Autonomous Universal Optic Nerve Axon Delineation from High Resolution Digitized Samples

Jeff K. Meunier

Department of Electrical Engineering and Computer Science,

Case Western Reserve University, Cleveland, OH, Email: jkm26@cwru.edu

Abstract

This paper presents a universal algorithm designed to delineate and count axioms precisely from digitized optic nerve cross-sections. The algorithm is designed to be used with any sized image and segments the work so it can be executed simultaneously on multiple computers and/or processors. With the application of filtering, histogram adjustments, edge detection, and size masking we are able to detect cellular structures in the image precisely while removing staining artifacts. A simple methodology is demonstrated in counting cells distorted with myelin-stained irregularities and its application to multiple samples in a production environment.

KEYWORDS

Optic nerve, Imaging, Canny, Otsu, Automated, Axon

INTRODUCTION

As medical technology advances, even minor tasks such as universal cell segmentation become incredibly complex. New scanning technologies allow us to see cross-sections of optic nerves in incredible detail outside a laboratory environment. A single nerve can have an enormous amount of cells that can take incredibly long time to count by hand.

With the use of computer technology, this task can be automated and produce results almost immediately. The task of mapping these nerves accurately becomes interdependent on high performance software and hardware. While high performance hardware and semi-automated counting software is commercially available [1], automated software is still in its developmental stages. This algorithm addresses the need for a universal cell segmentation package that can be customized to a specific application easily regardless of the digitized sample's size or complexity. In terms of execution time, each image segment can be processed independently in separate threads, and distributed to multiple processors or to a networked cluster of systems to increase productivity when multiples samples are processed.

ALGORITHM

1. The Input image is converted to a grayscale intensity image by eliminating hue and saturation information while retaining luminance [2].
2. The Image from (1) is segmented into multiple slices to reduce memory allocation requirements during processing.
3. The image segment from (2) is top-hat filtered to correct for uneven background illumination [3].
4. The image segment from (2) is added to the top-hat filtered image segment from (3).
5. The image segment from (4) is then subtracted from a bottom-hat filtered image segment from (2).
6. The filters from (3), (4), and (5) are cascaded together and run on the image in multiple epics depending on luminance.
7. The image from (6) is then further segmented into contextual regions.
8. A histogram is then constructed on each region from (7), and the gray levels are remapped so the histogram is flat.
9. The processed contextual regions from (8) are recombined to complete the contrasted limited adaptive histogram equalization (CLHE) [4].
10. The recombined image segment from (9) is converted to a binary image based on a global threshold using Otsu's method [5].
11. Background pixels from (10) that are inaccessible from the edge of the image are then filled.
12. Image from (11) is than eroded and dilated using the same parameters.
13. Small connected components in the resulting image segment from (12) are removed.
14. Cell edges from (13) are detected using a canny edge detector or by finding foreground pixels that are 4-connected to background pixels.
15. Improperly sized artifacts from (14) are masked out of the detection for each image segment.
16. The number of good cells from (15) is counted for each segment.
17. The masked image segment from (15) is recombined with the previously processed image segments and the total number of cells from (16) is added from all previous segments cell count itineration's resulting in a grand total cell count.

RESULTS AND DISCUSSION

Our primary goal was to delineate captured optic nerve cross-sections and extract the axioms precisely from the background. Due to memory limitations we must process slices of the image individually and recombine them to form our final delineated image. This method is advantageous because failures in the myelin-stained background removal are easily detectable (Figure 1) and those slices can be re-run by resizing the disk shaped structuring element in the cascading image filter.

