside the retina boundary? (0 or 1)
`totally outside onh`	Does this region reside completely outside the optic nerve head? (0 or 1)

Table 1.3: Boolean variables

Variable	**Description**
`region class`	Name of the Java class representing this region.
`region diam`	Diameter of the circle or width of the square.
`region area`	area
`retina name`	Name of the retina (usu. the folder's name, e.g. GFP9)
`retina number`	The numeric part of the retina's name (e.g. GFP9 $\rightarrow$ 9)

Table 1.4: Variables not specific to a region

Part II

Data Preparation and Processing

Chapter 2

Single Cell Injection Study

We demonstrated an approach to evaluating and optimizing the parameters for the random walk segmentation algorithm that is used for segmenting individual astrocytes. To accomplish this, we made use of the ground truth inferred from 54 individually injected cells that our biologist collaborators, Professor Steve Fisher and Gabe Luna, NRI, UCSB, have collected. Evaluation of segmentation algorithms in general have previously been discussed in the literature [39] and [40]. Some previous work has used hand-annotated ground truth to evaluate segmentation algorithms [41] [42] with a general-purpose ground truth database presented in [43]. Rather than using manually-segmented images as ground truth, our work utilizes images of Lucifer-Yellow dye to define the correct boundary of each cell's soma.

Three objectives of our study are: (i) to demonstrate that the random walk segmentation algorithm can be fine-tuned to produce reasonably good results when

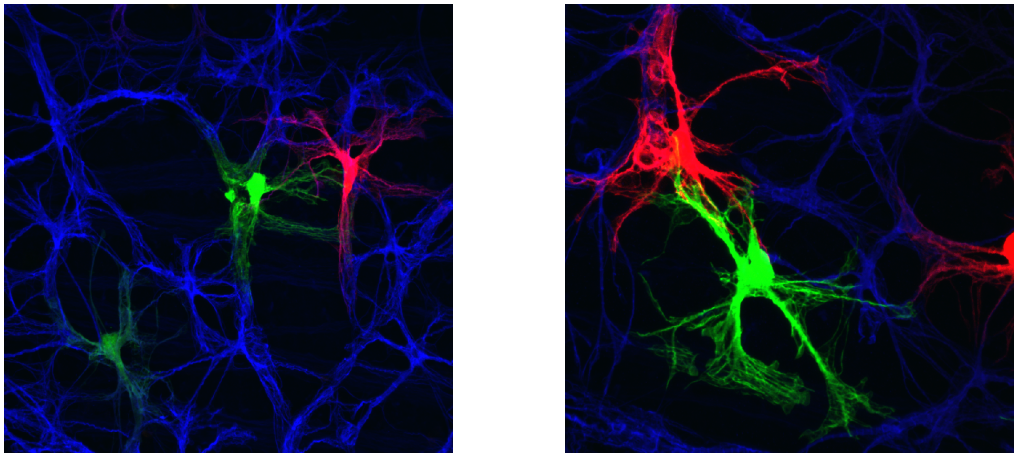


Figure 2.1: Example of multiple-cell injection

compared to the ground truth; (ii) to prove that parameter optimization based on one set of cells is generally applicable to another set of cells – this is in order to validate our approach in – (iii) to arrive at the optimum threshold parameter for the random walk segmentation algorithm that can be used to binarize cells for the purpose of building astrocyte networks or any further analyses.

2.1 Objectives

The three main objectives of the single cell injection study were set and implemented as follows:

1. For each individual cell, we determined its optimized parameter by using the ground truth. We then evaluated the segmentation result adjusted by each individually optimized parameter. (Note: parameters for different cells

may be different.)

Purpose: to test the limit of our segmentation method.

2. We randomly split the 54 cells into a training set (60%) and a test set (40%). We determined the best overall parameter from the training set, applied that parameter on the test set and evaluated the segmentation result. We compared the error rate on the test set against the error rate on the training set. We also compared the error rate on the test set against the best possible error rate that could have been achieved if we were to use the perfect parameters for the test set. This process was repeated several times.

   Purpose: To ascertain that our method of parameter optimization is generalizable. That is, one set of cells is a good representative of another set of cells.

3. We determined the best overall, single parameter for all the 54 cells. We evaluated the segmentation results adjusted by the overall best parameter.

   Purpose: To obtain the best parameter for use in the 8 full retina mosaics by treating the 54 injected cells as a training set. We intended to use this parameter to binarize the segmentation results, which is a necessary step for the calculations of important statistics such as cell perimeter, orientation, convex hull area, solidity, etc. Although – due to the lack of ground truth –

it was not possible to evaluate the validity of the binarizations in the entire mosaics, we can argue for its validity because the parameter we use has been proven to minimize the errors of the cells for which we have ground truth.

2.2 The individually injected cells

Initially, we have obtained 64 images of individually injected cells. They cells were assigned serial numbers from 1 to 64. Fifty-four cells remained after an initial image quality screening. The images of those cells are shown on Figure 2.2. Note that although the numbers run from 1 to 64, some numbers are missing. The physical dimension (width) of each square is 90.43 μm. Some clippings have occurred in order to show the body of the cells more clearly.

2.3 Definitions and abbreviation

The following terms are defined for use throughout Chapter 2.

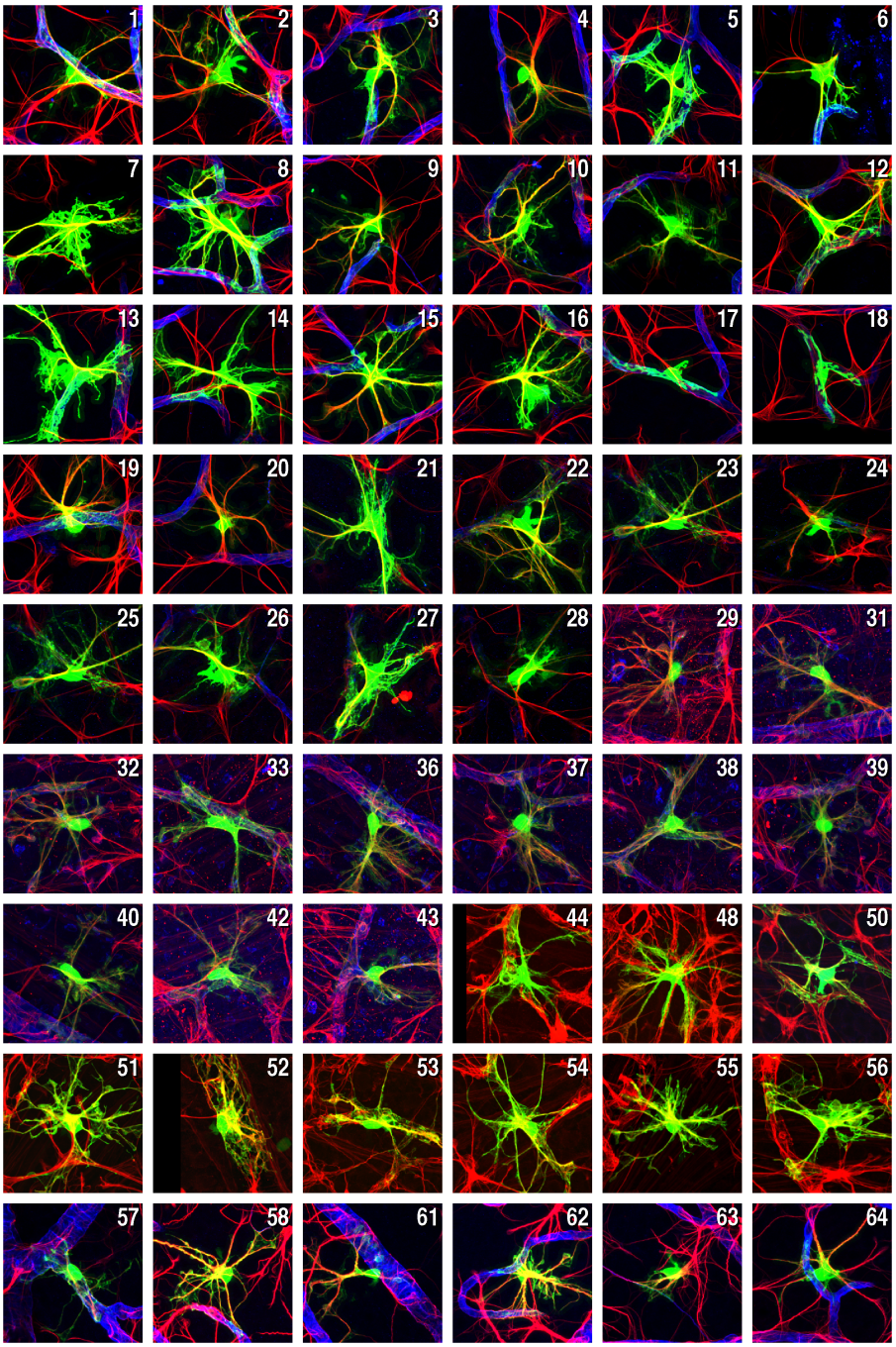
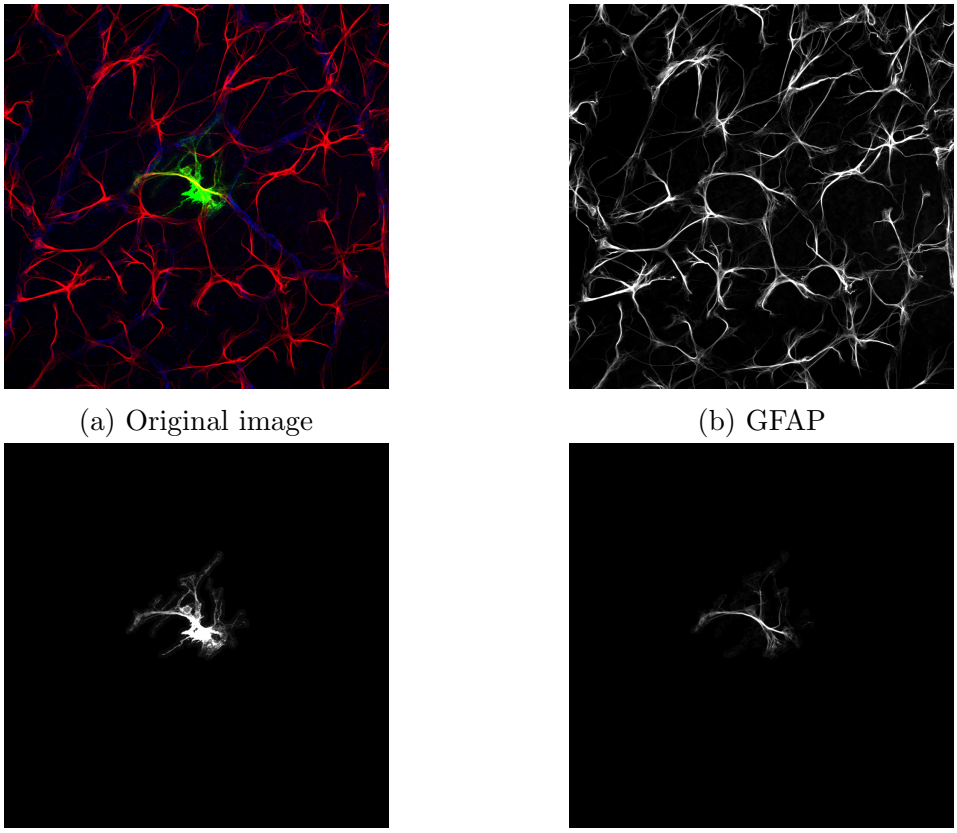


Figure 2.2: Original images of individually injected cells.
The physical dimension of each square is 90.43 μm.

Term	Definition
GFAP	Glial Fibrillary Acidic Protein – the stain used for all astrocytes' cytoskeletons (shown in red on Figure 2.2)
LY	Lucifer Yellow – a dye injected into a cell's body (soma) to show its structure (shown in green on Figure 2.2)
GT	Ground Truth
SR	Segmentation Result (grayscale output of random walk)
AS	Adjusted Segmentation result, derived from SR
FP	False Positive
FN	False Negative
error	FP + FN
ACC	Accuracy
ROI	Region of Interest
score	segmentation score with respect to ROI (between 0 and 1)

2.4 Definition of ground truth

The ground truth is defined as the intersection between Lucifer Yellow (LY) and GFAP. To explain the meaning of "intersection," Figure 2.3 shows an example



(a) Original image

(b) GFAP

(c) LY

(d) min(GFAP, LY)

Figure 2.3: Deriving ground truth for cell 26

cell in (a). The red channel is the GFAP as shown in (b). The green channel is the Lucifer Yellow as shown in (c).

While we may simply take the LY channel as the ground truth, it is not appropriate. When we optimize the parameter, we expect the adjusted result to resemble LY as much as possible. But since the segmentation algorithm only runs on the GFAP channel, as it would in real situations, it is unrealistic to expect LY as a result. Therefore, we decided to use a ground truth that is realistically achievable, by intersecting GFAP with LY.

To intersect the two channels, we use LY as a grayscale mask on GFAP. These three methods are functionally equivalent:

1. Using LY as a layer mask for GFAP in Adobe Photoshop, or

2. In Adobe Photoshop, put GFAP as the bottom layer. Put LY as a top layer. Change the blending mode of LY to "Darken", or

3. Treating each pixel's intensity as a fraction between 0 (black) and 1 (white); for each pixel, the minimum intensity value between the corresponding pixels in the GFAP and LY channels is chosen.

Therefore,

$$GT = min(GFAP, LY) \tag{2.1}$$

The ground truth for cell 26 is shown in Figure 2.3(d). The ground truth for all the 54 cells in our study are shown in Figure 2.4.

2.5 Calculating errors and scores

In this section, we discuss how we evaluated an adjustment of a segmentation result. We seek to define two terms: *error* and *score*. For a good segmentation result, the *error* should be low and the *score* should be high. The *score* should reflect a percentage between 0 and 100.

In the realm of binary classification, there is a well-defined concept of the following technical terms that can be used to evaluate a classifier: *precision, recall, accuracy, false positive,* and *false negative* (See, for example, [44], [45], [46] and [47].) However, the random walk segmentation algorithm that we use is not binary. Instead, it produces a probability map that can be viewed as a grayscale image which needs to be evaluated. In this section, we generalize the concept of the error measures from binary classification to our situation.

2.5.1 Error

False positive (FP) is defined to be the number of "items incorrectly labeled as belonging to the class"[45]. It is the number of pixels that the segmentation

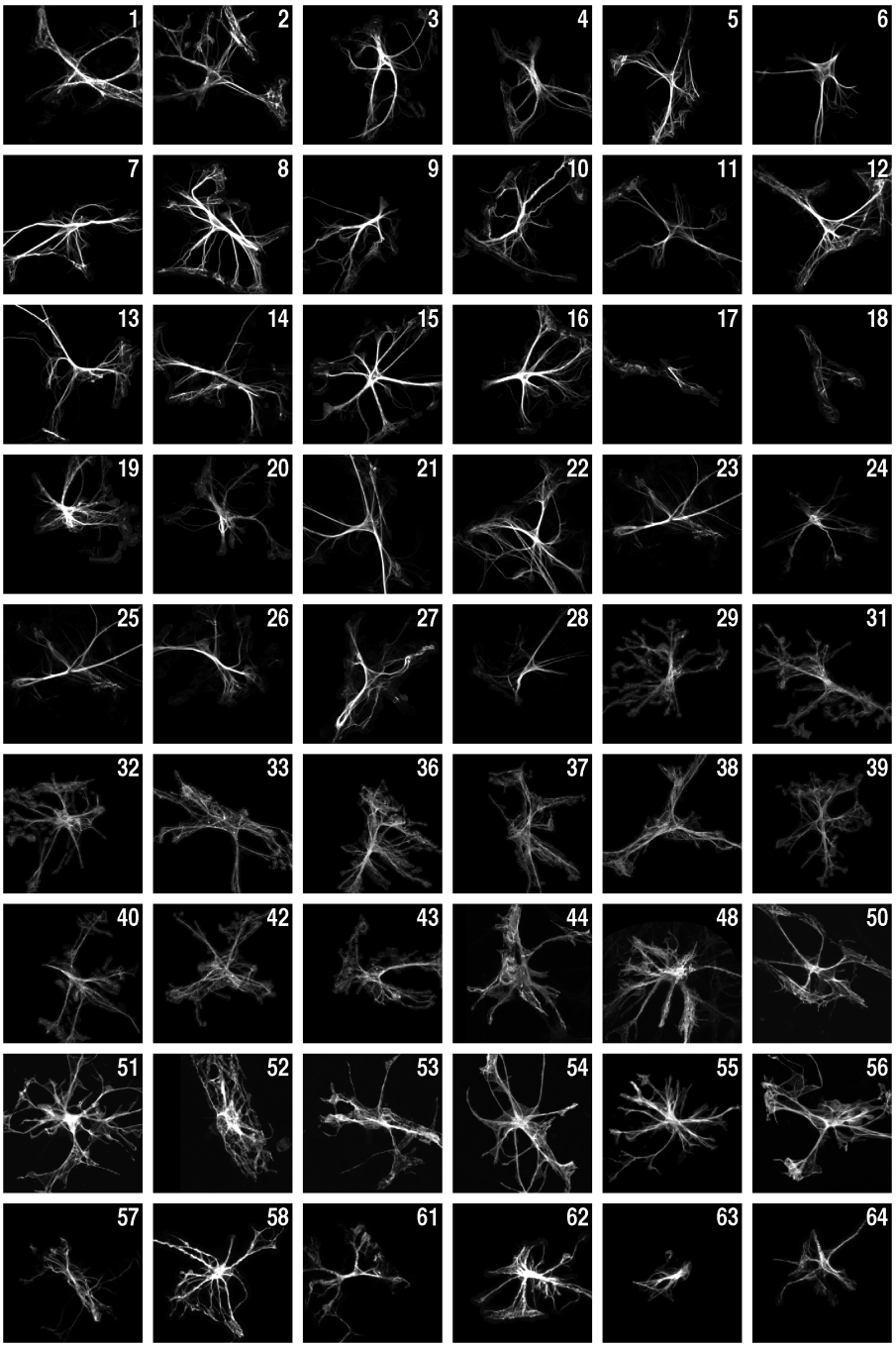


Figure 2.4: Ground truth derived from GFAP and Lucifer Yellow. The physical dimension of each square is 90.43 μm.

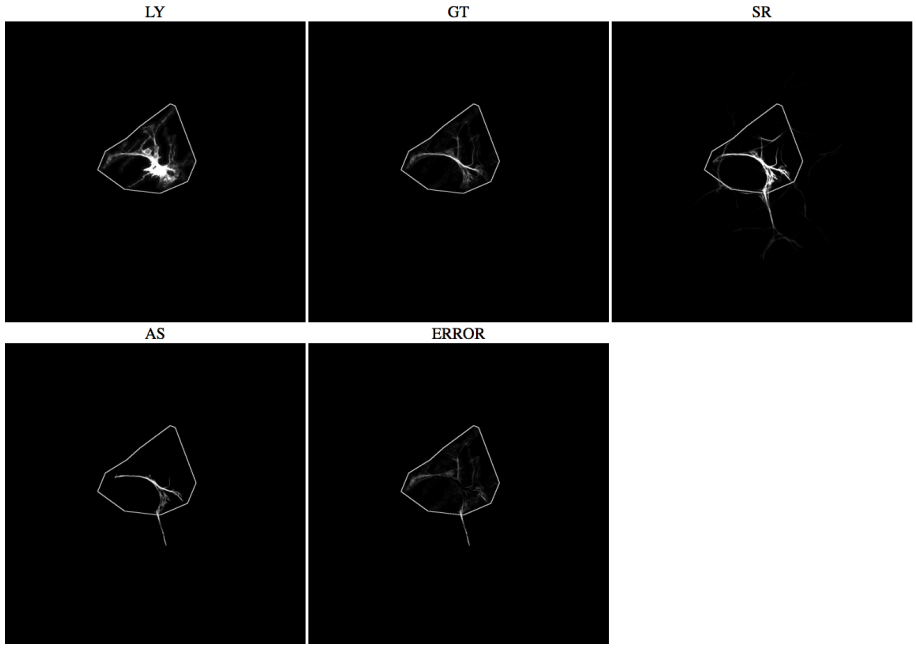


Figure 2.5: Calculating segmentation error and score for cell 26

result includes but that should not be included. For binary images, it is the number of pixels in AS but not in GT. In short, it is the number of white pixels in (AS - GT).[1] In this case, AS and GT are both grayscale. The same formula can be employed for the definition of false positives. Therefore, we define false positive to be:

$$FP = sum(AS - GT) \tag{2.2}$$

[1]Image subtraction is defined to be pixel-wise subtraction, where negative numbers are replaced with 0, equivalent to placing GT as a top layer in Photoshop and changing its blending mode to *Subtract*. Note that it is different than counting the number of white pixels in AS and subtracting from it the number of white pixels in GT.

where *sum*(image) is the sum of pixel values in that grayscale image. Note that each pixel is a fraction between 0 and 1 (0 = black, 1 = white). The sum of an image is analogous to the white pixel count of a binary image.

Similary, **false negative** (FN) is defined to be

$$FN = sum(GT - AS) \tag{2.3}$$

which is, conceptually, the number of pixels that the segmentation result misses.

We can define the total errors to be the sum of false positive and false negative.

$$total\_error = FP + FN \tag{2.4}$$

It can be observed that $total\_error = FP + FN = sum(AS - GT) + sum(GT - AS) = sum(|\text{pixelwise difference of AS and GT}|) =$ sum of differences of GT and AS. In binary images, it would represent the total number of error pixels.

Our definition of *error* (total error) can be used to rank the relative quality of two segmentation results. The result with a lower *error* is deemed better. To optimize a parameter for a single cell, we may vary that parameter across multiple values. For each parameter value, we produce the adjusted segmentation result (AS) and computed its *error* against GT. We pick the parameter value with the lowest associated *error*.

2.5.2 Score

The accuracy score is designed to be a percentage or a fraction between 0 and 1. The term *accuracy* is already well-defined for binary classification. That is,

$$ACC = \frac{TP + TN}{P + N} \tag{2.5}$$

where TP = true positive, TN = true negative, P = number of positive instances in ground truth, N = number of negative instances in ground truth [45]. Observe that P + N is the number of pixels in the ground truth = size(GT), while TP + TN is essentially the number of pixels (both white and black) that are classified correctly, which is size(GT) - sum(*error*). Therefore,

$$ACC = \frac{size(GT) - sum(error)}{size(GT)} \tag{2.6}$$

Conceptually, it is a probability that a correct answer will be given by the classifier if, for every pixel, it is asked to classify the pixel as white or black. It gives credits to both the positive and negative answers as long as the answers are correct.

We may expand that definition for grayscale images in Equation 2.6 by using exactly the same formula. However, for a set of segmentation results, we noticed that ACC almost always reached 0.99 or 1 even when there seemed to be a lot of errors. It is apparent that this measure is not informative and can be misleading

to report. One possible reason for this is because size(GT), the entire area of ground truth, in our case is relatively high, $1024 \times 1024 = 1,048,576$, compared to the sum(*error*).[2]

A simple solution is to slightly adjust the formula. Instead of using size(GT), we may use the size of a user-defined region of interest, or ROI.

$$score = \frac{area(ROI) - sum(error)}{area(ROI)} \tag{2.7}$$

A number of criteria were imposed on the characteristics of ROI, which should be relative to the size of the cell, not the size of the input image. It should not be just the tight area around the cell itself, but also its vicinity to capture any potential false positive that may be in the area. Since, given the same *error*, the size of ROI is negatively correlated with *score* by definition, the size of ROI must not be arbitrary to make the score reasonable and believable. Several ideas were discussed. The choices included:

1. a hand-drawn outline of the cell in the lucifer yellow (LY) channel.

2. a square one-third the size of the original image

3. a square whose size is 2X the sum(LY) of the biggest cell in the set.

[2]It is because our input image is much bigger than the actual domain for the cell. The intent of providing extra margins was to capture everything necessary, and to give room for the segmentation algorithm to roam.

4. a hand-drawn polygon outlining a reasonable domain of the cell (in LY channel).

The definition #4 was deemed the most appropriate because it guarantees to include all area of GT, and provides ample area in the vicinity to capture false positives. Hence, the polygons were drawn by our collaborator Gabe Luna to reflect a reasonable domain of the cell according to the view of a biologist. Figure 2.5 shows the polygon for cell 26 drawn over the LY, GT, SR, AS, and *error*. The polygons for all of the 54 cells are shown in Figure 2.6. The image is zoomed in and cropped at 2X to show details. The polygons for cell 9 and 54 extend outside their canvases in this figure but still stay inside the original image boundary. Although many polygons resemble a convex hull of LY, some are not necessarily convex.

The statistics for the areas of the ROI polygons, the ground truth, and Lucifer Yellow are shown in Table 2.1. The last fraction between the polygon and LY represents the relative size of the polygons compared to the injected dye.[3]

	min	max	mean
area(polygon)	8738 px	114732 px	17445.32 px
	(838.08 μm^2)	(11004.2 μm^2)	(5402.76 μm^2)

[3]For further analysis and reporting, the raw number of those values are available at `http://ilabsvn.cs.ucsb.edu/projects/retivis/cases10R/sums.csv` with the following columns: `lucifer.yellow.png` = LY, `gt_domain_mask.png` = the polygon, and `min(GFAP_LY).png` = the ground truth.

sum(LY)	2252.22 px	18633.4 px	8119.9 px
	(216.02 μm^2)	(1787.17 μm^2)	(778.8 μm^2)
sum(GT)	885.98 px	13587.61 px	4523.61 px
	(84.98 μm^2)	(1303.22 μm^2)	(433.87 μm^2)
$\frac{\text{area(polygon)}}{sum(LY)}$	3.07	14.06	7.74

Table 2.1: Ground-truth related statistics

In calculating the score using Equation 2.7, the polygon can be used as an ROI. In that case,

$$\begin{aligned} \text{score}_{\text{simple}} &= \frac{area(ROI) - \Sigma(error)}{area(ROI)} \\ &= \frac{area(polygon) - \Sigma(error_{inside} + error_{outside})}{area(polygon)} \\ &= \frac{area(polygon) - \Sigma(error_{inside}) - \Sigma(error_{outside})}{area(polygon)} \end{aligned} \tag{2.8}$$

When there is an error pixel (either false positive or false negative) outside of the polygon, the pixel is counted toward *error* to penalize the score. It can be argued that by accounting for the error pixels outside the polygon, the ROI has been expanded to include those pixels. Therefore, the definition of ROI was revised and clarified to include any error pixels outside of the polygon:

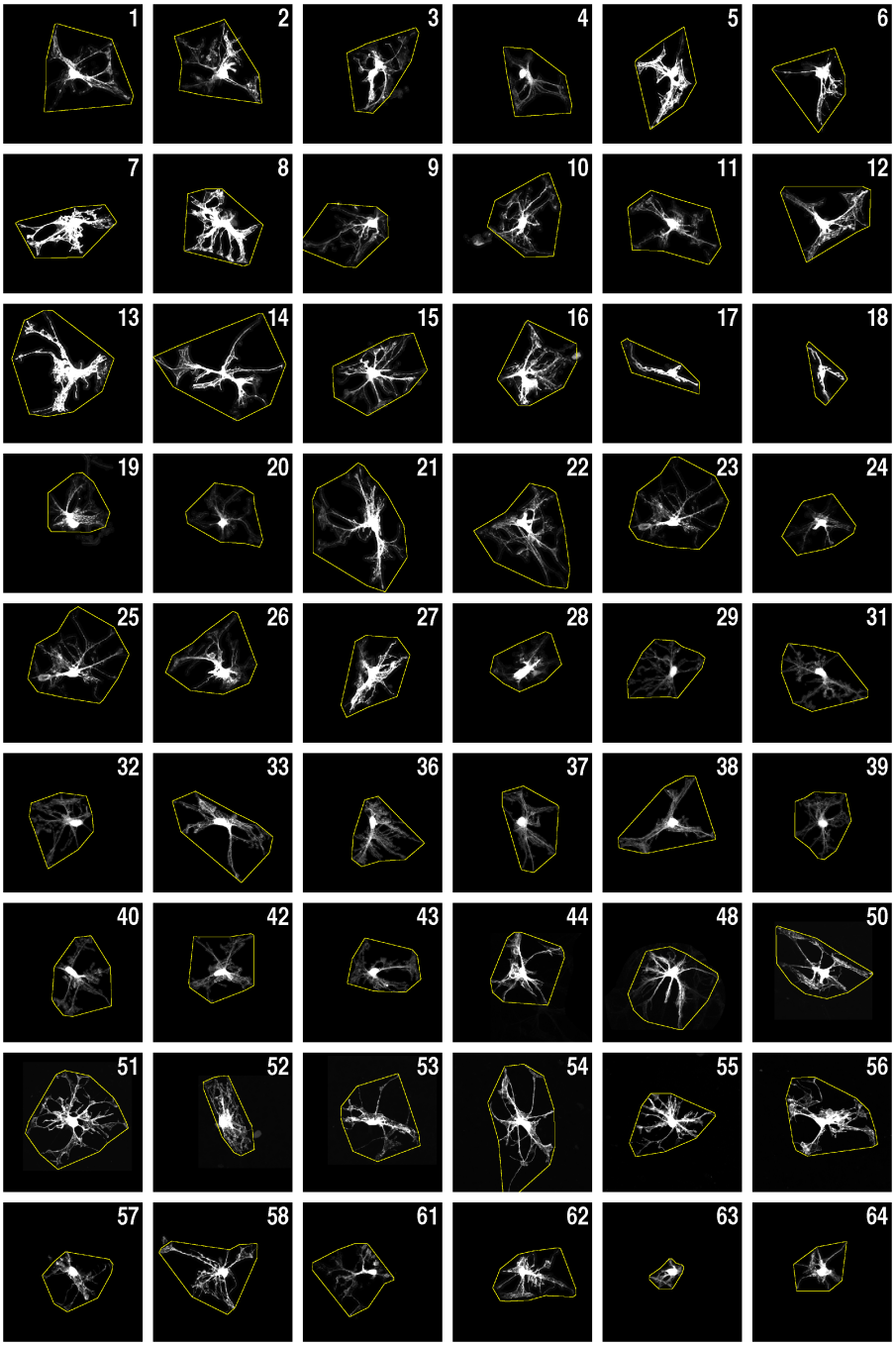


Figure 2.6: Polygons drawn by the biologist over Lucifer-Yellow (LY). The physical dimension (width) for each image is 158.56 μm.

$$ROI = polygon + \Sigma(error_{outside})$$

Thus,

$$\begin{aligned} score &= \frac{area(ROI) - \Sigma(error)}{area(ROI)} \\ &= \frac{area(polygon) + \Sigma(error_{outside}) - \Sigma(error_{inside} + error_{outside})}{area(polygon) + \Sigma(error_{outside})} \\ &= \frac{area(polygon) + \Sigma(error_{outside}) - \Sigma(error_{inside}) - \Sigma(error_{outside})}{area(polygon) + \Sigma(error_{outside})} \\ &= \frac{area(polygon) - \Sigma(error_{inside})}{area(polygon) + \Sigma(error_{outside})} \end{aligned}$$

Therefore,

$$score = \frac{area(polygon) - \Sigma(error_{inside})}{area(polygon) + \Sigma(error_{outside})} \tag{2.9}$$

In effect, compared to Equation 2.8, this new Equation 2.9 is changing the role of the term $error_{outside}$, from subtracting from the numerator to adding to the denominator. Both of these roles result in a lower number. Therefore, both the simple and standard definitions of *score* penalize the outside errors. We will use the standard definition as shown in Equation 2.9.

2.6 Parameters optimization for individual cells

2.6.1 Methodology

In this section, we seek to find a set of 54 best parameters for the 54 individually injected cells. For each cell_i, a number of parameter values are tested. The segmentation result is adjusted for each parameter value. The *error* of the adjusted segmentation result is calculated as described in Section 2.5.1. After all parameter values are tested, the parameter for which the *error* is minimum is picked as the optimum parameter for cell_i.

The parameter that is being optimized is called a **binary mask threshold**. Figure 2.7 shows an example for cell 26. The GFAP channel is an input to the segmentation program, producing a grayscale segmentation result (SR). The binarized SR is created by applying a threshold: for each pixel in SR, if the pixel value is below *the binary mask threshold* (in this case, 0.405), the output is 0; otherwise, the output is 1; hence, creating a binary image. This binary image is not the final output, however, because it does not resemble the ground truth image (which is grayscale). To create the final output, or "an adjusted output based on this threshold", or AS (adjusted segmentation result), we use the binarized SR as a mask, and apply that mask on the GFAP image. Since the mask is binary, the mask operation is a multiplication.

To find the optimum parameter for each individual cell, we vary the parameter from 0.0005 to 0.9 with an increment of 0.0005. There are 1,800 total parameter values examined. For each parameter value, we produced AS, computed its *error* (sum of difference with ground truth), and recorded it on a table. The row with the minimum sum of differences is considered the individually optimum threshold for the cell.[4]

2.6.2 Results

Figure 2.11 shows the result of parameter optimization for each individual cell. The gray/white background is GFAP. The green overlay is the individually optimized segmentation result. Each cell has a different optimized parameter, and those parameters are shown under *param* on the images. The accuracy scores are shown on Figure 2.12 where the colors show the following error components:

- green = true positive
- black = true negative
- red = false positive
- purple = false negative

[4]For cell 26, the table is kept at `http://ilabsvn.cs.ucsb.edu/projects/retivis/cases10R/cell026/info/maskingThresholdAnalysis.csv` (other cells have analogous URLs). The best parameter for cell 26 is 0.4555 with an error of 2170.49.

- yellow = partially false positive

  (area is part of GT but pixel is brighter in AS than in GT)

The values of optimized parameters are shown on Figure 2.8 and their *error*s shown on Figure 2.9. The ranges and means of the parameters and the *error*s are listed on Table 2.2.

	min	max	mean	median	SD
best parameter	0.035	0.9	0.4077	0.30975	0.318
error	949.0431	11540.13	3474.619	2838.537	2263.591

Table 2.2: Results of parameter optimization for individual cells

2.6.3 Relationship between ground truth size and error

Based on Figure 2.9, cell 54 has a relatively high *error*. A visual inspection of cells in Figure 2.2 suggests that cell 54 is relatively large. We hypothesize that the size of ground truth correlates with the individually optimized *error*. Figure 2.10 confirms this hypothesis. The *error*s are correlated with the size of ground truth with a correlation coefficient of 0.94. The red trend line in the graph is a regression line. A possible explanation is the following. For a bigger cell, there is more opportunity for error. The *error* calculation is the absolute number of FP + FN, not a fraction relative to the cell size. This does not mean that the

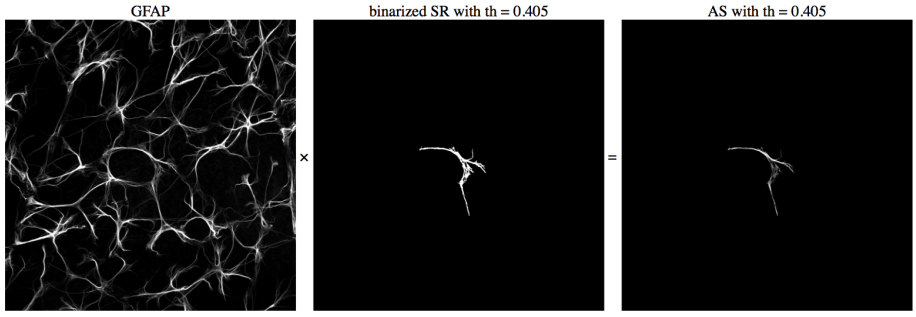


Figure 2.7: Parameter adjustment pipeline for cell 26

segmentation algorithm is less accurate, or having less *score*, for larger cells. In fact, relationship between the score and the cell size is relatively weak. The cell size (size of GT) is only slightly negatively correlated with the score, with a correlation coefficient of -0.58.

2.6.4 Relationship between polygon size and score

It is by definition that the score is influenced by the size of ROI (the hand-drawn polygon). The bigger the polygon, the more credits are given to the true negatives (the black & white area in Figure 2.12 where the segmentation classified correctly as not belonging to the cell), hence the higher the score. However, the correlation is not strong (only 0.29), which means that the score is not influenced primarily by the polygon size.

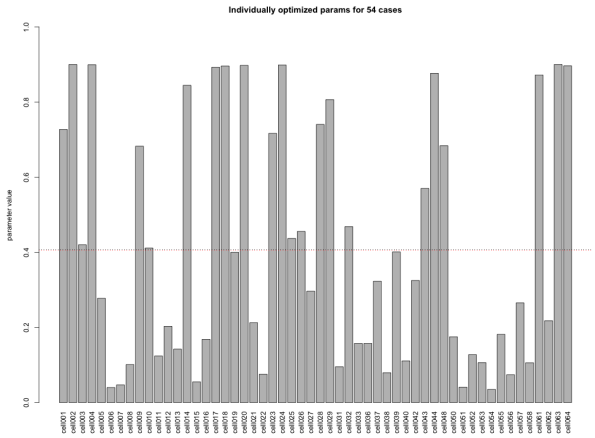


Figure 2.8: Values of individually optimized parameters

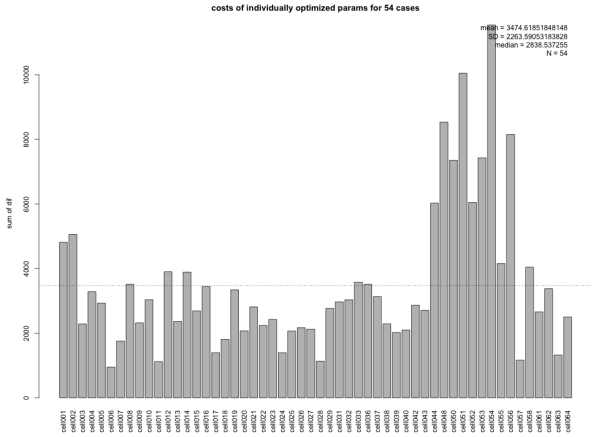


Figure 2.9: Costs (*errors*) of individually optimized parameters

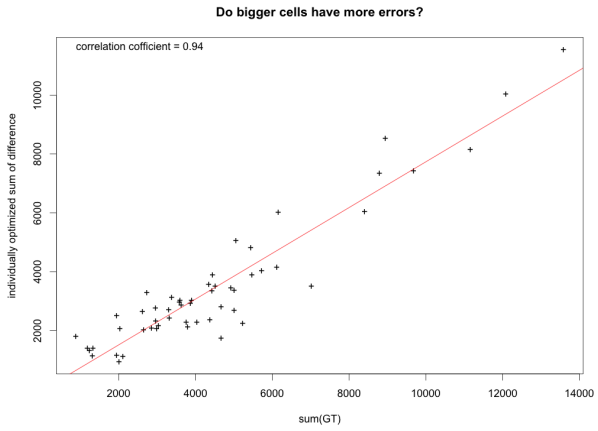


Figure 2.10: Relationship between *error* and size of ground truth

2.7 Cross validation experiment

2.7.1 Hypothesis

The purpose of the cross validation experiment is to demonstrate that our optimization methodology is generalizable. This depends heavily on the assumption that the 54 cells are a good representative of astrocytes in general. Specifically, we hope to argue that the optimized parameter based on the 54 cells obtained in Section 2.8 is a good parameter for other astrocytes. Because of the lack of ground truth for astrocytes in general, proving that statement directly is impossible. However, if we randomly split the 54 cells into a *training set* and a *test set*, it is possible to prove the following hypothesis:

> *"The overall best parameter for the training set is also a good parameter for the test set."*

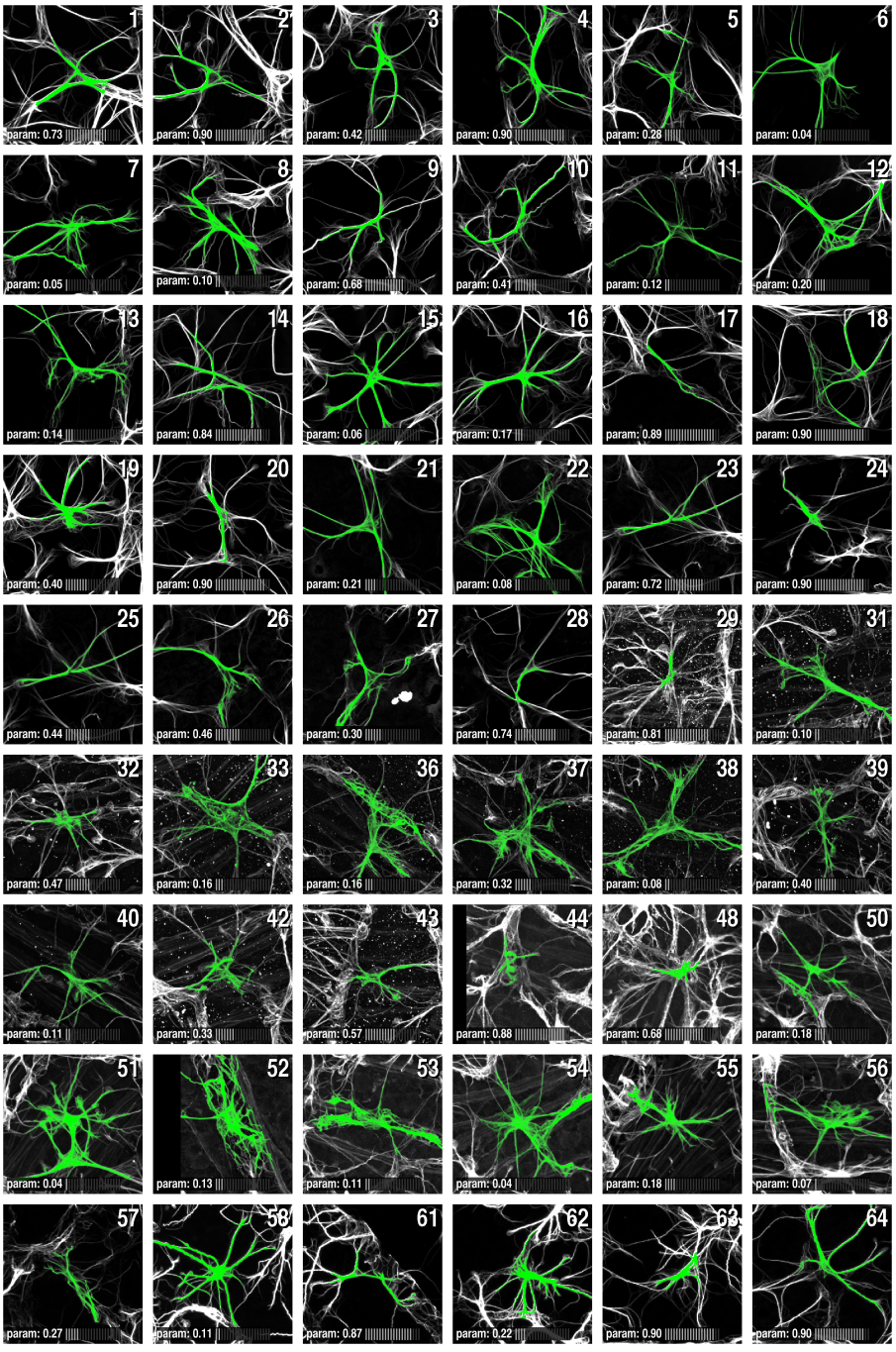


Figure 2.11: Results from individual cell optimization

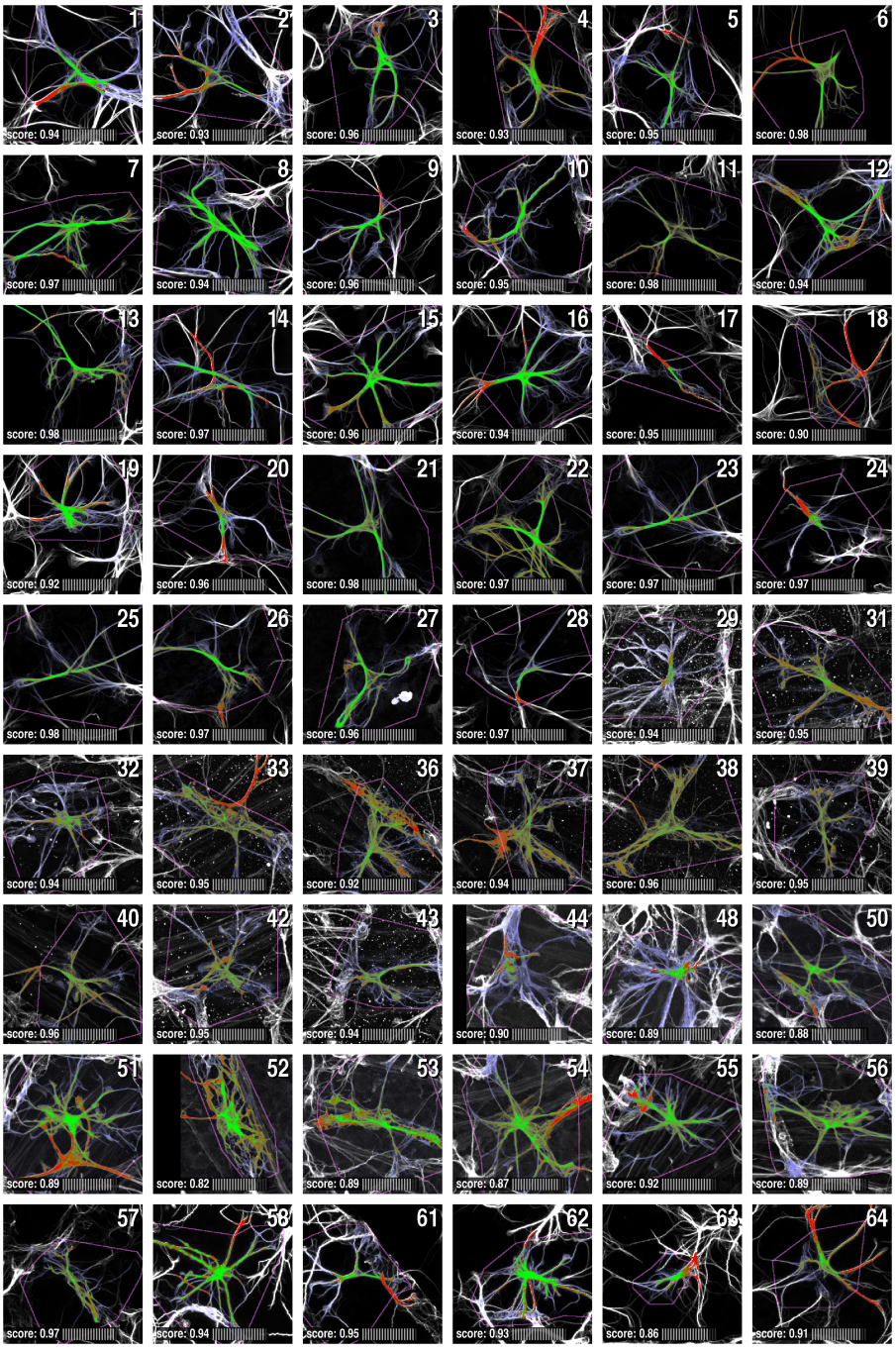


Figure 2.12: Results from individual cell optimization and their scores

2.7.2 Experimental setup

We repeat the experiment in multiple iterations. For each iteration, a random set of 32 cells (60%) is designated as a *training set*. The other 22 cells (40%) are assigned to a *test set*. The best parameter p for the *training set* is calculated as described in Section 2.8. The parameter p is good for the test set if it satisfies the following two conditions:

- **Condition #1:** The average error (error per image) in the test set is not much higher than the average error in the training set when using the same parameter p.
- **Condition #2:** Let q be the overall best parameter for the test set. The average error in the test set when using p is not much higher than the average error in the test set when using q.

For each iteration of the experiment, we computed all the relevant statistics pertaining the two conditions including:

1. **TestPerTrain**

   The running average of $\frac{\text{error of p in test set}}{\text{error of p in training set}}$. This is the comparison of the error in the test set to the error in the training set. Condition #1 dictates that this fraction should be as low as possible. (It can be less than 1 but the expectation is that it is above 1.)