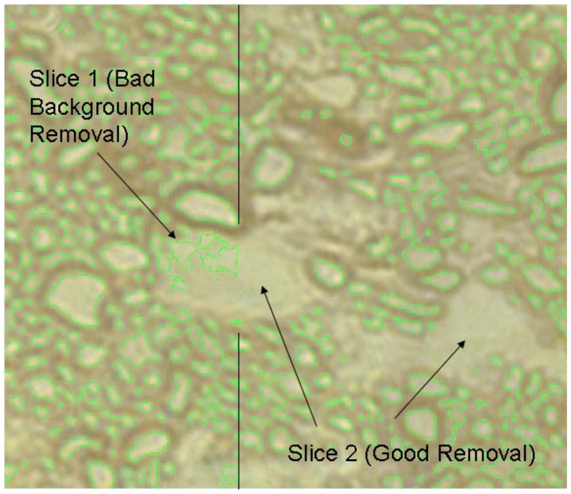

Figure 1: Background removal comparison

One limitation of the algorithm is when CLHE improperly segments a contextual region. This results in a cell or a group of cells with faded contrast and adds to detection difficulty. This is especially pertinent when the cells are near an area of background with artifacts. Improper contextual region segmentation was very rare on all the samples that were run and was detectable. The particular image segment that has a detected CLHE anomaly can be simply rerun after adjusting the cascaded image filter.

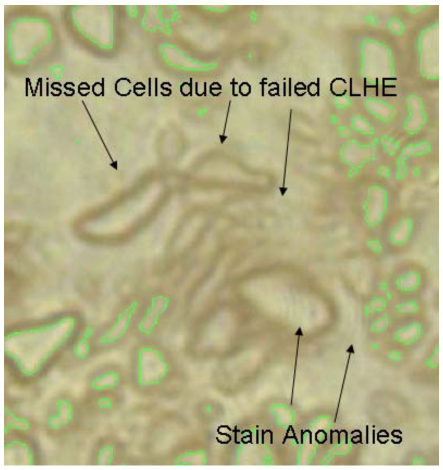

Figure 2: Overly aggressive background removal

Another limitation is oversized cells slipping through the background artifact removal filter as shown in Figure 3. The filter is optimized for searching for cells with a height of 61.6 pixels or less. Originally, extremely small cells with a radius of 1 pixels or less were filtered out. I found that if I filled all edge inaccessible background pixels [6], including extremely small cells in the filter, it helped compensate for some CLHE failures. It also allowed the complete removal of the undersized cell filter, which significantly decreased the algorithms execution time, and helped with the detection of faint cells. If there are a large number of stain anomalies the undersized cell filter is a necessity.

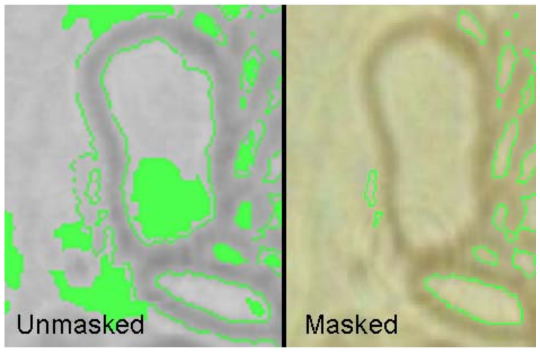

Figure 3: Detection failures due to oversized cell

Six gold standard samples were also processed to check the algorithm's counting accuracy. While these images are extremely low resolution and represent a worst case scenario for image quality, the detections were quite accurate in all but a single sample. Sample Five, in the top left of Figure 4, has severely blurred areas which don't compute well, and the gold standard sample in the bottom left of Figure 4 neglects to take some cells which are off the slide into account while it counts others. In the same figure, sample Six on the right has much better detections due to the more uniform background. Detection was optimal with this background using a structured element of size 30 and rejecting connected pixel sizes less then 0.04 in height.