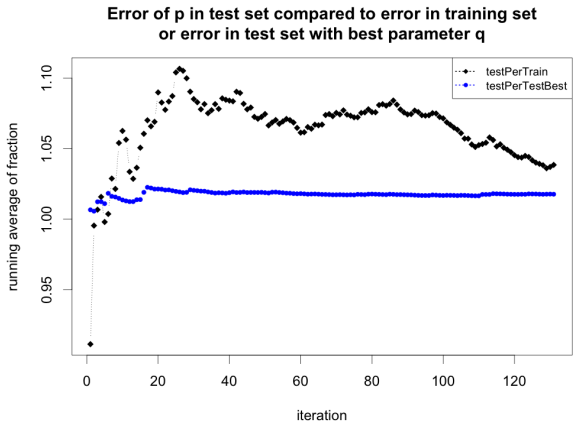


Figure 2.13: Relative *errors* in test set VS the training set

2. **TestPerTestBest**

   The running average of $\frac{\text{error of p in test set}}{\text{error of q in test set}}$. This is the comparison of the errors in the test set from using p (best parameter from the training set) VS using q (best parameter from the test set itself). Condition #2 dictates that this fraction should be as low as possible. (It will always be at least 1.)

2.7.3 Findings

After 131 iterations, we have found that *TestPerTestBest* converges, satisfying Condition #2, at 1.017663 while *TestPerTrain* still fluctuates, with the final average of 1.038451 and with the maximum running average of 1.106525 after the 26^{th} iteration. The running average of the two variables are shown in Figure 2.13 with the blue dots representing *TestPerTestBest*.

The fact that *TestPerTestBest* converges at a very low number suggests that the best parameter for the training set can be expected to perform very well on the test set: the expected error is only 1.77% higher than the best overall parameter for the test set itself.

The fact that *TestPerTrain* does not converge does not disprove our assumption. The value of this fraction depends on two factors: (i) how good the training set's best overall parameter p is on the test set (relevant factor); and (ii) how low the error can be in the test set compared to the training set (irrelevant factor). The fact that it still oscillates after 131 iterations suggest that factor (ii) plays an important role. Despite that, the final average is only 3.85% higher than the error in the training set. Condition #1 is also satisfied.

As will be shown in Section 2.8, our best overall parameter for the 54 cells is 0.405; it has an average error per image of 3903.419. Therefore, we may expect that if we use 0.405 on another set of image, the error will be within 3.85%, which is $1.038451 \times 3903.419 = 4053.509$.[5]

[5]Note that this is just an extension of our logical conclusion. It cannot be empirically verified unless we have ground truth for the other set of cells.

2.8 Parameter optimization for all cells

To the extent that our hypothesis in Section 2.7.1 has been proven, it is reasonable to assume that the parameter p that minimizes the total *error* in the set of 54 cells will also be a good parameter for other cells as well.

In this section, we describe an approach to determine the overall best parameter p that minimizes the total errors in the set of 54 cells.

The parameter p is varied over 1,800 values from 0.0005 to 0.9 with an increment of 0.0005. For each p, we adjust the segmentation results of all the 54 cells and calculate their *error* (sum of differences). The sum of all the *error*s (sum of sums of difference) is also calculated. Finally, the parameter p which produces the minimum sum of *error*s (called *total error*) is chosen. It is called the *overall best parameter*.

The overall best parameter is $p = 0.405$. Note that, even though the best parameter for each individual cell is likely different from 0.405, the overall best parameter – if all cells are to use the same parameter value – is 0.405. The errors range from 1129.31 to 12554.09 with a mean of 3903.419, median = 3166.867, and SD = 2448.499.

The graph in Figure 2.14 shows the *error*s of this parameter on each of the 54 cells. The size of each bar is proportional to the *error* for each cell. The

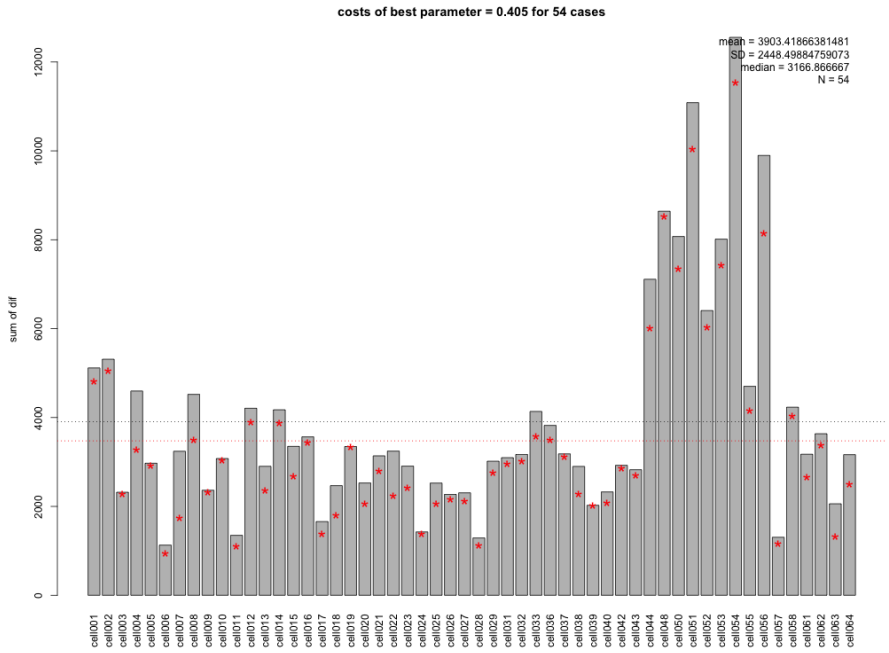


Figure 2.14: Distribution of errors for the overall best parameter. Red dots are the errors for the individually best parameters.

horizontal black dotted line is the mean of *error* (3903.419). The red asterisks mark the *error* of the individually optimized parameters (which must never be higher than the bar, by definition). The horizontal red dotted line is the mean of the *error* of the individually optimized parameters (3474.619).

The mean *error* of the overall best parameter is 12.34% higher than the mean *error* of the individually optimized parameter. For each cell, the *error* of the overall best parameter surpasses the *error* of the individually optimized parameter by 0.13% to 84.93%, with a mean of 14.70%.[6]

[6]Note that the ratio of the means of the *error* is different from the mean of individual ratios of the *error*.

The scores of the overall best parameters range from 0.786 to 0.979 with a mean of 0.930, median = 0.939, and SD = 0.040 as shown in Figure 2.15.

2.9 Results and discussion

The results reveal that: (i) the random walk algorithm can potentially be optimized to achieve an average accuracy score of 93.83%. The size of a cell is slightly negatively correlated with its segmentation score, with a correlation coefficient of -0.58. This suggests that the algorithm tends to work slightly better for smaller cells. (ii) the 54 ground truth images are representative of one another. The optimized parameter for one training set is also good for the test set no matter how the sets are divided. This was confirmed by randomly splitting the cells into training/test sets for 131 iterations. The expected error on the test set from using the best parameter from the training set is only 1.77% higher than the best overall parameter for the test set itself. Finally (iii), we found that 0.405 is the best overall threshold parameter for binarizing the 54 cell images where we have ground truth. The average accuracy score is 93.00%.

Our study can be repeated when more ground truth data becomes available. It can also be extended to study the nature of cell overlaps based on ground truth where two cells are individually injected with different dyes (Figure 2.1).

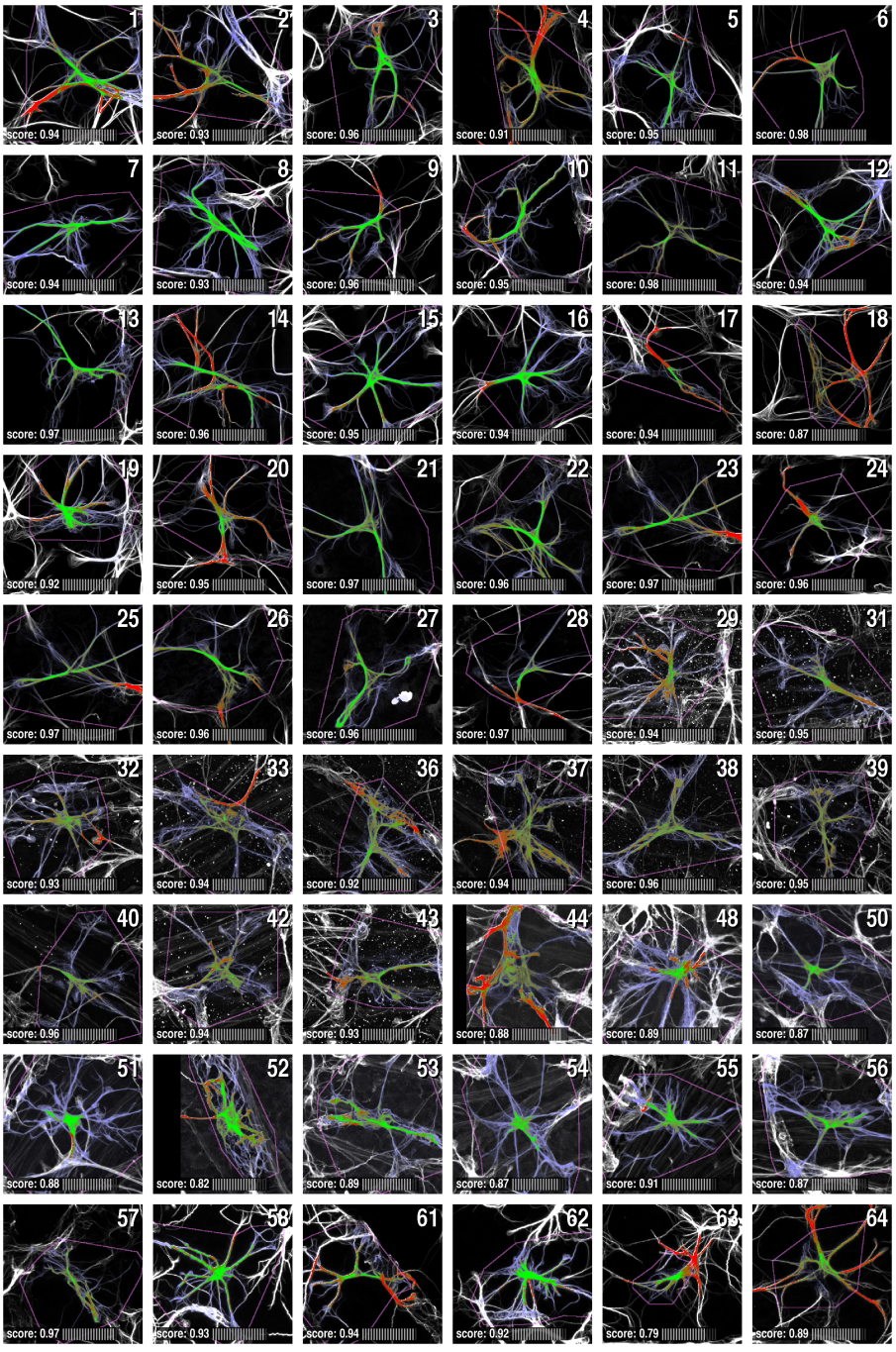


Figure 2.15: The scores of the overall best parameter

Although data acquisition for dual-injected cells is much more difficult than single cell injection, the Brainbow technique described in [48] and [49], where images of individual neurons of transgenic mice appear in multiple colors, may provide a rich source of ground truth data in the future when the technique is adopted for astrocytes.

Additional information about the single cell injection study including details, raw data, and images can be found at `http://ilabsvn.cs.ucsb.edu/projects/retivis/segop.html`.

Chapter 3

Network Study

Our initial goal in the astrocyte network study is to reliably and objectively construct astrocyte networks based on the overlap of their segmentation results. A specific objective is to obtain eight astrocyte networks for further analysis if there is an objectively clear reason that the networks are more plausible than other alternatives. A more general objective is to prepare multiple alternative networks based on reasonable techniques and parameter ranges to be chosen later with domain experts or with a more robust criteria when one becomes known.

We experimented with multiple approaches to building astrocyte networks for a further study of their network properties and characteristics. One major approach is based on overlapping grayscale images of the cells, and pruned by Voronoi connectivity (Section 3.2). Another major approach is based on overlapping binarized images of the cells (Section 3.3). This requires a proper selection of thresholds for binarization of segmentation results. We experimented with the optimized

threshold from the single cell injection study, an adaptive thresholding binarization algorithm (as implemented by Aruna Jammalamadaka), and multiple fixed thresholds. In the case of fixed thresholds, we visualized the eight binarized retinas in context according to 114 thresholds with various visualization techniques: coloring individual cells, coloring the connected components, or showing node-link diagrams. Our different visualizations based on static images with relevant connectivity details, large mosaics of multiple thresholds and multiple retinas (Figure 3.8), and synchronized video formats work together to help us determine proper binarization thresholds for network construction.

3.1 Defining an astrocyte network

A network (or graph) is a set of connections between astrocytes. A pair of astrocytes either has one or zero connection. A connection may be given a *weight* or *score*; the higher the weight, the stronger the connection.

Before any analysis can be done on the network, a few fundamental decisions need to be made in network formation. These include a criterion for connection between two cells, the connection weight metric, and any dependence on a parameter. If there is a parameter, a proper value or an algorithm of arriving at the proper value has to be determined.

Three approaches for defining the network have been attempted:

1. **Using overlapping circles**

   This approach was taken by Brian Ruttenberg [16]. From the segmentation results (which is grayscale), the area of each cell is calculated. A circle is drawn around each cell center with an area proportional to the area of the cell. Two cells have a connection if and only if the two circles intersect.

2. **Using binarized segmentation**

   Segmentation results are binarized (with a fixed or adaptive threshold). Two cells have a connection if and only if their binarized segmentation images overlap by at least 1 pixel (0.09591223 μm^2). The weight of the connection is the number of overlapping pixels.

3. **Using grayscale segmentation**

   For each pair of cells, the sum of the product of the grayscale segmentation images is calculated[1]. Two cells have a connection if that sum is above a certain threshold. The weight of the connection is the sum.

All the three approaches have their limitations and are dependent on some parameters or the choice of method. The first approach requires a method to calculate the cell's area from a grayscale segmentation result, and a decision on

[1]The sum of an image is the sum of pixel intensities. The product of two images is an image where each pixel is a product of its two corresponding pixels.

how to translate that area into the size of the circle. The second approach requires a method to binarize the cells. The third approach requires a decision on the threshold of the sum. It also raises the question of whether using the sum of the product is the right approach.[2]

3.2 Using Voronoi neighboring as a precondition

We have experimented with Approach #3, creating network from grayscale overlap of segmentation results. This approach has a major drawback in computation time and result complexity. The number of cell pairs with overlapping region of interests for which a grayscale image overlap needs to be computed quickly overwhelmed the system – taking more than 12 hours to compute one retina. Many far-apart cells still have some common pixels according to the segmentation results. The resulting graph is highly complicated and contains many connections between far-away cells that are biologically implausible.

To reduce complexity, we made a simplifying assumption:

> "If two cells are not Voronoi neighbors, they are not connected."

As a result, the number of pairs of cells that need to be considered for GFP1 dropped by a factor of 793.[3] The computation time arrived at under 10 min-

[2] An alternative is to use the sum of the pixel-wise minimum.
[3] from 11,288,376 to 14,229 pairs

utes/retina. The network is shown in Figure 3.1. Each line is a connection between two cells. Two cells have a connection if and only if they are Voronoi neighbors and their segmentation results have an overlap score higher than zero. The overlap score is the sum of the pixel-wise multiplication of the brightness values (each between 0 and 1). All connections are drawn with the same line color and thickness regardless of overlap score. There are 14,018 connections with the scores ranging from 10^{-6} to 4,221.47 with a mean of 383.96, median = 198.09 and SD = 488.19.

For all 8 retinas[4], the degrees range from 0 to 12. The median degree is 6 for every retina. If all of the connections are drawn with equal weight and opacity, the networks appear similar to the Voronoi diagrams themselves as shown in Figure 3.2. If the connection score is encoded into line thickness and opacity[5], the stronger connections become more visible as shown in Figure 3.3.

An informal survey of the data suggests that two cells may be physically connected even if they are not immediate Voronoi neighbors. In some instances, those connections may even be stronger than some Voronoi neighbors. We have briefly experimented with relaxing the Voronoi neighbor requirement to allow for

[4]GFP1, 2, 3, 8, 10, 11, 12, and 13

[5]with line width varying from 1 to 5 pixels, and opacity from 0.3 to 1.0, scaling linearly across the range of connection scores

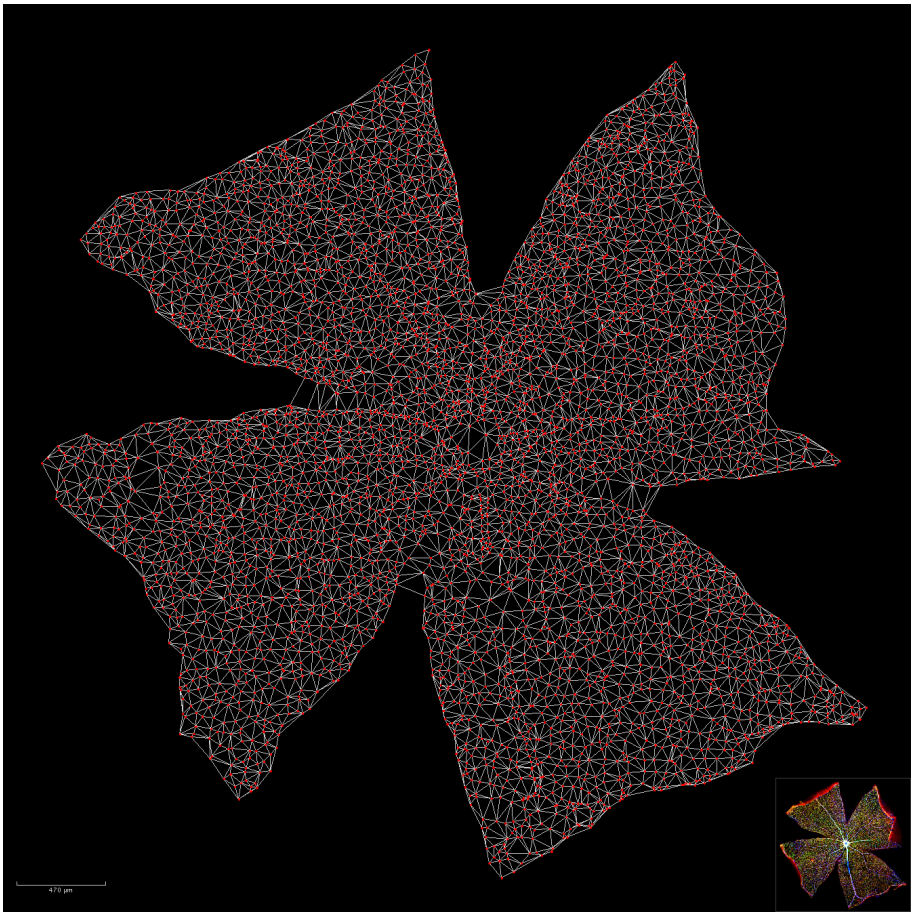


Figure 3.1: Grayscale overlap network of Voronoi neighbors for GFP1

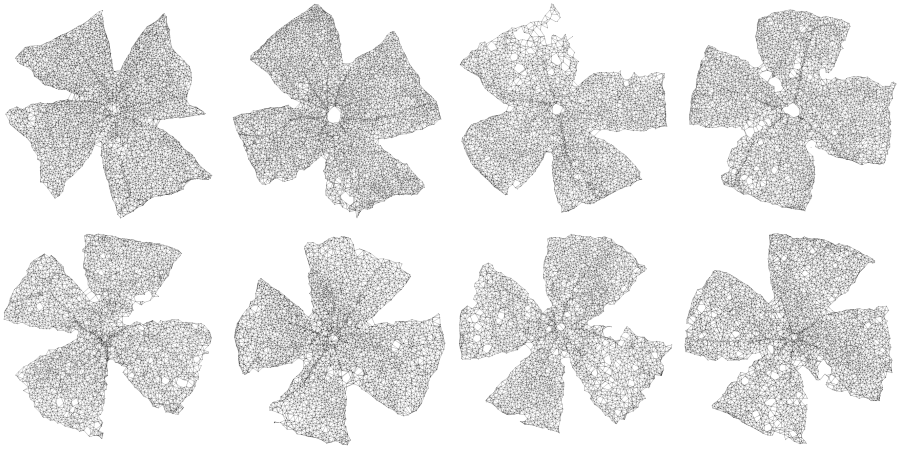


Figure 3.2: Networks drawn with equal weight

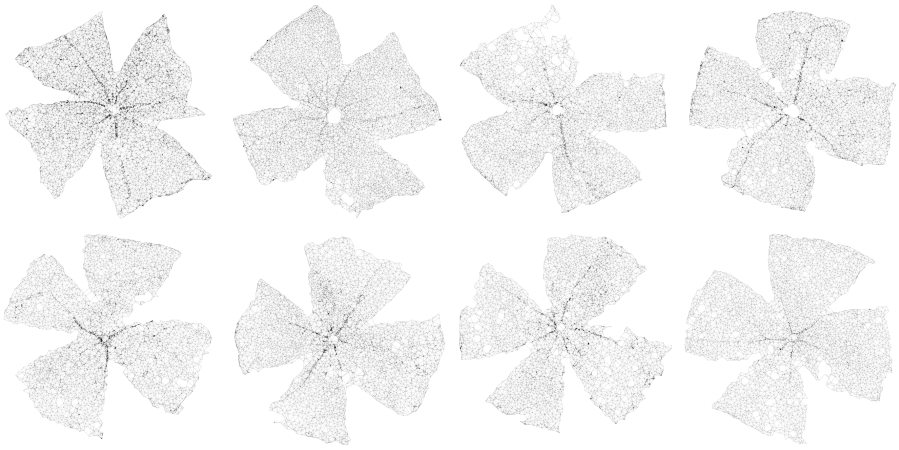


Figure 3.3: Networks drawn with varying weight and opacity

two-step neighbors (neighbor of neighbor) to connect. However, the resulting networks are too dense to be plausible for all retinas.

Therefore, it will be necessary to use a cut-off minimum weight threshold higher than zero. For this reason, we argue that it is necessary to study this cut-off parameter. While it is possible to perform this study based on grayscale overlap, it is more intuitive and computationally faster to do it in the binary image space.

3.3 Networks based on binary segmentation

3.3.1 Network generation

An overlap network based on binary segmentation is defined after the cells are segmented and binarized. Two cells have a connection if and only if their binary images overlap for at least one pixel. The number of pixels is the connection score or weight.

The algorithm to compute binary cell network is designed to work relatively fast by trading space for time. A 2D array the same size as the original image is allocated, each cell initialized to an empty linked list. For each white pixel of the binary image of every cell, the cell number is appended to the linked list located at the same array location as the pixel. After all cells are processed, each linked list is observed. For every cell pair contained in the same linked list, the connection score between the two cells is incremented.

Even though the number of possible cell pairs is $O(n^2)$ where n is the number of cells, the binary network can be computed relatively quickly in $O(n + wh + e)$ where w, h are image width and height and e = the actual number of edges. Since our image is large, $wh > e > n$; therefore, $O(n + wh + e) = O(wh)$. In practice, the $O(wh)$ algorithm to compute binary cell network is faster than the

$O(n^2)$ algorithm to compute the grayscale network because of a very high constant factor in grayscale overlap calculation.[6]

3.3.2 Using the parameter from cell injection study

When cells in GFP1 are binarized with threshold = 0.405 as obtained from Section 2.8, they are highly disconnected as shown in Figure 3.4. In consultation with our biology experts, we determined that it is unlikely that this correctly reflects all connections among Astrocytes. A reason for the non-transferability of the optimal binarization threshold obtained from our single-cell injection study might lie in different imaging and acquisition parameters and properties between the stained single cells and the 8 whole retinas. While it is unfortunate that this parameter cannot be used to generate a believable astrocyte network, we attempted the following estimation: in absence of reliable knowledge from the cell injection study, we seek to find a threshold which, when applied to binarize a retina, would connect the cells and create a network that appears biologically meaningful.

The threshold was varied from 0 to 0.9 with more samples at lower thresholds and fewer samples at higher thresholds. For each threshold, we binarized the cells

[6]Note that it is not immediately possible to use the same algorithm (utilizing a large 2D array) for grayscale network construction since the contribution to connection score from each pixel pair is fractional. The algorithm can be modified to accommodate this information, but it is likely to run into memory issues.

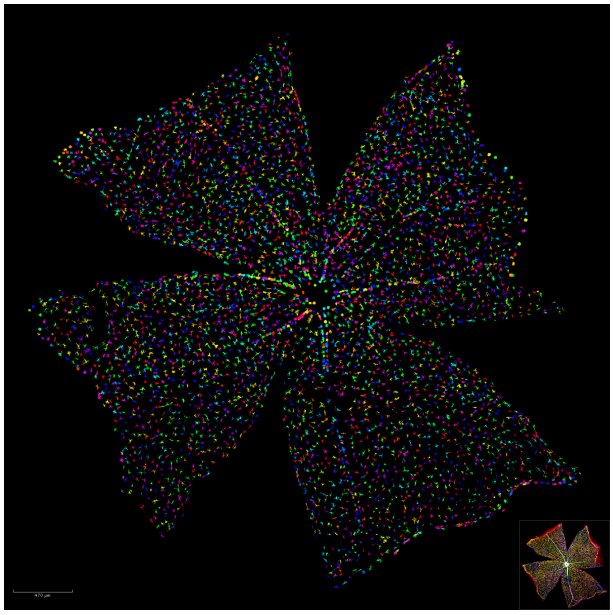


Figure 3.4: GFP1 binarized with threshold = 0.405

with that threshold and rendered the picture of the entire retina. Initially, we colored each cell with a random hue. To find a plausible threshold, we propose that the number of connected components and their sizes should be considered as an important factor. Therefore, for each rendering of the binarized retina, we computed all relevant statistics and created appropriate visualizations of the connected components and the networks.

3.3.3 Threshold and coverage analysis

For each binarized image of a retina, we calculate the *coverage percentage*, which is defined as "the size (pixel count) of the largest connected component divided by the total number of non-black pixels."

The *coverage percentage* is 100% when all of the cells are connected. When *coverage* is high, e.g. 99%, the largest component is much larger than the other smaller components (even if there are still a lot of unconnected components). We also calculate the number of islands (connected components), and $sumBin =$ total number of non-black pixels. As threshold increases, we expect the coverage to decrease, $numIslands$ to increase, and $sumBin$ to decrease.

For each threshold, a different color is assigned to each connected component[7]. To minimize the chance that nearby components have similar colors, we use a color map that is visually discrete rather than gradual[8]. For consistency, the first color in our map (purple) is always assigned to the largest component and, for every i, the i^{th} color is assigned to the i^{th} largest component.

The result is shown on Figure 3.6 with a zoomed-in view of GFP1 near the threshold 0.19 on Figure 3.5. The regions occupied by connected components are

[7]Using `label2rgb` function, we can create an RGB image from a Matlab's `labelmatrix` of the `bwconncomp` of the binarized image

[8]We used a modified version of the `'lines'` color map in Matlab. We make sure that no color is too dark by modifying the color map to change each luminosity from x to $0.5x + 0.5$ (so that each band is at least 50% bright)

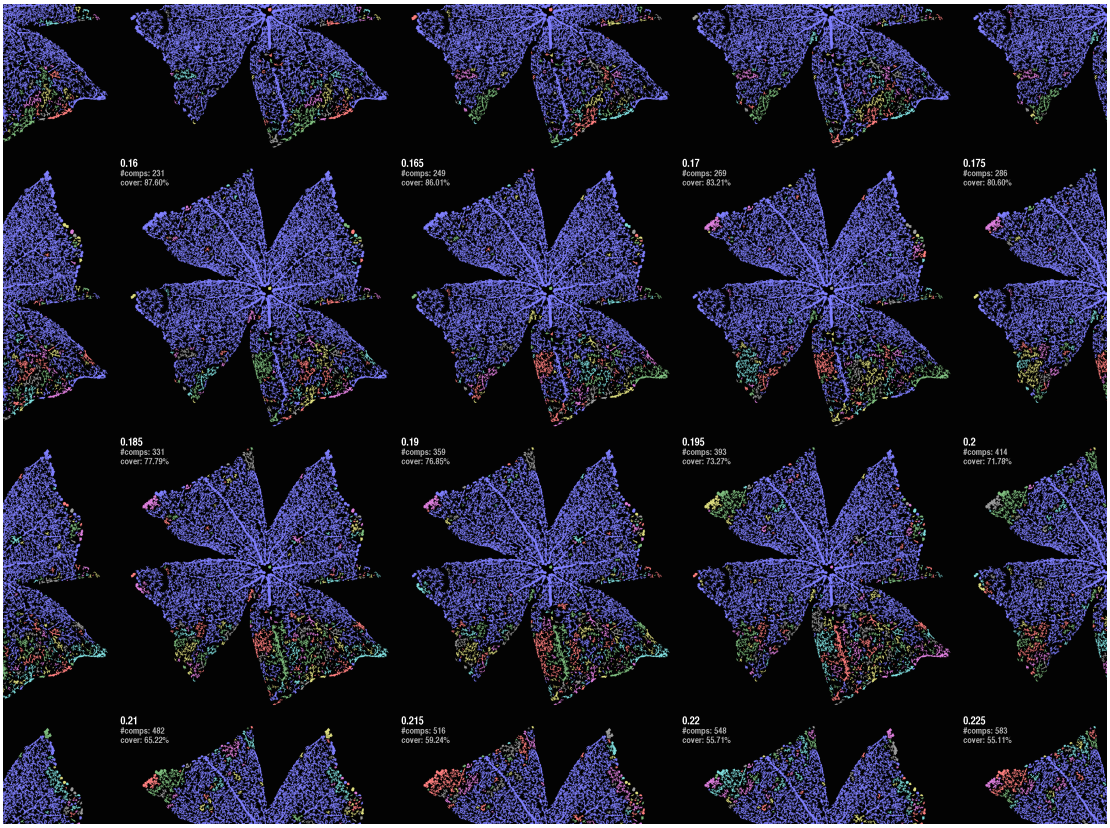


Figure 3.5: Some threshold variations of GFP1

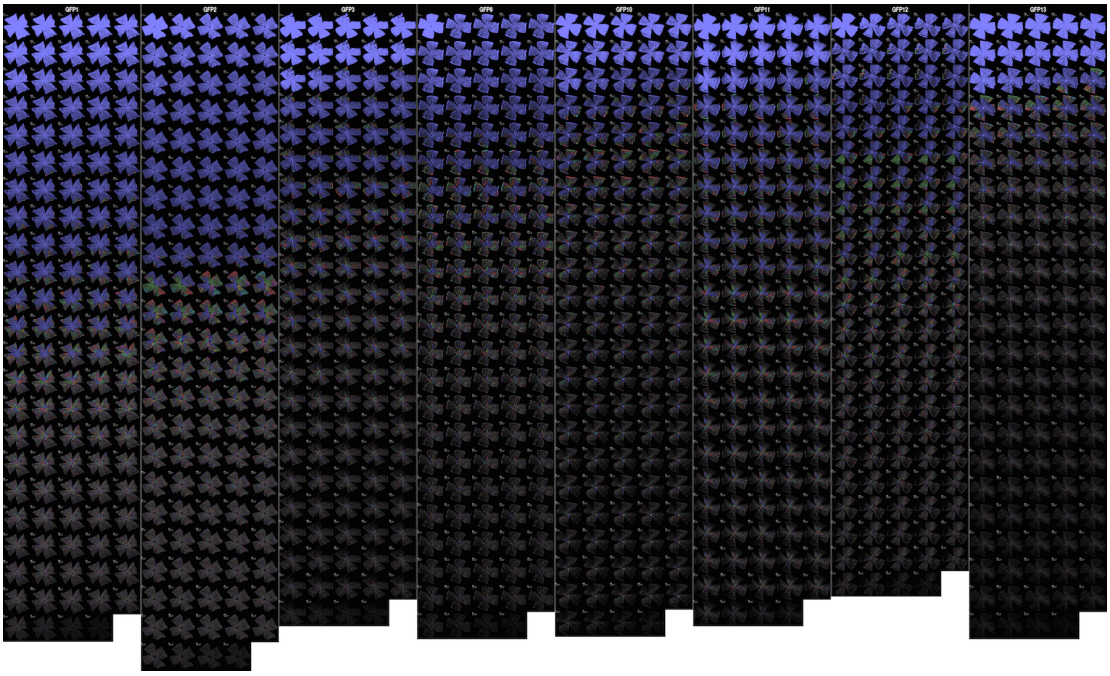


Figure 3.6: Binarization of the 8 retinas with 114 thresholds

more clearly visible compared to Figure 3.4. The biggest component always has the same color: purple. Hence, it is now possible to compare how the biggest component shrinks as the threshold increases.

From an informal visual exploration, we may decide to choose a threshold at which the coverage is around 75% or 50%. It would mean that, at that threshold, most astrocytes are connected. The network created among them, and among the smaller but not-insignificant components, may be interesting.

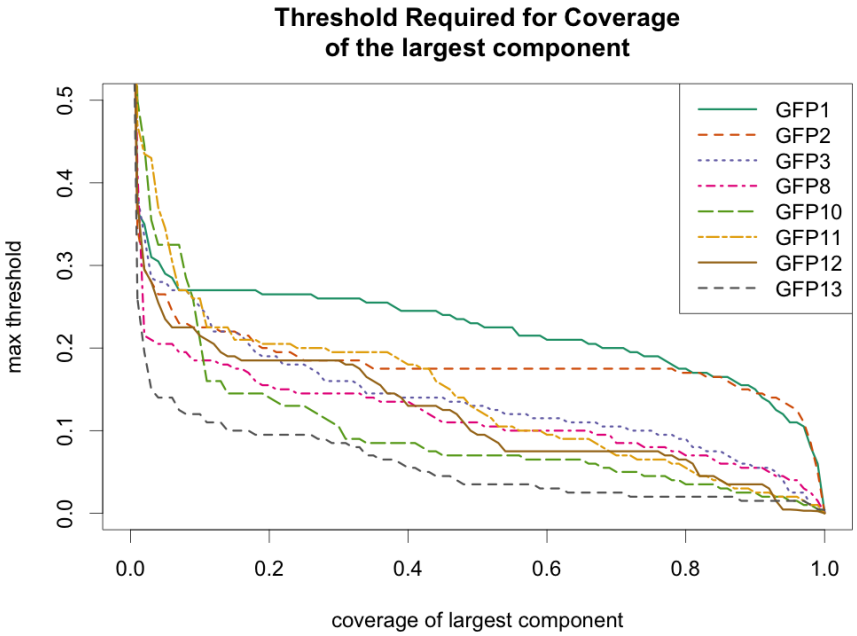


Figure 3.7: Thresholds required for coverage of the largest component

3.3.4 Thresholds required for minimum coverage

For a given *coverage percentage*, different thresholds may be required for different retinas to produce a binarization with at least that coverage. The higher the threshold, the lower the coverage. Therefore, a table has been computed where, for any minimum coverage percentage x ($x \in \mathbb{N}$, $0 < x \leq 100$), a set of 8 thresholds can be obtained such that they are the maximum known thresholds for which the binarizations have at least $x\%$ coverage.[9] A plot of these maximum thresholds required for each coverage amount is shown in Figure 3.7.

[9]The script in `th-VS-coverage.R` produces such table, which is saved to `AstrocyteRoot/info/thRequiredForCoverage.csv`.

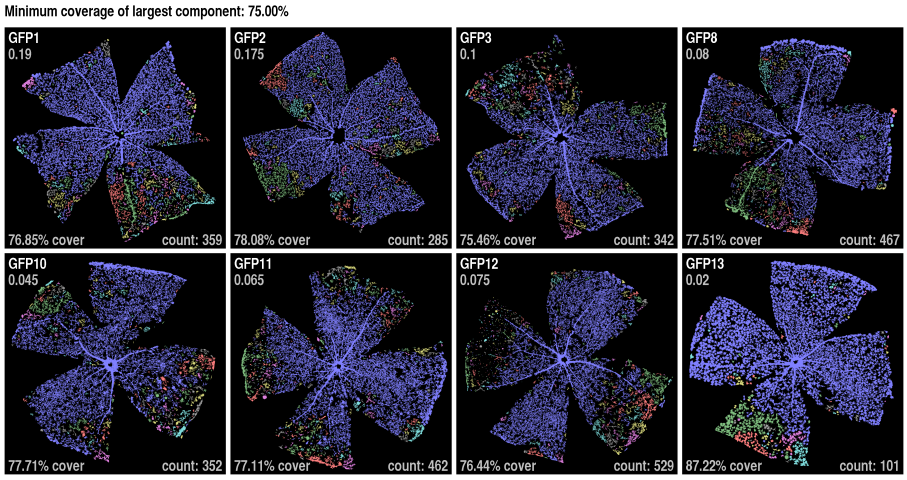


Figure 3.8: Binarizations with at least 75% coverage

From this information, a list of maximum thresholds can be obtained if it is determined that the networks must have at least a certain amount of coverage. For example, if the minimum coverage is 75%, the thresholds have to be at least 0.190, 0.175, 0.1000, 0.080, 0.045, 0.065, 0.0750, and 0.020 for the 8 retinas. The connected component and network visualizations are shown in Figure 3.8 and 3.9, respectively.

Two videos are created to show these coverage map for any coverage percentage [50] along with the network graphs [51][10]. In the two videos, the networks are organized by minimum coverage. For each frame in the video, the 8 retinas shown have the same minimum coverage (although they may have very different thresholds). As can be observed, for each frame, the retinas appear relatively

[10]Also available at `http://ilabsvn.cs.ucsb.edu/projects/retivis/pics/network/net-comps.mov` and `net-graphs.mov`

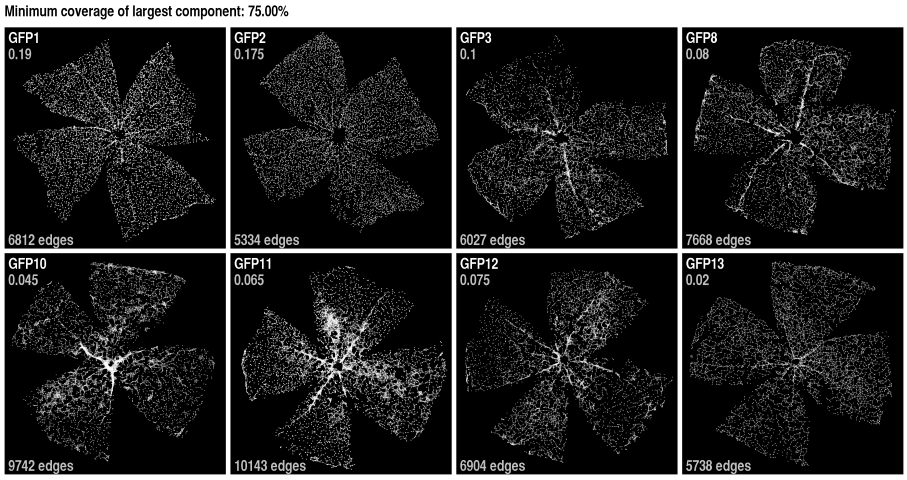


Figure 3.9: Networks with at least 75% coverage

consistent in the first video (`net-comps.mov`) that shows connected components, but they appear very different in the second video (`net-graphs.mov`) that shows the node-link diagram of the network.

We offer a possible explanation. To achieve a certain coverage percentage, one retina may be able to use a high threshold and the cells are still connected (albeit weakly) whereas another retina may require a very low threshold in order to make a sufficiently large connected component. When the threshold is very low, the cells becomes very large and the network becomes highly connected.

3.3.5 Other options in organizing the networks

Instead of varying the minimum coverage for each frame, the networks can be organized by simply varying the threshold. The thresholds are varied from 0.001

to 0.005 by a step of 0.001, from 0.005 to 0.5 by a slightly larger step of 0.005, and from 0.5 to 0.9 by a large step of 0.1.

For each frame of the video, a fixed threshold is applied to all retinas. The basic connectivity statistics are calculated. In one video, the coverage maps are created[11]; in the other video, the networks are visualized[12].

For both of these videos, each frame does not indicate any consistency across the retinas. This means that the image acquisition, preparation, or the natural differences among retinas do not allow for any consistency with respect to the binarization threshold.

Another option to organize the frames in the video, to enforce consistency across retinas in the node-link diagrams video, is to organize them by average degree.[13] Conceptually, for each frame, all retinas should have the same average degree. But such strict requirement may not be feasible because the threshold which yields exactly the desired average degree may not exist in our pre-computed set. Therefore, the definition of the frame can be relaxed. For each frame, the same minimum average degree is maintained for all retinas. For each retina, the maximum threshold that gives at least the minimum average degree is chosen. While this option may produce more consistent networks across the retinas, the

[11] `http://ilabsvn.cs.ucsb.edu/projects/retivis/pics/network/both/byTh-c.mov`
[12] `http://ilabsvn.cs.ucsb.edu/projects/retivis/pics/network/both/byTh-g.mov`
[13] degree = number of connections per cell

consistency is only artificial. By enforcing the average degree, any possible natural variations may be inadvertently overlooked.

3.4 Results of network study

We have generated and visualized astrocyte networks based on grayscale overlap of cells. The networks can be pruned based on the minimum overlap scores and on their Voronoi connectivity (immediate neighbors or two-step neighbors). The node link diagram can be visualized with either fixed edge intensity (Figure 3.2) or with varying thickness/intensity based on connection strength (Figure 3.3). This interactive view has allowed the biologists to detect some abnormally high connectivity for some cell pairs which were later determined to be an error in the cell center location data. Although error detection was not an explicit objective, it was a welcome side effect and a testament to the value of visualization.

We applied the optimum segmentation threshold as determined by our single-cell injection study (Chapter 2). Unfortunately, the resulting networks are too sparsely connected to be believable (Figure 3.4). We hypothesize that this is due to a still unknown difference in imaging or acquisition techniques between the single cell injection and the full retina mosaics.

We pre-computed networks based on cell binarization with multiple fixed thresholds. For each threshold, we recorded the node-link diagrams, the images of the full retinas populated with binarized cells, and relevant basic statistics such as the number of connected components, and the percentage coverage of the largest connected component (see, for example, Figure 3.5). We used the recorded data to visualize the networks in multiple ways, e.g. [50] [51].

Based on these visualizations, we concluded that

(i) No single fixed threshold implies reasonable networks for all retinas.

(ii) We cannot generate comparable networks for all retinas by specifying the minimum percentage coverage of the largest connected component either.

(iii) Based on subjective visual inspections and comparisons between the original images and the visualization of the binarized cells, the best thresholds for generating astrocyte networks for the eight retinas are listed in Table 3.1 with the number of connected components between 2 (GFP2) and 101 (GFP13) and the percentage coverage of largest component between 87.20% (GFP13) and 99.98% (GFP2).

(iv) The eight retina images are widely inconsistent in terms of illumination among themselves, and also among different areas within a single retina mosaic.

retina	threshold	numComps	coverage
GFP1	0.02	4	99.93%
GFP2	0.005	2	99.98%
GFP3	0.015	17	97.71%
GFP8	0.0015	11	99.61%
GFP10	0.005	13	99.54%
GFP11	0.015	48	97.61%
GFP12	0.0025	31	99.11%
GFP13	0.02	101	87.20%

Table 3.1: Best thresholds for creating networks based on visual observation

Additional information about the network study, including details, raw data, and images can be found at `http://ilabsvn.cs.ucsb.edu/projects/retivis/retinal-astrocyte-network.html`.

Chapter 4

Retina Clustering and Normalization

4.1 Introduction

The network study in Chapter 3 presented strong evidence that the 8 retinas are not consistent among themselves, at least regarding their imagery. Some retinas may be more similar than others. Although it is possible that a closer inspection of the networks and the connected component maps may yield some useful insight about the retinas' similarity or lack thereof, a more systematic, unbiased inspection of their other characteristics was deemed appropriate.

We clustered the eight retinas based on their histogram similarity of a certain set of variables. The objective was to study the nature of their differences, to determine the existence of a cluster whose members are generally similar, and to identify possible outliers. A secondary objective was to use this information as a

guideline on the necessity to normalize the retinas, or to possibly exclude certain retinas from being used in drawing general conclusions about astrocytes.

We attempted to cluster the retinas by defining the distance between them to be the distance between the histograms of some relevant variables. Twenty-one different variables that we identified in the beginning were later narrowed down to nine: Voronoi area, nearest neighbor distance, cell density, size of the cell in grayscale and in binary images, eccentricity, furthest geodesic point, distance to optic nerve head, and perimeter. See Chapter 5 for details of the nine variables and Section 1.6.2 for a listing of all variables.

The histograms for the variables were computed and visualized for all 8 retinas with equal scales in a single image (Figure 4.1). This allows for visual inspection of retina similarity or differences. Each variable was then individually analyzed for the 'distances' between retinas with respect to that variable. A matrix showing the heat map of the distances is shown for each variable. Two different metrics were used to compute distance between each pair of histograms: Minkowski distance of order 2 (Euclidean distance) [52] and a modified earth mover's distance (EMD) [53]. To visually verify that the distance metrics are correct, the 28 pairs of histograms for each variable are ranked according to each metrics and drawn from left to right (Figure 4.2). Also, all 28 x 9 = 252 pairs of histograms are ranked for their similarity and arranged from top-left (most similar) to bottom-right (most

different) to verify that the metric works properly in comparing histograms of two different variables (Figure 4.3).

With the validity of the distance metrics visually confirmed, we defined the distance between two retinas to be the root mean square of the 9 distances between their histograms. The similarity score, or attraction force, is defined to be inversely proportional to the retina distance. Finally, for each metric, we visualized the similarity matrix among the eight retinas (Figure 4.5). We identified the most different pairs and the most similar pairs, the outliers, and the most average retina.

4.2 Histogram profiles

The histograms of all the nine chosen variables are drawn for the eight retinas. They are shown in Figure 4.1. Each column represents a retina. Each row represents a variable (key). The histograms for each row are on the same scale, so they can be visually compared. The histograms for each column, however, are not directly comparable since different variables are drawn at different scales[1]. In all of these histograms, all data are kept – no removal of outliers or cells in damage areas – and there are 100 bins per histogram.

[1]the rendering of the histograms are not visually comparable but the number of items in their bins can be compared as explained in Section 4.4

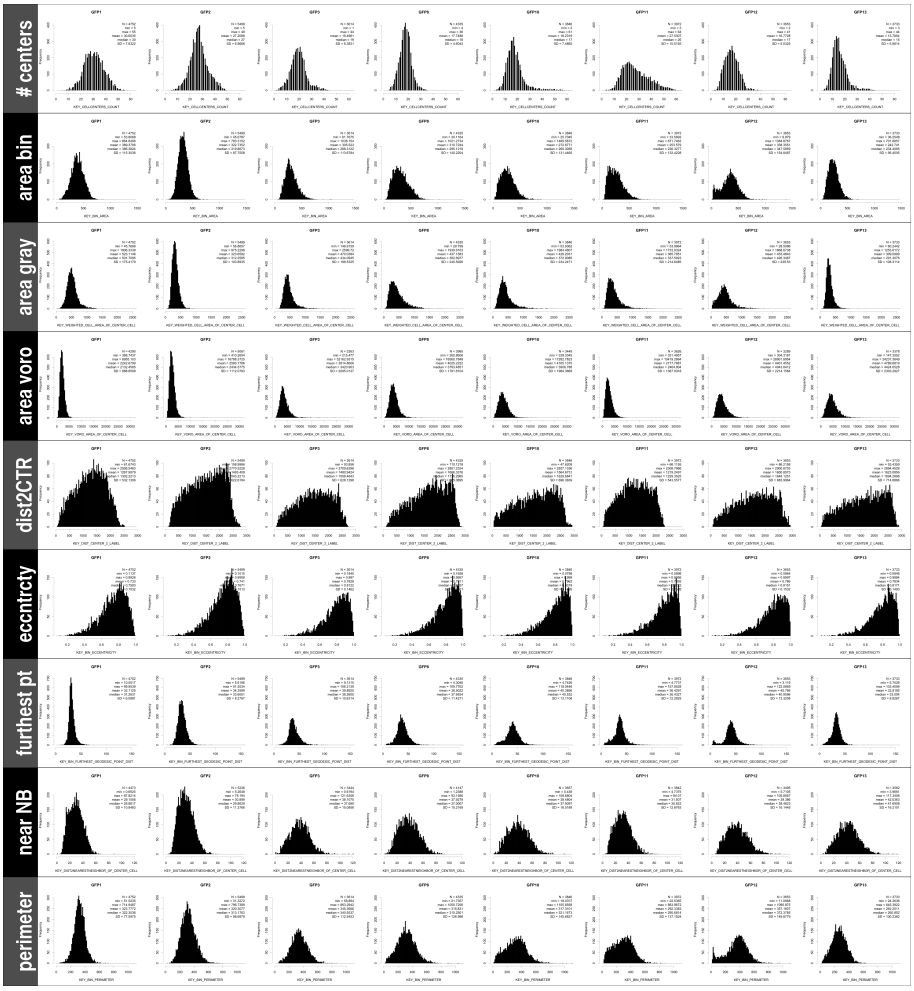


Figure 4.1: Histogram profiles for 9 relevant variables

An earlier version of the histogram profile with 21 variables, some of them are potentially relevant, is available online[2]. See Section 4.9 for more information about alternative setups and the similarity of their conclusions.