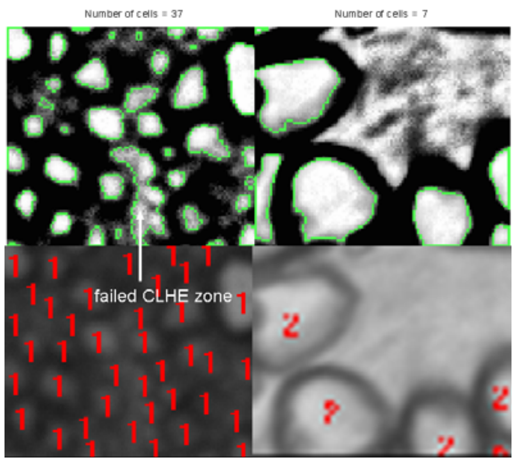

Figure 4: Samples 5 and 6 respectively

Sample One, in the top left of Figure 5, has a cluster of cells in the bottom center whose background luminance levels are significantly different from the rest of the image. The gold standard sample in the bottom left of Figure 5 also neglects to take some cells which are off the slide into account while it counts others. In Figure 5, sample two on the right, has much better detections due to the more uniform background. Detection was optimal with this background using a structured element of size 20 and rejecting connected sizes less then 0.04 pixels in height. The upper left detection in sample two is an off page cell, but the bottom right detection has the potential to be an off page cell as well. However, its surrounding background is eroded too aggressively.

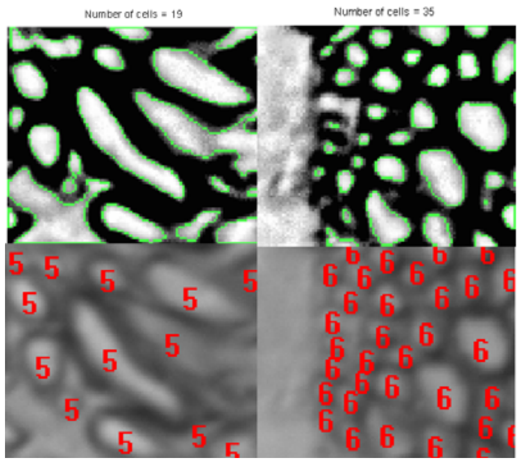

Figure 5: Samples 1 and 2 respectively

Sample Three, in the top left of Figure 6, has a fairly even background and its detections compute well. The gold standard sample in the bottom left of Figure 6 neglects to take some cells which are off the slide into account as in previous samples. Detection was optimal with this background using a bottom structured element of size 190, and top element of size 1 and rejecting connected sizes less then 2.2 pixels in height. In Figure 6, sample four on the right has nearly perfect detections including the area in the bottom left. While the large area of this detected cluster is an artifact and smaller area is an off page cell, it is a valid count considering the cluster is only detect as one cell.

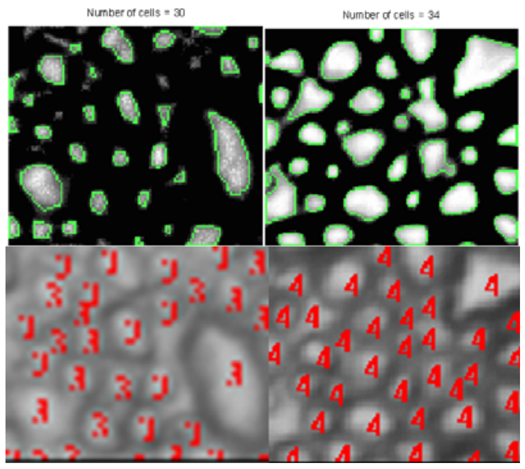

Figure 6: Samples 3 and 4 respectively

SUMMARY

All samples and the full sized mosaic were initially run through the algorithm with the same universal parameters in order to verify its ability to handle any scenario. In the full sized color image mosaic the algorithm routinely found 57,080 cells using the perimeter edge detector. Although this large image was broken into smaller segments, unused variables had to be constantly removed from memory to avoid any overruns. The universal algorithm used static parameters for its cascaded filter and minimum sized cell masking was set to zero. The universal cascaded filter utilized a flat disk-shaped structuring element [7] with a radius of 20.

With these universal settings the absolute deviation from the number of detections with the algorithm was quite low, as seen in Table 1. Perimeter edge detection was less aggressive with stain anomalies and was faster and more accurate than the Canny edge detection method. On sample number 5 the Canny edge detector took 0.333 seconds* to complete while the Perimeter edge detector took only 0.044 seconds* of execution time. If execution time is not an issue, a more advanced custom morphological edge detector can be used to produce a better result [8].