[2]Available at `http://ilabsvn.cs.ucsb.edu/projects/retivis/pics/allHistsAllRets_th.405_1405141118.png`

4.3 Metrics for histogram comparison

Several options are available for a metric that compares two histograms. Given two histograms with the same number of bins, bin size and bin boundaries, the metric should output a distance d where d is proportional to the perceptual difference between the plots of the two histograms. Although several statistical tests exist, e.g. Chi-square, Kolmogorov-Smirnov, and Kuiper [54], we argue that a simple visual verification of the ranking (Figures 4.2 and 4.3) is still needed to provide confidence in the metric.

The following metrics have been considered and implemented:

1. **EMD:** the Earth Mover's Distance between the two histograms. "Given two distributions, one can be seen as a mass of earth properly spread in space, the other as a collection of holes in that same space. Then, the EMD measures the least amount of work needed to fill the holes with earth." [53] Two histograms are assumed to have the same population size, or Σ(counts), so they are normalized by dividing the count in each bin by the sum of counts from all bins. Therefore, $\Sigma(\text{counts}) = 1$ after normalization. Note that this metric cannot account for the fact that one histogram may be large while the other one very small due to the difference in population sizes.

2. **mockEMD:** a modified version of the Earth Mover's Distance concept. The two histograms are modified to have the same Σ(counts) by first calculating the difference of sums: Δ. This is the amount of earth (dirt) that must be added to the smaller histogram. There are many different options for this step. We may:

   (i) add Δ to the first bin and let the earth mover move it to other bins, adding to the cost; or

   (ii) add Δ to all the bins uniformly; i.e. $\frac{\Delta}{\texttt{\#bins}}$ is added to each bin; or

   (iii) add Δ to the bins proportional to the amount of earth already in the bin.

   For simplicity, we chose Option (ii) – adding it uniformly.

   We then normalized the two histograms (dividing them with sum) and calculated `EMD`. We multiplied `EMD` with the sum of the larger histogram so that the amount reflects the real earth mover's work on the original histogram, not the normalized one. We then added Δ to the distance to reflect the additional work of acquiring new earth mass.

3. **minkowski.1:** the Minkowski Distance of order 1, the sum of bin-wise differences between the histogram. Also known as the Manhattan distance.

Generally, a Minkowski distance is defined as a distance between two n dimensional points P and Q where

$$P = (x_1, x_2, x_3, \ldots, x_n) \in \mathbb{R}^n$$

$$Q = (y_1, y_2, y_3, \ldots, y_n) \in \mathbb{R}^n$$

When used as a distance between histograms, P defines the first histogram and x_i is the size of the i^{th} bin; Q similarly defines the second histogram. The Minkowski distance of order p equals [52] [55]

$$\left(\sum_{i=1}^{n} |x_i - y_i|^p \right)^{1/p}$$

4. **minkowski.2:** the Minkowski Distance of order 2, the square root of the sum of squares of the bin-wise differences. Also known as the Euclidean distance.

Although we have implemented and tested all the four metrics, this thesis will focus on `mockEMD`. It has been observed that the `minkowski.2` metric also produces satisfactory outcome, based on visual verifications. The general conclusions from

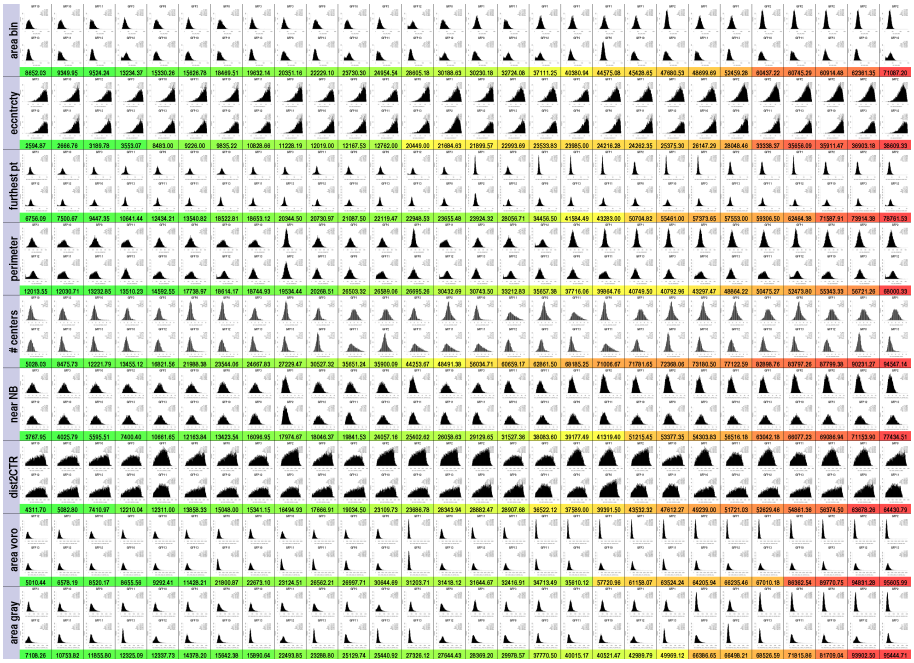


Figure 4.2: Sorted histogram distances for each variable

the final distance matrix visualizations among retinas are mostly similar between the two metrics.

For each variable, we compared all pairs of the histograms from the 8 retinas. There are $\binom{8}{2} = 28$ pairs per variable. The pairs were sorted from left to right according to their similarity. The result, shown in Figure 4.2, convinced us that the metric is effective at distinguishing similar histogram pairs from different pairs.

4.4 Comparing histograms of two variables

We have at least two reasonable metrics, `mockEMD` and `minkowski.2`, to compare different retinas with respect to any given variable. For any two pairs of histograms of the same variable (from two retinas), we can judge which pair is more similar by comparing the distances. In the situation where one pair is of one variable while the other pair is of another variable, however, we need to verify if it still holds true that if one pair has less distance than the other, the pair is indeed more similar.

We argue that it is true. Similarity of histograms is reflected in the visual similarity of the shape of the histograms, which reflects the shape of the distribution over the entire range of the variable. Two variables may span different ranges in their raw values, but when their histograms are drawn, the entire ranges of both are forced to fit in bins 1 to 100. Both `mockEMD` and `minkowski.2` only calculate distance based on those 100 bins, ignoring the distances between the bins. As for the y-axis of the histogram, which is the frequency, we may be concerned that the y-axes are not of the same scale across different variables. However, the scaling on the y-axis is used only for graphically rendering the histograms as images. When calculating the distances, the Y scaling is irrelevant.

It is possible to visually confirm this by ranking all $9 \times 28 = 252$ pairs of histograms by their distances (there are 9 variables with 28 pairs of retinas per variable). In order to compare them visually, histograms were drawn so that the Y axis of all variables are of the same scale. In Figures 4.1 and 4.2, the histograms of the same variable have the same y-max, which is the max count of the 8 histograms for the variable. But histograms from different variables have different y-max. In Figure 4.3, we re-drew all histograms, setting the new y-max to be the maximum of all previous 9 values of y-max. [3]

In Figure 4.3 the 252 histogram pairs were sorted by their distance from low to high (most similar to most different). Each pair occupies a square. The variable and the retina names are shown on the top two lines of the square. The bottom two lines show the raw `mockEMD` distance and the normalized distance as a percentage of the maximum. The colors are interpolated linearly in HSB color space between green and red. The histograms are shown side by side for visual comparison. The distribution of the distances are shown on Figure 4.4(a). Based on a visual observation of this ranking, we have concluded that the distance metric works reasonably well at comparing histograms.

[3] An outlier could make the rest of the histograms look flat to the ground. However, that is the only way that guarantee that we see all the data. An alternative is to pick the second highest y-max out of the 9 values. Some clippings will occur.

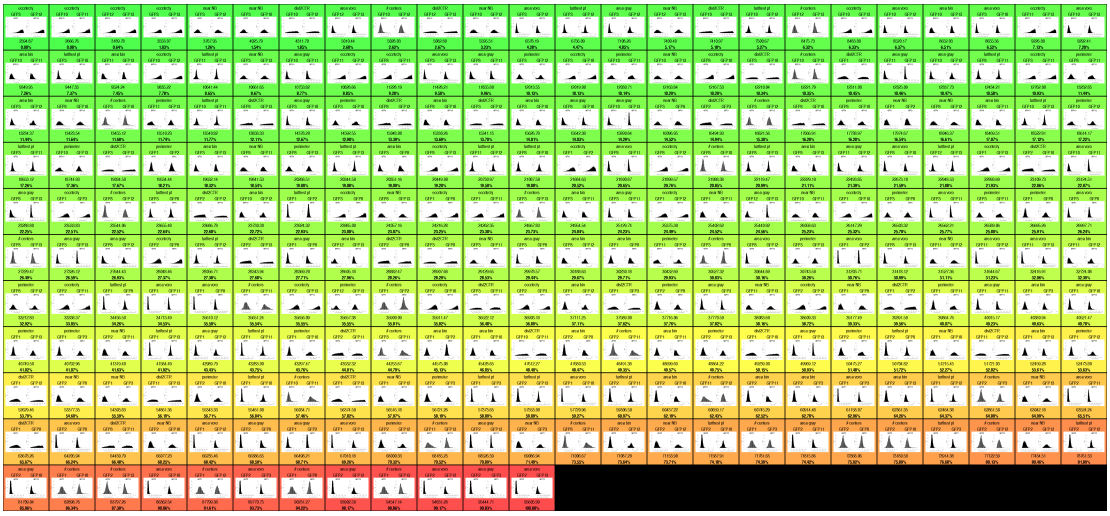
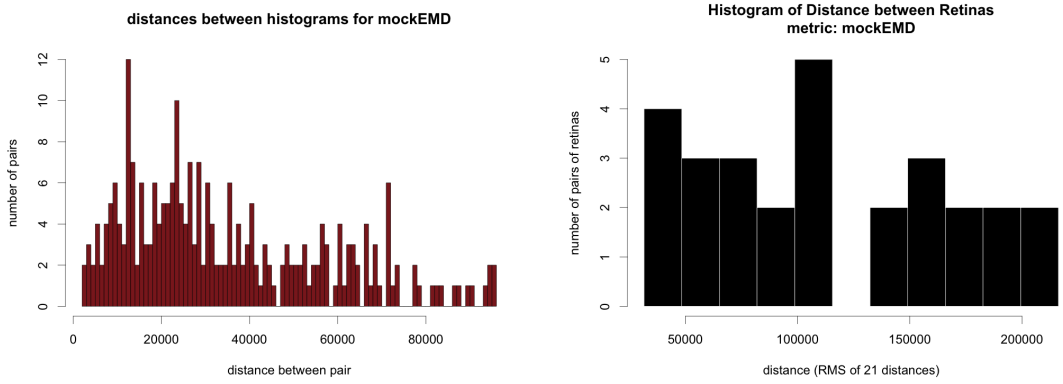


Figure 4.3: Sorted histogram distances for all variables



(a) between histogram pairs (b) between retinas

Figure 4.4: Distribution of distances

4.5 Distance and similarity between two retinas

The histogram distance between variables are used in the calculation of distance between two retinas. Considering each of the 9 variables as a dimension, the histogram distance represents the distance along that particular dimension. Therefore, we define the distance between two retinas to be the Euclidean distance in the 9-dimensional space – the square root of the sum of squares of the histogram distances.

$$\texttt{retina distance} = \sqrt{\sum_{i=1}^{9} \texttt{variable}_i{}^2} \tag{4.1}$$

The histogram of the distances according to `mockEMD` is shown in Figure 4.4(b).

The following observations hold true for the distances: (i) the ranges are unknown beforehand; (ii) it does not always start at 0;[4] and (iii) there's no upper bound. These observations guide the design of a similarity measure between retinas based on these distances.

To cluster the retinas based on their similarity, the relationships between the 8 retinas should be treated as a similarity graph. The graph has 8 nodes for the 8 retinas. The retinas are linked with edges. In a graphical layout, the nodes that are more similar should appear near one another while nodes that are more different should be further away. To achieve this, we can utilize a force-directed

[4] although it cannot go below zero.

layout where the nodes naturally repel one another (to prevent collapse) and the edges represent the attraction force between the nodes based on their similarity.

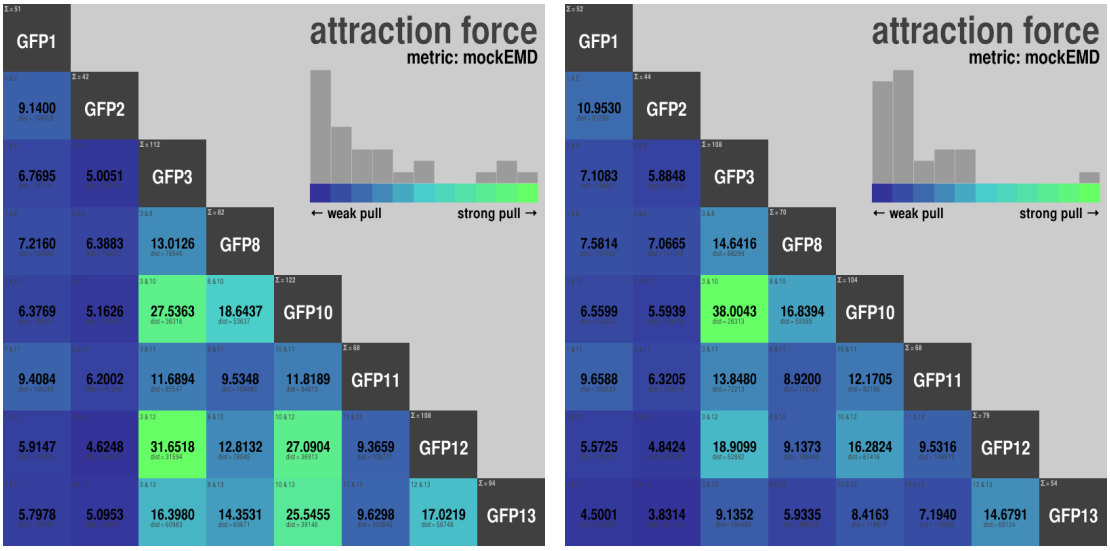
The distance measure needed to be translated into a similarity measure or attraction force. The attraction force between two retinas should be a function of the distance so that, the higher the distance, the lower the attraction force. Therefore, we designed the attraction force to be inversely proportional to the distance. To bring the numbers into a more proper range, we multiplied them by 10^6 (an arbitrary number).

$$\texttt{attraction} = \frac{10^6}{\texttt{retina distance}} \tag{4.2}$$

Even though this constant factor does not affect clustering, it contributes to the readability of the final visualization of the similarity matrix.

4.6 The similarity matrix

The attractions for all the 28 pairs of retinas are shown in Figure 4.5(a). The large number in each cell is the attraction force between the two retinas. The higher the number, the more similar the retinas, and the more green (and brighter) it appears. The histogram of these attraction forces are shown on the top right. The small numbers under the attraction forces are the raw distances, which



(a) before normalization (b) after normalization

Figure 4.5: Adjacency matrices comparing the retinas.

were used to calculate the attraction force but not directly used for calculating the square color. The retina names are shown on the diagonal along with the sum of their edge weights (designated by Σ at the top left of each square). These sum reflects how close the retina is to the other retinas overall[5].

4.7 Retina clustering observations

We have demonstrated an approach to comparing and clustering retinas based on chosen characteristics. Both the second order Minkowski's and the modified earth mover's distances are found to be suitable metrics of computing distances

[5]It is the sum of all numbers to the left of or below the square.

between histograms. For the simplicity, only results from the modified EMD (`mockEMD`) is reported here. In the first iteration where 21 variables are used for clustering[6], GFP3 and GFP12 are the most similar. GFP1 and GFP2, with their two lowest sum of edge weights, are the "outliers" because they are the least similar to others. GFP3 - 13 form a loosely connected clique with GFP3 being the "binding force" or an "average retina" who is the most similar to other retinas overall (because of the highest sum of edge weights). GFP2 and GFP13 are the most different.

In the second iteration where we selected only nine variables for clustering, with results shown in Figure 4.5(a), GFP1 and GFP2 are still the outliers (true for both metrics). GFP3 and GFP12 are still the most similar. GFP2 and GFP12 are the most different with a similarity score of 4.62.

After retina normalization, which will be described in Section 4.8, we clustered the retinas again. The results are shown in Figure 4.5(b). GFP1 and GFP2 are still the outliers. GFP10 and GFP12 are the most similar. GFP2 and GFP13 are the most different with a similarity score of 5.39, higher than in the case of non-normalized retinas. This suggests that normalizing the retinas has an effect of bringing the most different pairs closer together.

[6]result not shown here but available at `http://ilabsvn.cs.ucsb.edu/projects/retivis/cluster1.html#140520_1`

When the cells in damaged areas are excluded from histogram calculation (and after normalization), however, the overall similarity among retinas actually decreased. A preliminary investigation suggested that the increased difference in population sizes between retinas may be responsible for the increased distances between them.

4.8 Retina normalization

We normalized the images of the eight retinas to reduce discrepancies among them which are apparent in Figure 4.6. The main objective is to obtain a new set of images that are more evenly illuminated both within and between images. A version of normalized images is shown in Figure 4.7 which shows a significant difference from the original. In the final version, the difference in illumination is more subtle as shown in Figure 4.9.

We normalized the retinas using an approach based on adaptive thresholding [56] as shown in Figure 4.8. We applied an adaptive thresholding on the green channel to binarize it as an intermediate step.[7] This binarized version served as a tool to boost the brightness of darker parts of the image so it became more evenly illuminated. We then blurred the binarized green channel by some amount

[7]Adaptive thresholding takes two parameter: ws (window size) and C [56].

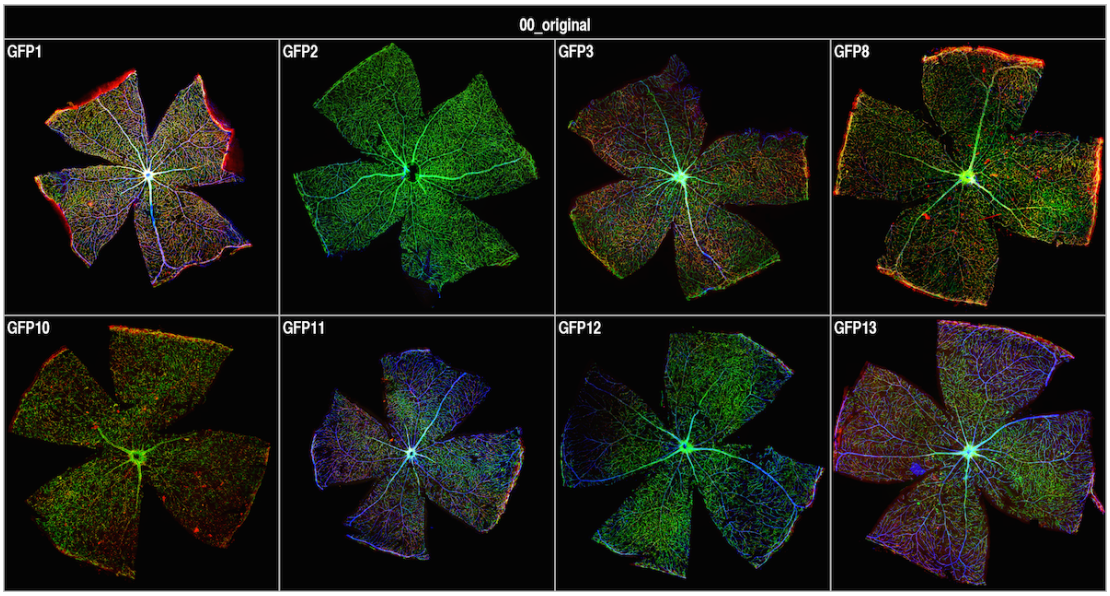


Figure 4.6: The original images before normalization

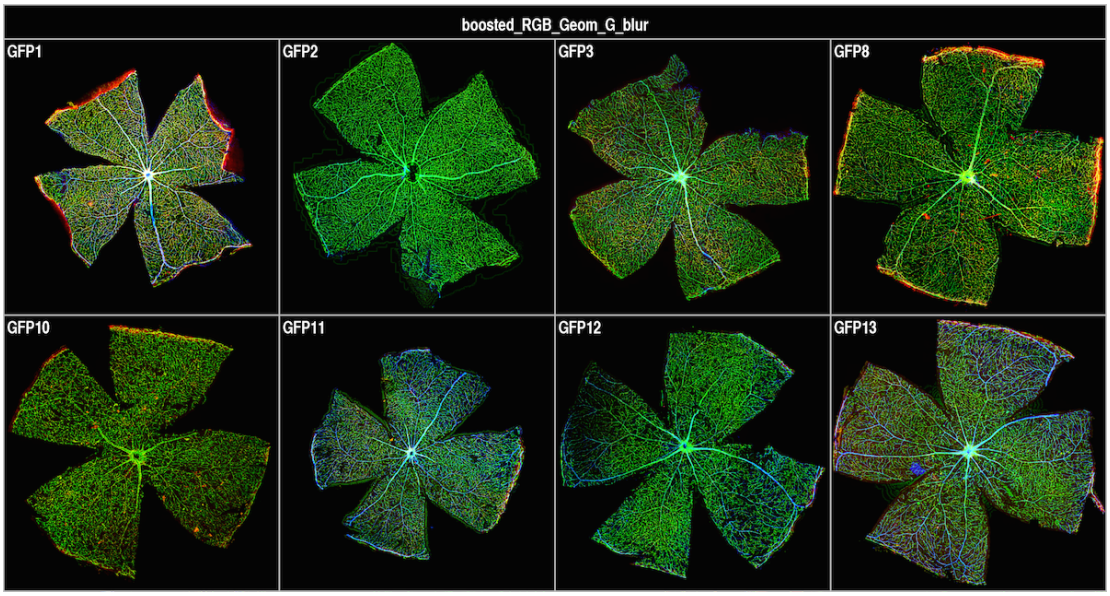


Figure 4.7: An extreme version of normalized images

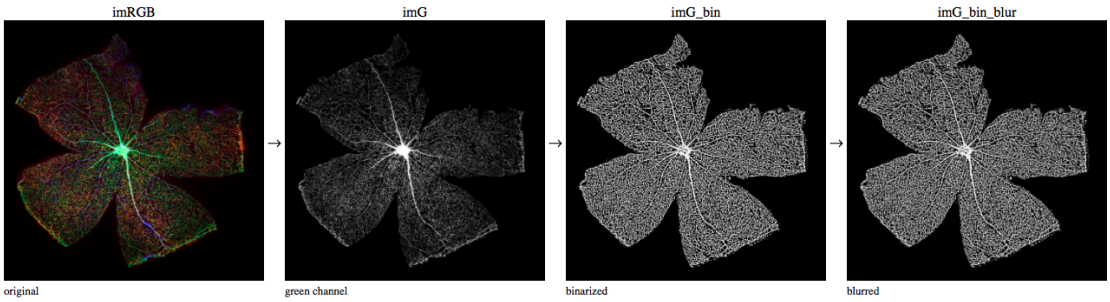


Figure 4.8: A normalization pipeline based on adaptive thresholding

to prevent sharp edge artifacts, and added some proportion of that image to the original green channel using a formula:

$$O = G + \max(0, B - G) \times G^{\gamma}$$

where

O = output

G = original green channel

B = the blurred binarized green channel

γ = a parameter

After normalization, we re-segmented and recomputed all the statistics for all the eight retinas.

We have used the following parameters for all the retinas in the normalization process: window size = 804, $C = 0.05$ (for adaptive thresholding), blur radius = 6, and $\gamma = 1/3$. From preliminary inspections, we have noticed no significant qual-

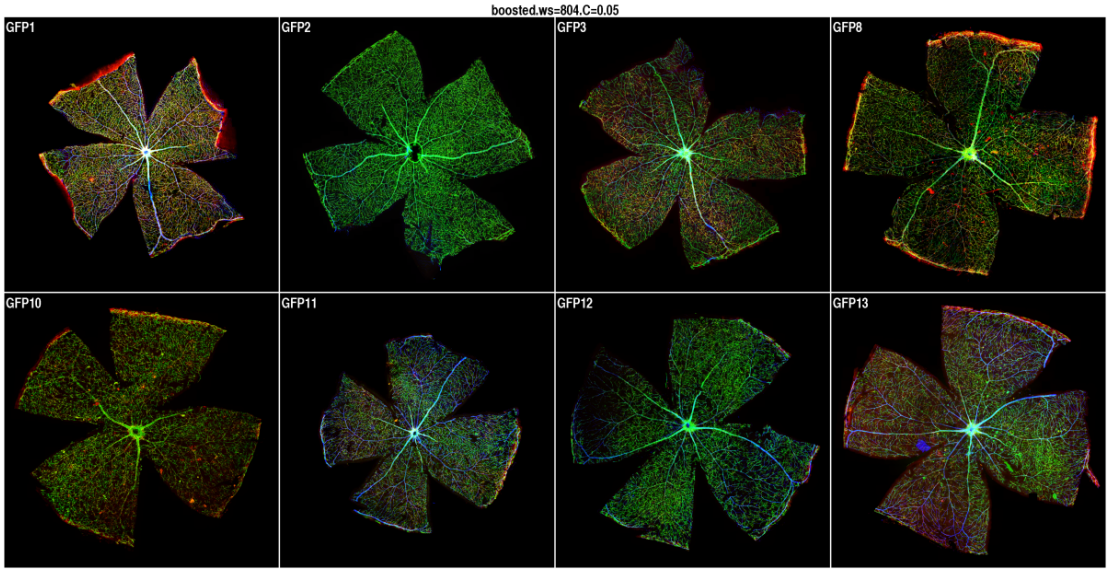


Figure 4.9: The normalized images
(The strip artifacts on GFP8 and 13 are not present in the final version.)

itative difference in the distribution of the nine relevant variables. This suggests that our analysis methods are robust against slight changes in image properties.

Our decision to use adaptive thresholding to normalize the retinas has yielded an acceptable result. We had hoped that the normalization effect would be stronger. But in choosing our parameter set, we have to make sure that the decision is universal – that is, all retinas must use the same set of parameters – to prevent any human biases. The *blur* parameter is instrumental in the strength of the final results. Although, lower *blur* results in a much more evenly illuminated set of images, the results would only be satisfactory when viewed at a lower than 10% magnification because of the strong edge and halo artifacts at the 100% zoom

level. For future work, an approach based on adaptive histogram normalization should be explored. (See [57] [58] and [59].)

4.9 Alternative results and other information

All possible scenarios for comparing and clustering retinas have been experimented. Eight possibilities arose from the following series of decisions.

1. Whether or not to normalize the retinas.

2. Whether or not to exclude cells under the damage areas

3. Which distance metric to use for histogram comparisons: `minkowski.2` or `mockEMD`

The directory `compareRets` is located under `AstrocyteRoot/info`. Under `compareRets`, two subdirectories `compareNotNormalized` and `compareNormalized` contain results of retina histogram comparison and clustering before and after normalization, respectively. Under each of those directories, two subdirectories `auto-histograms-collection-withDamage` and `auto-histograms-collection` represent the scenarios where all cells are included in the analysis VS where the cells on the damage masks are excluded, respectively. Inside each of the `auto-histograms-collection*` directories,

- the histogram profile similar to Figure 4.1 is the file `allHistsAllRets.png` located under `combined`.
- the similarity matrix similar to Figure 4.5 is the file `matrix.mockEMD.png` located under `compare/mockEMD/graphDrawings`. For a difference metric, replace `mockEMD` with `minkowski.2`.
- Individual histograms for every variable and every retina are located at `byKey/KEY/retName.png` while the raw bin counts (a series of 100 numbers) are recorded at `retName.counts.txt` in the same location.

Additional information about the retina clustering including details, raw data, and images can be found at `http://ilabsvn.cs.ucsb.edu/projects/retivis/cluster1.html`. For retina normalization, additional information can be found at `normalize.html` on the same site.

Part III

Visualization results

Chapter 5

Variables of interest

5.1 Version of the retinas

Our data set consists of eight full retina mosaics. The images of the retinas have been normalized as describe in Section 4.8. The cell center locations are manually marked and segmented by a random walk segmentation algorithm as described in Sections 1.1.2 and 1.1.3. The segmentation results are binarized with adaptive thresholding. Cells whose centers are located on the damage areas as manually masked by the biologist are excluded from the statistics.

5.2 Selection of variables

Several variables are available for region-based analysis. They are listed and described in Section 1.6.2. Specifically, only numerical variables listed on Table 1.1 can be visualized as a heat map. An initial selection of 21 variables is made

based on criteria that they are potentially interesting and they are not obviously duplicates of one another. Although some variables still represent the same biological quantity in different ways, such as `[area bin]` and `[area gray]`, they may provide useful insights into the binarization process and may give us a degree of confidence in the interpretation of any possible conclusion. For easy reference, each of the the 21 variables is assigned a concise and informative nickname listed on Table 5.1.[1]

Variable	**Nickname**
`avg chan0 r`	`[r]`
`avg chan1 g`	`[g]`
`avg chan2 b`	`[b]`
`bin area`	`[area bin]`
`bin convex hull area`	`[area hull]`
`bin convex hull holes`	`[hull holes]`
`bin eccentricity`	`[eccntrcty]`
`bin extent`	`[extent]`
`bin filledarea`	`[area filled]`
`bin fraction of convex hull`	`[solidity]`
`bin furthest geodesic point dist`	`[furthest pt]`

[1]These nicknames were also used for the histogram profile in Figure 4.1.

`bin majoraxislength`	`[axis major]`
`bin minoraxislength`	`[axis minor]`
`bin num holes`	`[holes]`
`bin orientation`	`[orient]`
`bin perimeter`	`[perimeter]`
`cellcenters count`	`[# centers]`
`dist2nearestneighbor of center cell`	`[near nb]`
`dist center 2 label`	`[dist2ctr]`
`voro area of center cell`	`[area voro]`
`weighted cell area of center cell`	`[area gray]`

Table 5.1: The 21 variables and their nicknames

The variables whose names start win `bin` depend on the binarization of the segmentation results. All other variables are not influenced by the segmentation results except for `weighted cell area of center cell` (or `[area gray]`).

The 21 variables are visualized and viewed. A discussion between the collaborating computer scientists and biologists on the visualization results led us to conclude that the selection can be further narrowed down to 9 variables as described in Section 5.3.

5.3 Descriptions of variables

We are interested in visualizing the following variables. The order given below is the same order as the detailed report in Appendix A. The first two variables, `[area bin]` and `[area gray]`, are related to the size of cells. The next three, `[area voro]`, `[near NB]`, and `[# centers]` are related to cell population density. The variable `[eccntrcty]`[2] is related to cell shape, along with `[furthest pt]` and `[perimeter]` which are also slightly related to cell size.

1. `bin_area [area bin]`

   The binarized cell area. This is the areas of cells after binarization with adaptive thresholding. It is calculated by counting the pixels in the binary image.

2. `weighted_cell_area_of_center_cell [area gray]`

   The weighted cell area. This is calculated from the grayscale segmentation result by summing up the pixel values, each pixel being between 0 and 1. This is not influenced by the choice of binarization threshold or by the adaptive threshold implementation, but it is influenced by the random walk segmentation algorithm.

[2]It is abbreviated from 'eccentricity'. The nicknames were initially designed for use in Figure 4.1 which offers very limited vertical space for variable labels, hence the need for an extreme abbreviation.

3. `voro_area_of_center_cell [area voro]`

   The area of Voronoi region of the cell. For each retina, a voronoi diagram is created from the locations of the cell centers. For cells whose voronoi region intersects with the retina boundary or the optic nerve head, this value is undefined. An example of voronoi cells is shown in Figure 1.9(f).

4. `dist2nearestneighbor_of_center_cell [near nb]`

   The distance to the nearest neighbor. If this distance is more than the distance to the boundary, it is considered undefined. This is because there might be another cell on the opposite side of the cut that is closer than the known nearest neighbor. Figure 5.1 shows the white arrows pointing from every cell to its nearest neighbor. Some cells are mutual nearest neighbors as indicated by a double-headed arrow ($\leftrightarrow$). Although every cell has a nearest neighbor, some cells are not a nearest neighbor of any other cell – as indicated by cells without an incoming arrow head.

5. `cellcenters_count [# centers]`

   The cell centers count. This represents the density of cell centers. It is the number of cell centers within 150 μm radius of the cell center. Note that this value can be abnormally low around the edge of the retina because the 300 μm-diameter circle may fall outside of the boundary. We have considered

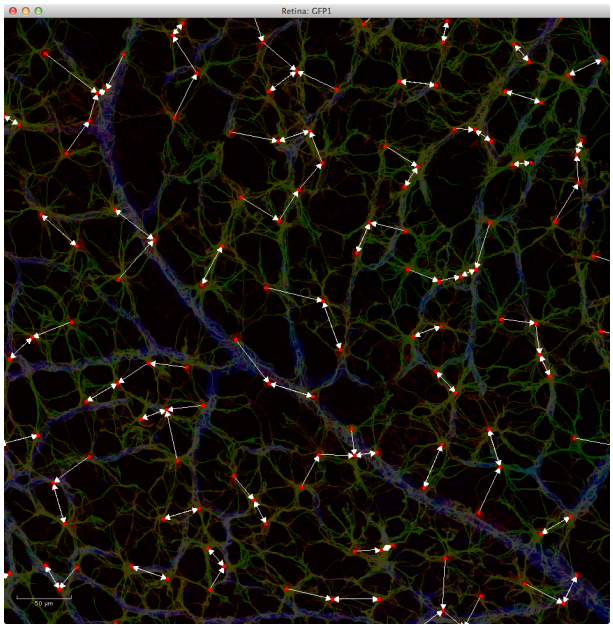


Figure 5.1: Nearest neighbors graph

using another variable, the density – which is this value divided by the area of the intersection between the circle and the retina boundary, but decided against it because that variable suffers from artifacts of abnormally high values when the intersected area is very small.

6. `bin_eccentricity [eccntrcty]`

   According to MATLAB manual, "the eccentricity is the ratio of the distance between the foci of the ellipse and its major axis length." The ellipse here refers to an "ellipse that has the same second-moments as the region." [38] A circle has the lowest eccentricity (0) while a line segment has the highest eccentricity (1). When applied to a binarized image of a cell, it represents

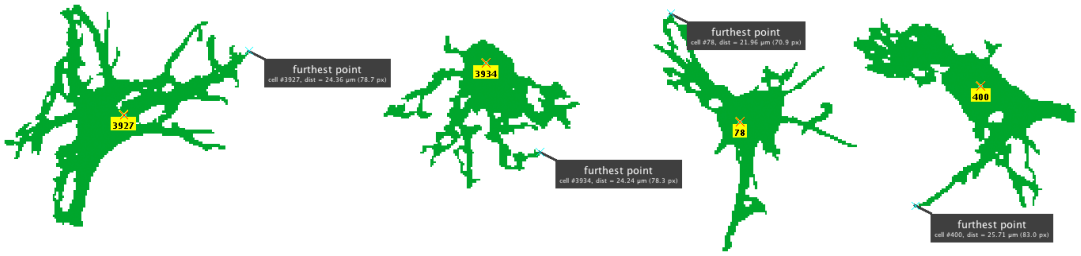


Figure 5.2: Furthest point distances

how elongated the cell is. Cells with lower eccentricity are shaped like blobs. Cells with higher eccentricity are more elongated in certain directions.

7. `bin_furthest_geodesic_point_dist [furthest pt]`

   This is the distance from the cell center to the furthest point on the binarized cell body. It is not a Euclidean distance, but rather a distance of walking along the cell (geodesic distance). In the special circumstance where the cell center falls outside the binarized cell body, a straight line is drawn from the center to the closest landing point on the cell body, and this value is the length of this line plus the geodesic distance to the furthest point on the cell from that landing point. Figure 5.2 provides some examples.

8. `bin_perimeter [perimeter]`

   This is the perimeter of the binarized cells.

9. `dist_center_2_label [dist2ctr]`

   The distance from the cell center to the optic nerve head.

Chapter 6

The visual design

6.1 A single retina view

Each retina is drawn inside a square box on a white background. There is a small disc that represents the value for each cell. The size of each disc is designed to be large enough to be clearly visible but small enough to not overlap too much with the neighbors in dense areas. The discs are semi-transparent to allow for some overlapping values to be shown. The values are categorized into three groups: the bottom five percent, the middle range, and the top five percent. Complying with normal conventions, the bottom and top groups are assigned the color blue and red, respectively.[1]

[1] In photography, as in microscopy, blue is typically assigned to designate underexposed areas whereas red designates overexposed areas.

6.2 Color scheme

For the values in the middle range, the colors are linearly interpolated in the HSB color space between two colors. The low value color is a very light blue (`#DEEDFF`) and the high value color is a darker pink (`#FF8082`). This color scheme was chosen because

- The low-value color is designed to be similar to blue, the color chosen for the bottom group.
- The high-value color is designed to be similar to red, the color chosen for the top group.
- The low-value and the high-value colors are not too similar to blue and red so that the bottom and the top groups can stand out – to facilitate easy detection of the outliers.
- The low-value color is designed to be lighter (brighter) than the high value color so that the high values stand out more on a white background.

To demonstrate this color scheme, the distances to the optic nerve head of cells in GFP12 are visualized in Figure 6.1. This variable is chosen because an obvious pattern is known – the lowest values are in the middle and the value grows to the highest on the periphery. From this image, the color scheme is shown to

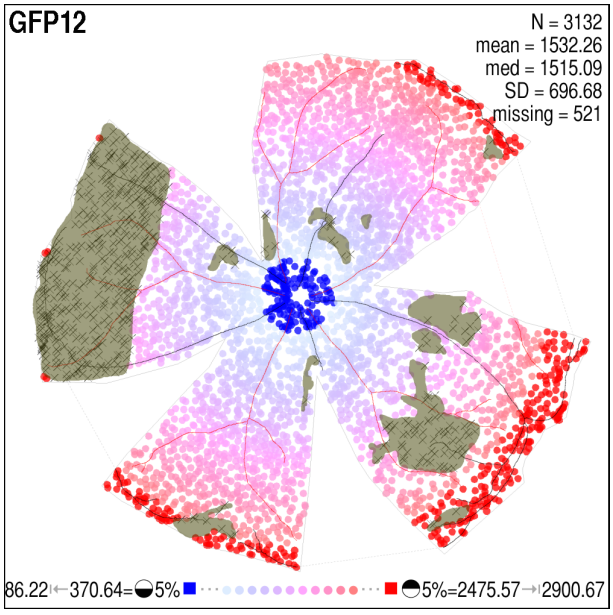


Figure 6.1: A single retina view showing distance to the optic nerve head

be effective at highlighting the lowest and highest values. It is also effective at representing the range of values in the middle range with double encoding, both in hue (from light blue to pink) and in intensity (from light to dark).

6.3 Contextual elements

Some areas of the retina suffer tissue damage during the sample preparation and imaging stage. Those areas are manually masked by our collaborating biologist, Gabe Luna. The damage masks are shown in dark yellow. The cells whose centers fall inside the damage area are marked with a cross ($\times$), signifying a missing value. Some cells may be outside the damage area and still be marked

with a cross; in such situation, the value of that variable for the cell is missing for some other reasons; for example, the Voronoi areas of cells near the border may be undefined.[2]

The veins and arteries are manually tracked by Aruna Jammalamadaka based on their branching characteristics using NeuronStudio [60]. The arteries are shown in red and the veins are shown in black. In the original track files, the width of the blood vessels are recorded but the information is ignored in the visualization to reduce visual complexity. In future study, an option to highlight the branching points and the difference in diameters may become useful because of more recent knowledge about the vessels, such as their morphology in optic nerve head drusen [61][3]. When a blood vessel crosses the retina boundary to connect with the opposite side of the cut, a thin dotted line is drawn to provide just enough visual cue of continuity without increasing visual complexity.

The retina boundary, manually created by Gabe Luna using `Retivis`, is drawn with a thin gray line. On most retinas where cells are distributed over the entire retina, this explicit visual representation of the boundary is not needed because it is implied from the locations of the cells. However, cells may be missing near

[2]We consider any Voronoi cells that intersect with the retina boundary to be undefined even if the area is not open.

[3]an abnormal condition in the ONH, also called optic disk drusen. See [62] for more information.

a border of a damage part of some retina, hence the boundary lines are deemed helpful.

To reduce visual complexity, the optic nerve head is not shown.

6.4 Supplemental information

The retina's name is shown on the top-left corner. Simple statistics are shown on the top-right. These include the number of available data points (N), the mean, the median (med), standard deviation (SD), and the number of missing values. Written at the bottom are the min value, the 5^{th} and 95^{th} percentiles, and the max values. The color scheme is also shown, reminding the viewer that blue represents the bottom five percent and red represents the top five percent, and that the colors of the 11 dots represent the range of value in the middle. A series of dots (•), instead of a continuous gradient stripe, are used because they better resemble the data points in the visualization above.

6.5 Multiple retina view

The eight retinas are displayed in eight equal-size square boxes. All retinas are scaled by the same factor. The scale factor is the largest such that all retinas still fit inside the squares. The variable's name is shown at the top.

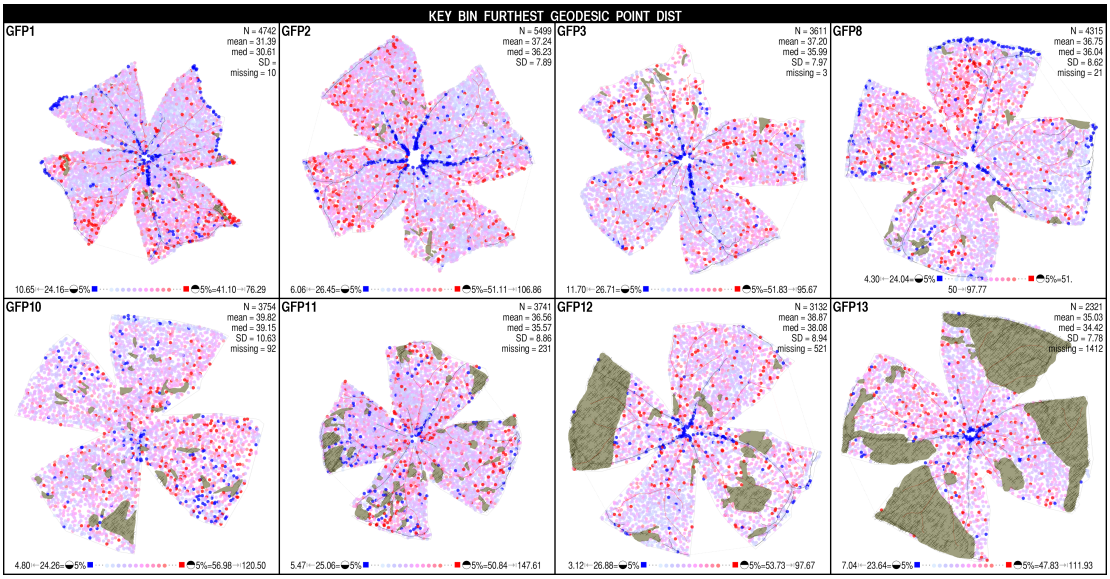


Figure 6.2: A multi-retina view showing geodesic distance to the furthest point. The data points are not pooled together.

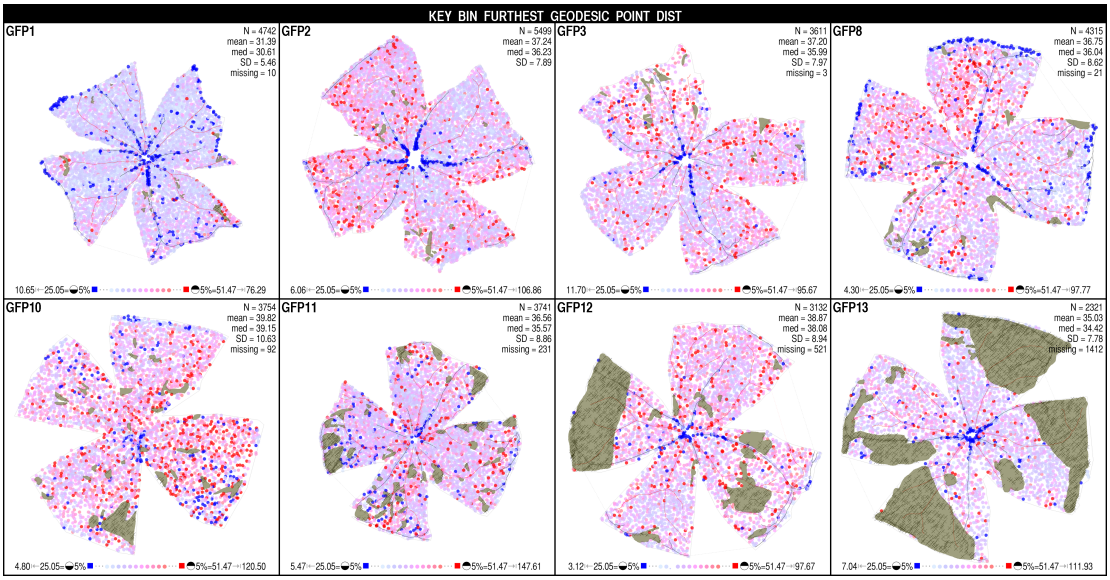


Figure 6.3: A multi-retina view showing geodesic distance to the furthest point. The data points are pooled together.

In Figure 6.2, the visualization for each retina is rendered individually. The color scale for each retina is calculated for the data of that particular retina, independent of other retinas. For this reason, the top five percent and the bottom five percent for each retina are different from that of other retinas, and are reflected in the bottom legends of the squares. Every retina always has some data at the top 5% and some at the bottom 5%, so red and blue dots are always present. This allows for patterns to emerge on each retina regardless of the variation of ranges among the retinas. Essentially, the process is similar to normalizing the values[4] from the eight retinas before visualizing. An advantage of this approach is that even though the visualizations are normalized, the raw data are not. An analyst can still look at the variations in the means, the medians, and standard deviations although such observations are not visual.

Another approach of calculating the color scale is to pool the data together. Instead of rendering each retina individually and independently of each other, all the valid data points from all retinas are first pooled together. The top and bottom 5% of *all* the data are calculated (as opposed to the top and bottom 5% of *each* retina). Only data above and below such amounts are colored in red and blue. The rest of the data are scaled linearly according to the two percentiles.

[4]normalizing of a data set moves its mean to 0 and the standard deviation to 1. This is accomplished by subtracting every data point by the mean and dividing them by the standard deviation, as done in [21].

Therefore, it is possible that a retina may not have any red dots or blue dots. In this view, patterns on some retinas which do not possess extreme values may be difficult to detect. However, variations between retinas will be more visible. An example of this is shown in Figure 6.3.

Chapter 7

Discussion and Future Work

Many trends are more visible in GFP1 and 2 (and to a lesser degree, GFP11) than other retinas. Generally, trends are more visible near the optic nerve head (ONH). GFP1 and 2 are the most densely populated and GFP2 has the biggest cell sizes. Some patterns that appear elsewhere tend to disappear or appear to be random (or even reversed) near, but outside, the damage areas. A possible explanation is that the damage area masks are more restrictive than the reality. Some areas in the vicinity outside the damage mask may also be damaged. Most interesting patterns involve the blood vessel structure. It has proven useful to overlay the veins and the arteries with different colors because the patterns are mostly different.