Table 1: Universal Detection results using filter elements of size 20 and no minimum sized cell masking

Sample #	1	2	3	4	5	6
# of Cells (gold Standard)	40	5	30	36	13	33
Perimeter Edge Detector	37	7	28	34	19	34
Absolute Deviation	3	2	2	2	6	1
Canny Edge Detector	37	9	31	35	21	36
Absolute Deviation	3	4	1	1	8	3

The algorithm was tested using dynamic parameters for the top hat and bottom hat filters. Cell sizes below minimum dynamic thresholds are also masked out. This is useful for off page anomalies that did not fit the typical cell size archetype. The dynamic approach resulted in more accurate cell detections and a better total count as shown in Table 2. Computation times were slower when using a minimum cell size greater than zero and significantly slower when using large filter elements. Sample number 3 used a very large bottom hat filter element which, when combined with the small top hat filter element, took a full 3.584 seconds* for 2 itinerations. This is incredibly long when compared with the universal detection size of 20 for both combined filters that takes only 0.18 seconds* of execution time for 2 itinerations.

Table 2: Detection results using dynamic minimum cell sizes and filter elements

Sample #	1	2	3	4	5	6
# of Cells (gold Standard)	40	5	30	36	13	33
Perimeter Edge Detector	37	7	30	34	19	35
Absolute Deviation	3	2	0	2	6	2
Top Hat Filter Element Size	20	20	1	20	20	30
Bottom Hat Filter Element Size	20	20	190	20	20	30
Min Cell Size Used	0.04	0	2.2	0	0	0.04

Regardless of a digitized sample's size or complexity, universal detection is viable. Once the parameters and methodologies are standardized for a set of samples, their digitization will result in samples with homogenous histograms for their backgrounds and foregrounds. The algorithm can then be optimized by a trained observer by manipulating three simple parameters shown in Table 2. Once optimized, an entire batch of samples can be counted fully autonomously with the utmost accuracy.

ACKNOWLEDGMENTS
This Algorithm was developed from a strong foundation of various image processing methodologies learned in an EECS 490 Class taught by Dr. Frank Merat. Without Dr. Merat's insightful lectures and course work none of this would have been possible.

REFERENCES

[1] Reynaud, J., G. Cull, L. Wang, C. F. Burgoyne, and G. A. Cioffi. "A New Hybrid Algorithm for Automated Axon Counting in Normal Optic Nerves." Devers Eye Institute, 2007. Abstract. OASIS 3297 (2007): 8311.

[2] "Image Processing Toolbox Documentation." The Mathworks Website. 5 Nov. 2007 <http://www.mathworks.com/access/helpdesk/help/toolbox/images/rgb2gray.htmll>.

[3] "Image Processing Toolbox Documentation." The Mathworks Website. 5 Nov. 2007 <http://www.mathworks.com/access/helpdesk/help/toolbox/images/imtophat.html>.

[4] "Image Processing Toolbox Documentation." The Mathworks Website. 5 Nov. 2007 <http://www.mathworks.com/access/helpdesk/help/toolbox/images/adapthisteq.html>.

[5] "Image Processing Toolbox Documentation." The Mathworks Website. 5 Nov. 2007 <http://www.mathworks.com/access/helpdesk/help/toolbox/images/graythresh.html>.

[6] "Image Processing Toolbox Documentation." The Mathworks Website. 5 Nov. 2007 <http://www.mathworks.com/access/helpdesk/help/toolbox/images/imfill.html>.

[7] "Image Processing Toolbox Documentation." The Mathworks Website. 5 Nov. 2007 <http://www.mathworks.com/access/helpdesk/help/toolbox/images/strel.htmll>.

[8] Nain, Neeta, Vijay Laxmi, Ankur K. Jain, and Rakesh Agarwal. MORPHOLOGICAL EDGE DETECTION AND CORNER DETECTION ALGORITHM USING CHAIN-ENCODING. The 2006 International Conference on Image Processing, Computer Vision, & Pattern Recognition, June 2006, Malaviya National Institute of Technology. 5 Nov. 2007 <http://ww1.ucmss.com/books/LFS/CSREA2006/IPC4042.pdf>.

*Execution times based on an Intel Core 2 Duo 2.16 GHz CPU with 2GB DDR2 667MHz SDRAM running MATLAB R2007a