7.1 Patterns along the veins

We found that the following observations hold along the veins:

- lower Voronoi areas, suggesting that cells are densely populated. See Figure A.3.

- shorter nearest neighbor distances (only weakly visible on some retinas), suggesting that the cells are densely populated. See Figure A.4.

- higher cell center counts in the 150 μm vicinity of each cell, esp for cells near the ONH, strongly suggesting that cells are densely populated. See Figure A.5.

- bigger cells according to weighted cells area (very clear trend). See Figure A.2.

- bigger cells according to adaptive threshold binarization (clear trend). See Figure A.1.

- lower eccentricity (very weak trend), suggesting that cells are shaped like a circle or a blob. See Figure A.6.

- lower geodesic distance to furthest point, suggesting that cells are small and not complex (not having many branches). See Figure A.7.

- lower perimeter (weak signal), suggesting that the cells do not have many branches or holes. See Figure A.8.

The above observations, taken together, suggest that the cells along the veins are bigger than the rest of the retinas in terms of areas and that the areas along the veins are densely packed with cells as observed from the cell center counts, the Voronoi area, and the nearest neighbor distance.

The observation that cells along the veins are large seems to have suggested that they should occupy a large portion of the image. However, the geodesic distances to the furthest point are generally low along the veins, suggesting that it has a low radius, occupying only a small area. Its lower perimeter also points toward smaller cells. These seemingly contradicting trends can be reconciled by observing the lower eccentricity along the veins. Even though the eccentricity trend is very weak, it explicitly suggests that the cells are shaped like circles rather than elongated. One plausible hypothesis is that the cells along the veins are shaped like a blob. This hypothesis is compatible with the trend of low perimeter, suggesting that cells do not have may branches which would have increased the perimeter. The lack of branches also explain the low geodesic distance to the furthest point, which is a clear trend, because a circle is a shape where the distance to the furthest point is minimized for any given cell area.

It remains to be studied whether the hypothesis on the cell shapes is a result of a biological property of cells along the veins or an artifact of the random walk segmentation algorithm in a densely populated area.

7.2 Patterns along the arteries

We found that the following observations hold along the arteries:

- higher Voronoi areas (weak trend) than along veins, suggesting that the areas are sparsely populated. See Figure A.3.
- lower number of cell center counts in the 150 μm vicinity than along veins, also suggesting that the areas are more sparsely populated than along veins. See Figure A.5.
- A mixture of both large and small cells, but more small cells according to adaptive thresholding binarization (Figure A.1) than that from weighted cell area (Figure A.2), possibly suggesting that binarization causes small cells to appear even smaller.
- higher eccentricity (very weak signal), suggesting that cells are elongated. See Figure A.6.
- higher geodesic distance to the furthest point, suggesting that cell processes reach farther. See Figure A.7.

Generally, the trends along the arteries are less visible than those along veins, but still more visible than in other areas of the retinas. Based on this visual observation alone, it is not yet conclusive if the cells along the arteries are bigger

or smaller than over the rest of the retina. But it is known that they are smaller than the cells along the veins.

Cells along the arteries may be sparsely populated according to the lower number of cell counts in the vicinity. But this trend is only weakly supported by the high Voronoi areas, and not supported nor refuted by the lack of any trends regarding the nearest neighbor distance. It may be possible that cells can be sparsely populated but still have low distances to the nearest neighbor. When cells are arranged in a certain way, removing a number of cells from the population may not significantly increase the nearest neighbor distances especially if the removed cells are not nearest neighbors of any other cells.[1]

An interesting observation is that the cells along the arteries have high geodesic distance to the furthest point, and to a lesser degree, there is a pattern of higher eccentricity. This may suggest that the cells have branches of unequal lengths in certain directions. The branches reach out relatively far – resulting in higher geodesic distance, but not equally far in all directions – resulting in higher eccentricity.

A plausible assumption based on these observation is that the cells along arteries are branchy, and the branches reach out in certain directions. It remains to

[1]For example, Distribution A can be much more dense than Distribution B but still have the same nearest neighbor distance. In A, cells are packed on a regular grid of 10 μm apart. In B, all but two cells are removed; the two cells are 10 μm apart. Both have an average nearest neighbor distance of 10 μm.

be studied whether these branches reach out for another blood vessel, or another far-away astrocyte, or another specific structure.

7.3 Future work

It may be beneficial to reconsider some other variables in Table 5.1 for a closer inspection. The hypothesis that cells along the veins are shaped like blobs may warrant a look at `[holes]`, `[hull holes]`, and `[solidity]` which should be low, low, and high, respectively if the hypothesis is true.

The properties of cells along the arteries suggest that they may shape like stars and that they may reach out in certain directions. While we have a variable that represents direction, `[orient]`, it only represents the angle with respect to the horizontal line – an arbitrary construct. Another variable representing the relative direction with respect to some other prominent structures such as a major blood vessel or the ONH should be included for further analysis.

While the shapes of the cells along the veins or arteries may be deduced from different variables such as `[area bin]`, `[furthest pt]`, and `[eccntrcty]`, it may be more convincing to directly and visually confirm the assumption about their shapes. A new visualization technique could be proposed where a random sample of cells can be queried and visualized on a grid (instead of on the retina) based

on certain criteria. For examples, we should be able to view the shapes of cells that are within 20 μm of any vein on one-half of the screen while the other half shows the cells that are outside that region.

Along the veins, cells have large sizes and the areas are densely packed. It may be possible that cells overlap more in this region. Our visualization of a simulated cell network in which we encode the overlap scores as line thickness and opacity, similar to Figure 1.12(c) have suggested that this is true. Another variable, called `[overlap]` could be introduced into the analysis. It should be defined as the number of pixels overlapped with at least one or more neighbors. It remains to be decided whether to count a pixel twice if it overlaps with more than one neighbors. It also should be considered whether to express this amount of overlap as the absolute number of pixels, or as a percentage of the cell size.

An initial exploration of this idea has been attempted. Figure 7.1 shows the location of cells with the most overlaps (assuming non-normalized retinas, random walk segmentation, and adaptive thresholding binarization). The amount shown is the sum of the number of pixels that each cell overlaps with its neighbors. For each retina, the lowest 5% are shown in blue; the top 5% are shown in red; and the rest are shown in colors ranging from very light gray (low value) to black (high value). The high values appear to form a structure similar to the veins. These

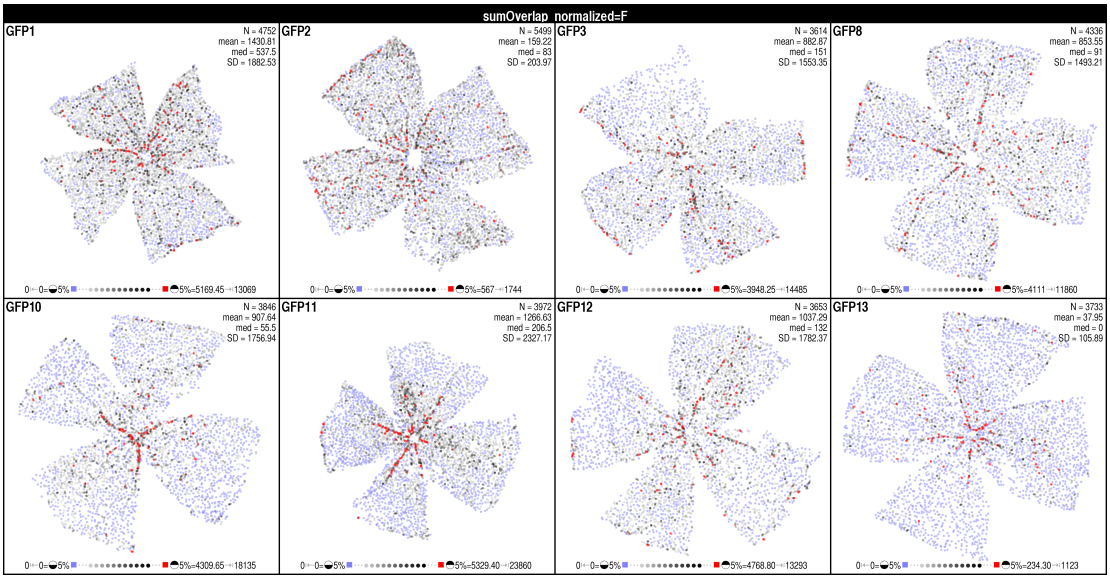


Figure 7.1: Amount of overlaps in non-normalized retinas

are the areas where the pixels are more likely to be counted twice (or more) when `[area bin]` is calculated.

While single variable observations are important and already rich with potential insights, a visual survey of the relationship between variable pairs may be even more interesting. As observed here, the trends are stronger near the ONH or near a major blood vessels. We may be able to detect a relationship between any two variables more quickly when an appropriate third variable (such as `[dist2ctr]`) is used as a filter.

In this thesis, we have proposed a comprehensive visualization system integrated with a highly flexible region-based analysis tool. In close consultation with our collaborating biologists, we developed and applied our toolkit on a large

data set of microscopic image mosaics of retinal astrocytes. Our single cell injection study, the network study, and retina clustering and normalization helped us prepare and process the data. Finally, we presented the final visualization and reported our observation and interpretation. We hope that this thesis may stimulate further work and insights in the field of mammalian retinal astrocyte analysis.

Bibliography

[1] P. Klumphu and B. H. Lipshutz, "nok: A phytosterol-based amphiphile enabling transition-metal-catalyzed couplings in water at room temperature," *The Journal of organic chemistry*, vol. 79, no. 3, pp. 888–900, 2014.

[2] A. P. Jardosh, P. Suwannatat, T. Höllerer, E. M. Belding, and K. C. Almeroth, "Scuba: focus and context for real-time mesh network health diagnosis," in *Passive and Active Network Measurement*, pp. 162–171, Springer, 2008.

[3] Y. Kubota and T. Suda, "Feedback mechanism between blood vessels and astrocytes in retinal vascular development," *Trends in Cardiovascular Medicine*, vol. 19, no. 2, pp. 38 – 43, 2009.

[4] M. Pekny, U. Wilhelmsson, and M. Pekna, "The dual role of astrocyte activation and reactive gliosis," *Neuroscience letters*, vol. 565, pp. 30–38, 2014.

[5] G. Prasanna, R. Krishnamoorthy, and T. Yorio, "Endothelin, astrocytes and glaucoma," *Experimental Eye Research*, vol. In Press, Corrected Proof, 2010.

[6] T. Nakazawa, M. Takeda, G. P. Lewis, K. S. Cho, J. Jiao, U. Wilhelmsson, S. K. Fisher, M. Pekny, D. F. Chen, and J. W. Miller, "Attenuated glial reactions and photoreceptor degeneration after retinal detachment in mice deficient in glial fibrillary acidic protein and vimentin," *Investigative Ophthalmology and Visual Science*, vol. 48, no. 6, pp. 2760–2768, June 2007.

[7] J. Stone and Z. Dreher, "Relationship between astrocytes, ganglion cells and vasculature of the retina," *The Journal of Comparative Neurology*, vol. 255, no. 1, pp. 35 – 49, 1987.

[8] P. Suwannatat, G. Luna, B. Ruttenberg, R. Raviv, G. Lewis, S. K. Fisher, and T. Höllerer, "Interactive visualization of retinal astrocyte images," in *Biomedical Imaging: From Nano to Macro, 2011 IEEE International Symposium on*, pp. 242–245, March 2011.

[9] P. Suwannatat, G. Luna, G. P. Lewis, S. K. Fisher, and T. Höllerer, "Scalable Interactive Analysis of Retinal Astrocyte Networks," *BioVis*, 2012.

[10] M. Brenner, "Role of GFAP in CNS injuries," *Neuroscience letters*, vol. 565, pp. 7–13, 2014.

[11] Mayachitra, "Mayachitra imago — Bioimage management and analysis software." `http://mayachitra.com/imago`, 2012.

[12] Y. Al-Kofahi, W. Lassoued, W. Lee, and B. Roysam, "Improved automatic detection and segmentation of cell nuclei in histopathology images," *Biomedical Engineering, IEEE Transactions on*, vol. 57, no. 4, pp. 841–852, 2010.

[13] J. Byun, M. R. Verardo, B. Sumengen, G. P. Lewis, B. Manjunath, and S. K. Fisher, "Automated tool for the detection of cell nuclei in digital microscopic images: application to retinal images," *Mol Vis*, vol. 12, pp. 949–960, 2006.

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

[15] V. Ljosa and A. K. Singh, "Probabilistic segmentation and analysis of horizontal cells," in *Data Mining, 2006. ICDM'06. Sixth International Conference on*, pp. 980–985, IEEE, 2006.

[16] B. E. Ruttenberg, *Managing and Mining Biological Images*. PhD thesis, University of California at Santa Barbara, Santa Barbara, CA, USA, 2012. AAI3545126.

[17] Wikipedia, "Embarrassingly parallel — wikipedia, the free encyclopedia." `http://en.wikipedia.org/w/index.php?title=Embarrassingly_parallel&oldid=622429327`, 2014. [Online; accessed 26-August-2014].

[18] D. Gunter and B. Tierney, "Netlogger: a toolkit for distributed system performance tuning and debugging," in *Integrated Network Management, 2003. IFIP/IEEE Eighth International Symposium on*, pp. 97–100, March 2003.

[19] M. K. Aguilera, J. C. Mogul, J. L. Wiener, P. Reynolds, and A. Muthitacharoen, "Performance debugging for distributed systems of black boxes," in *Proceedings of the Nineteenth ACM Symposium on Operating Systems Principles*, SOSP '03, (New York, NY, USA), pp. 74–89, ACM, 2003.

[20] P. Suwannatat, "GFP1 segmentation - starting (26s at 112X)." `http://youtu.be/bsvgSun8eS8`, 2014. [Online; accessed 26-August-2014].

[21] A. Jammalamadaka, *Spatial Pattern Modeling and Discovery in Biological Images*. PhD thesis, University of California at Santa Barbara, Santa Barbara, CA, USA, 2014.

[22] K. Jiang, Q.-M. Liao, and S.-Y. Dai, "A novel white blood cell segmentation scheme using scale-space filtering and watershed clustering," in *Machine Learning and Cybernetics, 2003 International Conference on*, vol. 5, pp. 2820–2825, IEEE, 2003.

[23] N. Malpica, C. Ortiz de Solorzano, J. J. Vaquero, A. Santos, I. Vallcorba, J. M. Garcia-Sagredo, and F. d. Pozo, "Applying watershed algorithms to the segmentation of clustered nuclei," 1997.

[24] P. Umesh Adiga and B. Chaudhuri, "An efficient method based on watershed and rule-based merging for segmentation of 3-d histo-pathological images," *Pattern Recognition*, vol. 34, no. 7, pp. 1449–1458, 2001.

[25] C. Healey, "Choosing effective colours for data visualization," in *Visualization '96. Proceedings.*, pp. 263–270, Oct 1996.

[26] A. Saad, T. Mller, and G. Hamarneh, "Probexplorer: Uncertainty-guided exploration and editing of probabilistic medical image segmentation," *Computer Graphics Forum*, vol. 29, no. 3, pp. 1113–1122, 2010.

[27] J.-S. Prassni, T. Ropinski, and K. Hinrichs, "Uncertainty-aware guided volume segmentation," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 16, pp. 1358–1365, Nov 2010.

[28] A. Saad, G. Hamarneh, and T. Moller, "Exploration and visualization of segmentation uncertainty using shape and appearance prior information," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 16, pp. 1366–1375, Nov 2010.

[29] E. H. Adelson, C. H. Anderson, J. R. Bergen, P. J. Burt, and J. M. Ogden, "Pyramid methods in image processing," *RCA engineer*, vol. 29, no. 6, pp. 33–41, 1984.

[30] S. Oaks and H. Wong, *Java Threads*. Nutshell handbooks, O'Reilly Media, 2004.

[31] Y. Ling, T. Mullen, and X. Lin, "Analysis of Optimal Thread Pool Size," *SIGOPS Oper. Syst. Rev.*, vol. 34, pp. 42–55, Apr. 2000.

[32] J. Kaplan, "matlabcontrol-A Java API to interact with MATLAB." `http://code.google.com/p/matlabcontrol`, 2011.

[33] S. Urbanek, "Rserve - Binary R server." `http://rforge.net/Rserve`, 2013.

[34] S. Urbanek, "REngine - Java interface to R." `https://github.com/s-u/REngine`, 2007.

[35] V. Solutions, "JTS Topology Suite." `http://www.vividsolutions.com/jts`, 2006.

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

[37] J. Maurer, C.R., R. Qi, and V. Raghavan, "A linear time algorithm for computing exact euclidean distance transforms of binary images in arbitrary dimensions," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 25, pp. 265–270, Feb 2003.

[38] MathWorks, "Measure properties of image regions - MATLAB regionprops." `http://www.mathworks.com/help/images/ref/regionprops.html`, 2014. [Online; accessed 26-August-2014].

[39] R. Unnikrishnan, C. Pantofaru, and M. Hebert, "A measure for objective evaluation of image segmentation algorithms," in *Computer Vision and Pattern Recognition-Workshops, 2005. CVPR Workshops. IEEE Computer Society Conference on*, pp. 34–34, IEEE, 2005.

[40] E. D. Gelasca, J. Byun, B. Obara, and B. Manjunath, "Evaluation and benchmark for biological image segmentation," in *Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on*, pp. 1816–1819, IEEE, 2008.

[41] Z. Yin, R. Bise, M. Chen, and T. Kanade, "Cell segmentation in microscopy imagery using a bag of local bayesian classifiers," in *Biomedical Imaging: From Nano to Macro, 2010 IEEE International Symposium on*, pp. 125–128, IEEE, 2010.

[42] L. P. Coelho, A. Shariff, and R. F. Murphy, "Nuclear segmentation in microscope cell images: a hand-segmented dataset and comparison of algorithms," in *Biomedical Imaging: From Nano to Macro, 2009. ISBI'09. IEEE International Symposium on*, pp. 518–521, IEEE, 2009.

[43] B. Yao, X. Yang, and S.-C. Zhu, "Introduction to a large-scale general purpose ground truth database: Methodology, annotation tool and benchmarks," in *Energy Minimization Methods in Computer Vision and Pattern Recognition* (A. Yuille, S.-C. Zhu, D. Cremers, and Y. Wang, eds.), vol. 4679 of *Lecture Notes in Computer Science*, pp. 169–183, Springer Berlin Heidelberg, 2007.

[44] J. Davis and M. Goadrich, "The relationship between precision-recall and roc curves," in *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, (New York, NY, USA), pp. 233–240, ACM, 2006.

[45] Wikipedia, "Precision and recall — wikipedia, the free encyclopedia." `http://en.wikipedia.org/w/index.php?title=Precision_and_recall&oldid=624435821`, 2014. [Online; accessed 6-September-2014].

[46] C. Goutte and E. Gaussier, "A probabilistic interpretation of precision, recall and f-score, with implication for evaluation," in *Advances in Information Retrieval* (D. Losada and J. Fernndez-Luna, eds.), vol. 3408 of *Lecture Notes in Computer Science*, pp. 345–359, Springer Berlin Heidelberg, 2005.

[47] D. M. Powers, "Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation," 2011.

[48] J. Livet, T. A. Weissman, H. Kang, R. W. Draft, J. Lu, R. A. Bennis, J. R. Sanes, and J. W. Lichtman, "Transgenic strategies for combinatorial expression of fluorescent proteins in the nervous system," *Nature*, vol. 450, no. 7166, pp. 56–62, 2007.

[49] P. Mahou, M. Zimmerley, K. Loulier, K. S. Matho, G. Labroille, X. Morin, W. Supatto, J. Livet, D. Débarre, and E. Beaurepaire, "Multicolor two-photon tissue imaging by wavelength mixing," *Nature methods*, vol. 9, no. 8, pp. 815–818, 2012.

[50] P. Suwannatat, "Thresholds required for coverage of retinas." `http://youtu.be/Hbm6dvQnwS4`, 2014. [Online; accessed 26-August-2014].

[51] P. Suwannatat, "Graphs created by the thresholds required for certain coverage." `http://youtu.be/We_SheeDyyM`, 2014. [Online; accessed 26-August-2014].

[52] M. Ichino and H. Yaguchi, "Generalized minkowski metrics for mixed feature-type data analysis," *Systems, Man and Cybernetics, IEEE Transactions on*, vol. 24, pp. 698–708, Apr 1994.

[53] Y. Rubner, C. Tomasi, and L. J. Guibas, "The earth mover's distance as a metric for image retrieval," *International Journal of Computer Vision*, vol. 40, no. 2, pp. 99–121, 2000.

[54] R. Brunelli and O. Mich, "Histograms analysis for image retrieval," *Pattern Recognition*, vol. 34, no. 8, pp. 1625 – 1637, 2001.

[55] Wikipedia, "Minkowski distance — wikipedia, the free encyclopedia." `http://en.wikipedia.org/w/index.php?title=Minkowski_distance&oldid=620379123`, 2014. [Online; accessed 4-September-2014].

[56] G. Xiong, "Local adaptive thresholding - file exchange - matlab central." `http://www.mathworks.com/matlabcentral/fileexchange/8647-local-adaptive-thresholding`, 2006. [Online; accessed 26-August-2014].

[57] S. M. Pizer, E. P. Amburn, J. D. Austin, R. Cromartie, A. Geselowitz, T. Greer, B. ter Haar Romeny, J. B. Zimmerman, and K. Zuiderveld, "Adaptive histogram equalization and its variations," *Computer Vision, Graphics, and Image Processing*, vol. 39, no. 3, pp. 355 – 368, 1987.

[58] J. Stark, "Adaptive image contrast enhancement using generalizations of histogram equalization," *Image Processing, IEEE Transactions on*, vol. 9, pp. 889–896, May 2000.

[59] J.-Y. Kim, L.-S. Kim, and S.-H. Hwang, "An advanced contrast enhancement using partially overlapped sub-block histogram equalization," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 11, pp. 475–484, Apr 2001.

[60] A. Rodriguez, D. B. Ehlenberger, D. L. Dickstein, P. R. Hof, and S. L. Wearne, "Automated three-dimensional detection and shape classification of dendritic spines from fluorescence microscopy images," *PloS one*, vol. 3, no. 4, p. e1997, 2008.

[61] A. V. Pilat, F. A. Proudlock, R. J. McLean, M. C. Lawden, and I. Gottlob, "Morphology of Retinal Vessels in Patients With Optic Nerve Head Drusen and Optic Disc Edema," *Investigative ophthalmology & visual science*, vol. 55, no. 6, pp. 3484–3490, 2014.

[62] C. Auw-Haedrich, F. Staubach, and H. Witschel, "Optic disk drusen," *Survey of Ophthalmology*, vol. 47, no. 6, pp. 515 – 532, 2002.

Appendices

Appendix A

Details on specific variables

We report the visualization results of eight of the nine variables along with their histograms[1]. For each variable, we show the visualization of values after the retina has been normalized. Adaptive thresholding [56] is used to binarize the cell segmentation results. The cells in the masked damage areas are excluded. To bring out the patterns more clearly, the data is not pooled together – i.e. each retina is rendered individually using its own color scale.

To prepare this report, an automated script[2] organizes the multiple-retina visualizations of each variable – for both pooled and non-pooled versions, the histograms, the comparison matrix with respect to the variable, and the alternative results for non-normalized retinas – and presents them as cross-linked web pages[3].

We observed each variable visually, looking for trends of low and high values regarding any structures, and recorded our observations to a text file, one text file for each variable[4]. The human's observation records are then integrated back on to the auto-report web pages by re-running the same `reportKeys` script. The visualizations and the observations can then be viewed as an integrated report. Based on this, we were able to refine and add further insights into the observation text, especially on the relationship between related variables (e.g. about cell shapes). The `reportKeys` script was re-run after each update to bring the online reports up to date. Based on the final reading of the detailed reports, we produced a narration that highlights the findings and potential insights as reported in Chapter 7.

[1] We omitted [dist2ctr] (the distance from the optic nerve head to the cell center) because it is used primarily to validate and demonstrate the color scheme. While a correlation analysis of this variable against another variable is important, it is not interesting as a stand-alone variable.

[2] The script is run by `./retivis macro=reportKeys`

[3] Available at `http://goo.gl/rXt4Nx`

[4] Located at the directory `AstrocyteRoot/info/observationText`

A.1 Variable: [area bin]

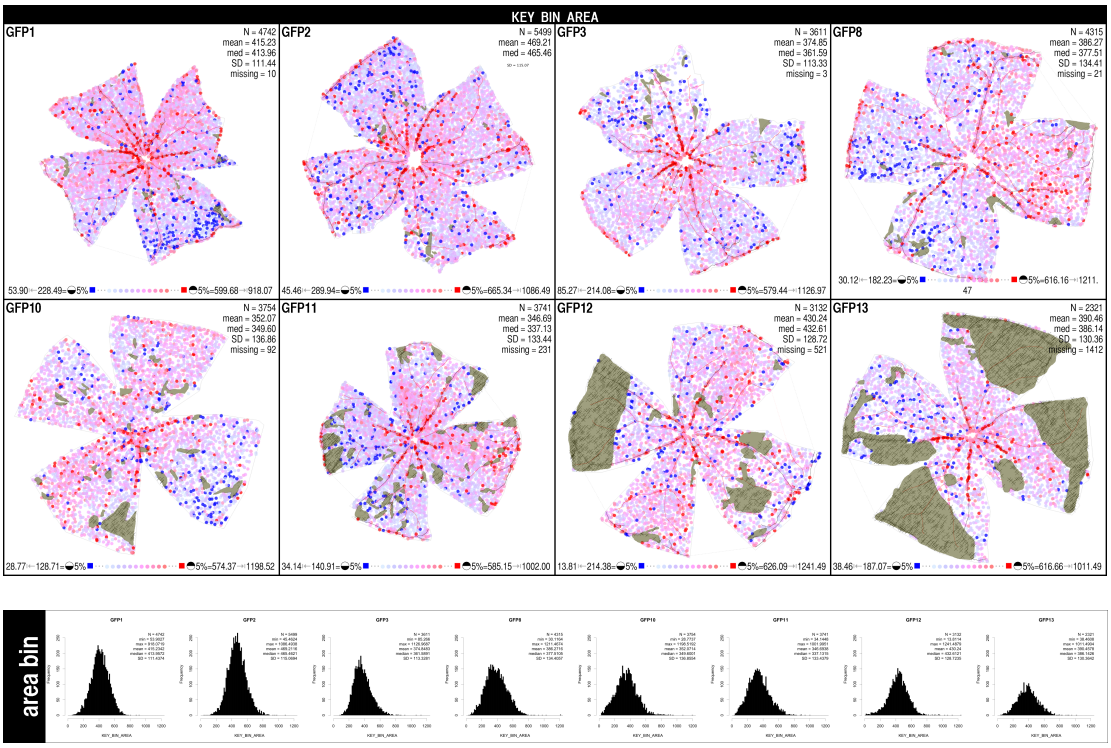


Figure A.1: Visualization and histograms of [area bin]
Technical name: `key_bin_area`

- This is the areas of cells after binarization with adaptive thresholding.
- Near the veins, the trend is clear that cells are the largest. This trend is the same as, but a little less clear than, that found in the weighted cell areas.
- Near the arteries, there also seems to be a trend, but the trend is ambiguous. There are both red dots and blue dots. We see more blue dots (small cells) near arteries here in binarized area than in the weighted grayscale area version. It may suggest that adaptive thresholding binarization causes small cells near the arteries to appear even smaller (hence the existence of more red cells).
- GFP2 still has the biggest cells, according to the pooled version where we see more red dots in GFP2 than in other retinas, and the vertical arrangement of histograms. The means and medians also confirm this.

A.2 Variable: [area gray]

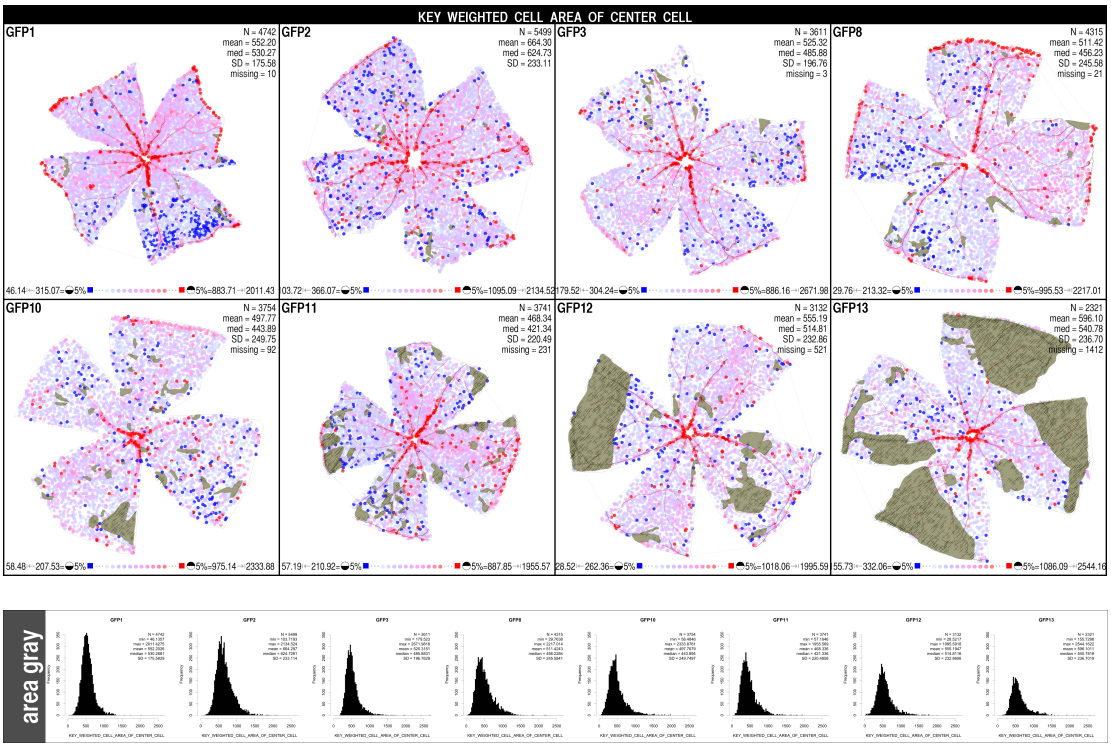


Figure A.2: Visualization and histograms of [area gray]
Technical name: `key_weighted_cell_area_of_center_cell`

- This is the weighted area of cell, meaning the sum of the intensity values for all the pixels in the grayscale segmentation result. This value does not depend on a binarization method or threshold.
- Both in the non-pooled and pooled version, we see a clear trend that red is hugging black. This means the biggest cells live along the veins.
- For the arteries (red lines), the trend is much less clear. From just these visualizations, we cannot say that smallest cells live near the arteries. Nor can we say that cells along the arteries are bigger than the general area although we seem to see that pattern in a few places, its not conclusive.
- The pooled version shows that GFP2 has the biggest cells (along the veins). A vertical arrangement of histograms, and a reading of means and medians, shows that as well.

A.3 Variable: [area voro]

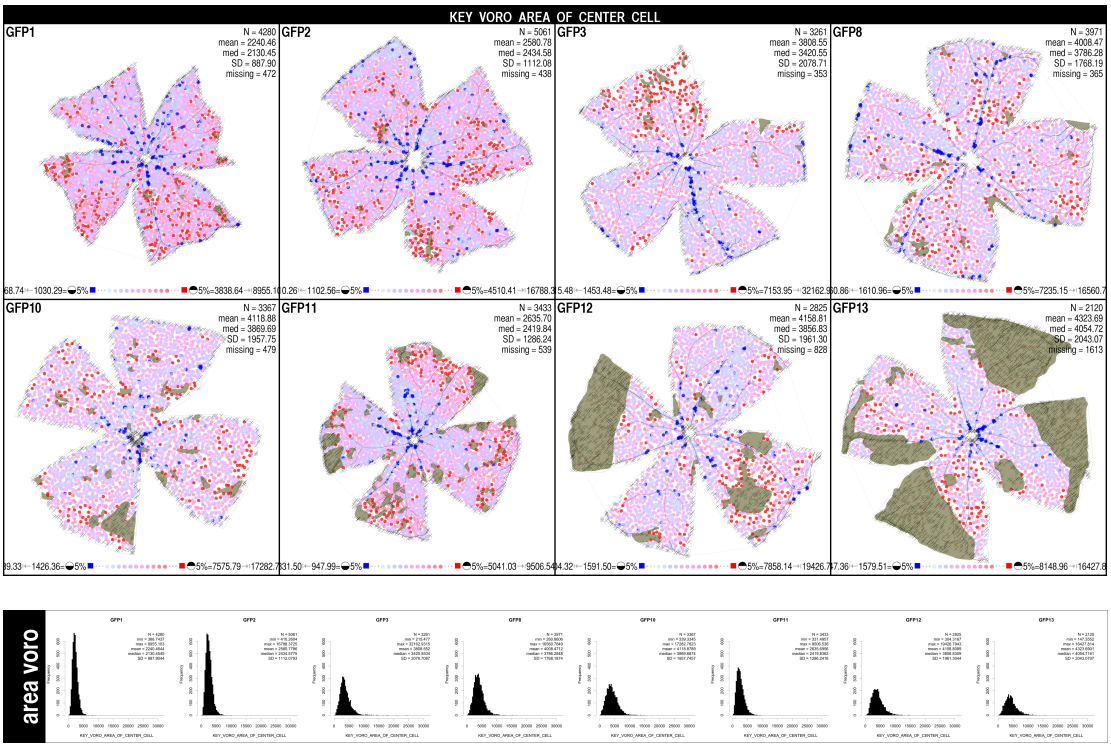


Figure A.3: Visualization and histograms of [area voro]
Technical name: `key_voro_area_of_center_cell`

- This is the area of the Voronoi region around cell center. High values (red) mean big Voronoi regions, implying less dense area. Low values (blue) means more dense area.
- We see blue hugging veins (black lines). That suggests that the areas around veins are dense.
- To a lesser degree, reds are hugging arteries (red lines). This suggests that the areas around arteries are less dense.
- When we pool data together, the trend of blue hugging veins become even more visible for GFP1 and 2.

A.4 Variable: [near NB]

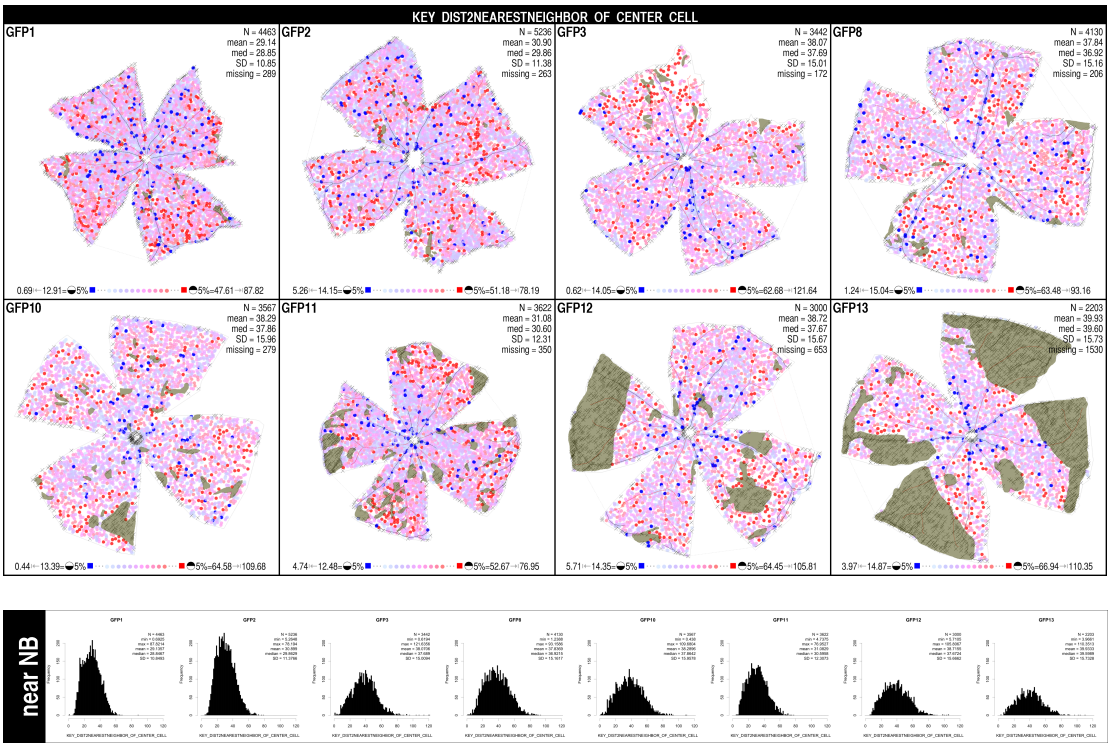


Figure A.4: Visualization and histograms of [near NB]
Technical name: `key_dist2nearestneighbor_of_center_cell`

- This is the distance to nearest neighbor.
- Intuitively, this should correlate well with Voronoi area but it doesnt (at least visually). The trend is not as clear as Voronoi area.
- GFP1 and GFP2 almost do not show any trend at all. This might be because they are the most dense retinas (as seen from the pooled version). When the density is high enough to a point, the nearest neighbor distance are always low and vary very little from region to region. The SDs (standard deviation) support this statement: we see that the SDs for GFP1 and GFP2 are around 11 whereas other retinas have higher SDs around 15.
- In GFP8, 11, 12, and 13, we see a weak trend that blue dots are hugging black lines. Red lines sometimes also have blue dots, but sometimes have red dots. This suggests that areas around veins are dense. For arteries, the conclusion is not clear.

- Histograms of GFP2 and GFP13 are the most different, possibly because of the high number of missing cells from GFP13 (1530 cells).

A.5 Variable: [# centers]

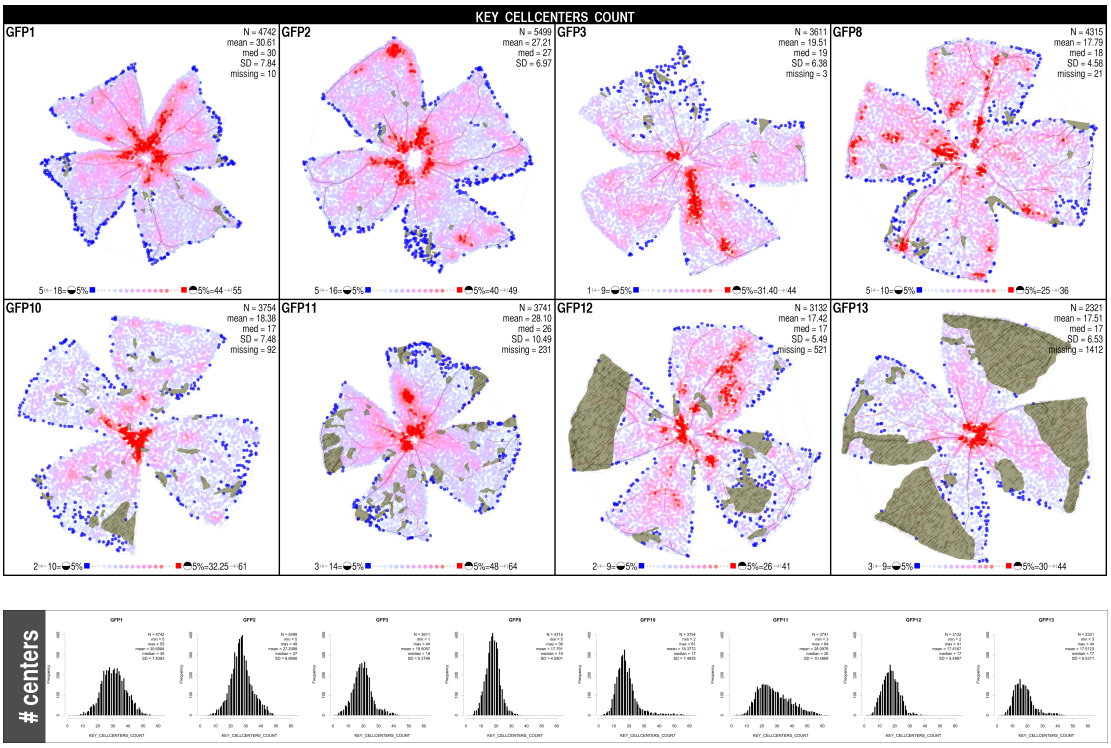


Figure A.5: Visualization and histograms of [# centers]
Technical name: `key_cellcenters_count`

- This represents the density of cell centers. It is the number of cell centers within 150 micron radius of the cell center. The higher number (red), the more dense the area.
- Seemingly, the areas around veins have high density of cells.
- Arteries seem to have lower density (but on some retinas, still higher than general areas)
- The pooled version shows that GFP1, 2, and 11 have the most dense area.
- GFP3 has a very low populated area near the top around the neighborhood of many damage areas. The same situation applies for GFP12 in the lower right wing, and GFP13 left wing.

A.6 Variable: [eccntrcty]

Figure A.6: Visualization and histograms of [eccntrcty]
Technical name: `key_bin_eccentricity`

- According to MATLAB manual, "the eccentricity is the ratio of the distance between the foci of the ellipse and its major axis length." The ellipse here means an "ellipse that has the same second-moments as the region."
- Low value = blue = resembling a circle. High value = red = elongated, resembling a line segment.
- No clear trend is visible either with the pooled or the non-pooled version.
- Some veins have a very faint trend toward blue (circle-like) e.g. GFP13 lower branch. This may be because veins have high density. And when density is high, random walk segmentation sees a lot of green and segments the cell like a blob; or it can be a biological property.
- Some arteries have a very faint trend toward red (elongated) e.g. GFP12 upper branch.

- Visually the histograms for all retinas appear generally similar to one another. GFP2 and 13 are the most different and thats due to the difference in the number of usable cells, not because of eccentricity.

A.7 Variable: [furthest pt]

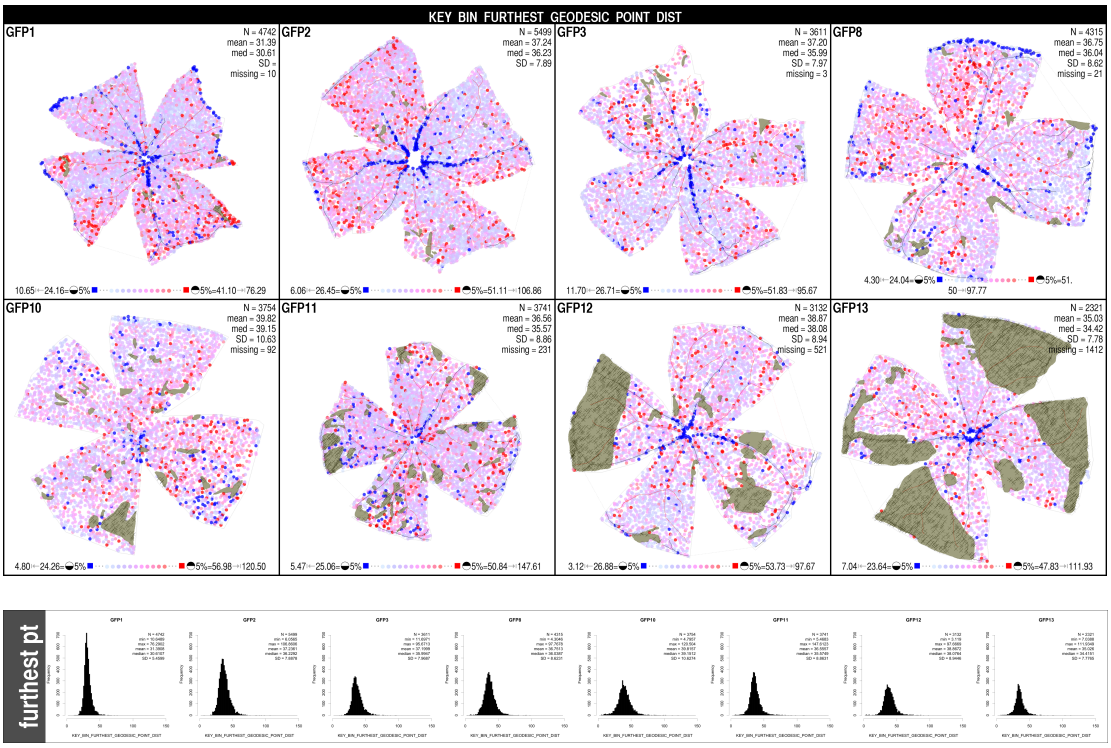


Figure A.7: Visualization and histograms of [furthest pt]
Technical name: `key_bin_furthest_geodesic_point_dist`

- This is the distance from the cell center to the furthest point on the binarized cell body. It is not a Euclidean distance, but rather a distance of walking along the cell (geodesic distance).
- Blue = short distance = small and uncomplex cell. Red = long distance = large or complex cells.
- There is a clear trend of blue hugging the black lines (veins) in all retinas except GFP10 and 11. This suggests cells near the veins are small and uncomplex. This suggestion that cells near veins are small contradict with observation about cell areas (both grayscale and binarized). But we have observed that cells near veins have low eccentricity. That is, they are circle-like. This suggests that cells along the veins are shaped like blobs. They have high areas and short radius.

- We see some trend that cells near arteries have high distance to the furthest point. This means that their branch reach out further despite having less overall area than cells near veins.
- Pooling the data together have almost no effect on the visibility of the trends.

A.8 Variable: [perimeter]

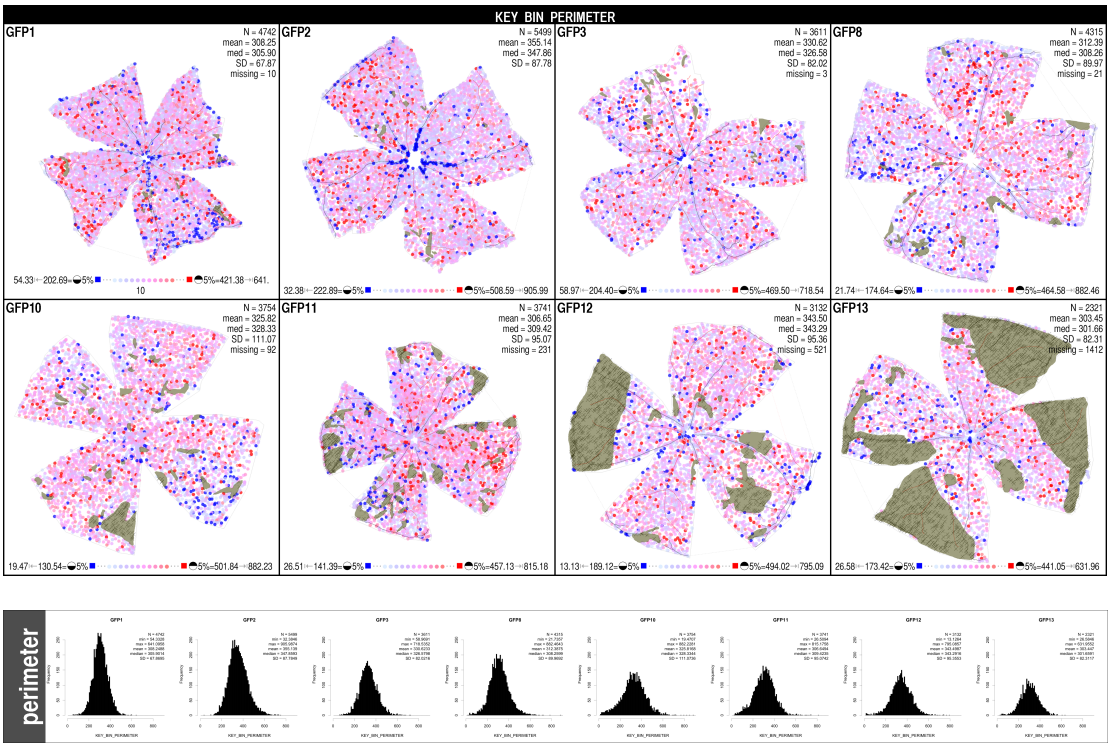


Figure A.8: Visualization and histograms of [perimeter]
Technical name: `key_bin_perimeter`

- This is the perimeter of the binarized cells.
- We see almost no trend except for GFP1 and GFP2.
- In GFP1 and GFP2, we see some trend of blue hugging black. This may suggest that cells along the veins have low perimeter. This is compatible with the assumption that these cells are shaped like blobs rather than stars.
- When we pool the data together, no trend is visible at all.