A Novel Method for Automated Cell Image Selection

Shuman Guo

Follow this and additional works at: http://scholarworks.gsu.edu/math_theses

Recommended Citation
http://scholarworks.gsu.edu/math_theses/119

This Thesis is brought to you for free and open access by the Department of Mathematics and Statistics at ScholarWorks @ Georgia State University. It has been accepted for inclusion in Mathematics Theses by an authorized administrator of ScholarWorks @ Georgia State University. For more information, please contact scholarworks@gsu.edu.
A NOVEL METHOD FOR AUTOMATED CELL IMAGE SELECTION

by

SHUMAN GUO

Under the Direction of Yuanhui Xiao

ABSTRACT

Retinal pigment epithelium (RPE) is a key site of pathogenesis of age-related macular degeneration (AMD). A key first step toward developing statistical quantifications of RPE morphology is image analysis of RPE flatmount. This thesis work aims to facilitate image analysis by developing a procedure for automated selecting regions with biological information from flatmount images. Our new approach, based on clustering analysis, can extract informative regions from a typical flatmount image of a mouse eye within one minute, a three order magnitude time saving improvement from the current manual procedure. This method is already contributing to the image analysis of RPE flatmounts.

INDEX WORDS: RPE, AMD, Cutbox, K-Means Clustering, Variability
A NOVEL METHOD FOR AUTOMATED CELL IMAGE SELECTION

by

SHUMAN GUO

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of
Master of Science
in the College of Arts and Sciences
Georgia State University
2012
A NOVEL METHOD FOR AUTOMATED CELL IMAGE SELECTION

by

SHUMAN GUO

Committee Chair: Yuanhui Xiao

Committee: Yi Jiang
Yichuan Zhao

Electronic Version Approved:

Office of Graduate Studies
College of Arts and Sciences
Georgia State University
December 2012
DEDICATION

First of all, this thesis paper is lovingly dedicated to my advisor Yuanhui Xiao and Co-advisor Yi Jiang. Without their support and love this project would not have been made possible.

Also, I would like to thank my parents, who support me throughout my studies.
# TABLE OF CONTENTS

LIST OF TABLES .................................................................................................................. vii  
LIST OF FIGURES .................................................................................................................. viii  
1 INTRODUCTION..................................................................................................................... 1  
\hspace{0.5em} 1.1 The Goal of the Project ............................................................................................. 3  
2 METHOD ................................................................................................................................ 4  
\hspace{0.5em} 2.1 Data Preparation ...................................................................................................... 4  
\hspace{0.5em} \hspace{0.5em} 2.1.1 Experimental Images (Flatmount RPE Image) .................................................. 4  
\hspace{0.5em} \hspace{0.5em} 2.1.2 Basics of Image Processing and Analysis for Data Preparation ......................... 4  
\hspace{0.5em} 2.2 Algorithm .................................................................................................................. 11  
3 RESULTS ............................................................................................................................... 17  
4 CONCLUSIONS .................................................................................................................... 22  
REFERENCES .......................................................................................................................... 24  
APPENDIX ............................................................................................................................... 25
LIST OF TABLES

Table 1 Data after K-means clustering .................................................................................................. 18
LIST OF FIGURES

Figure 1(a). Schematic illustration of the eye anatomy ......................................................... 1
Figure 1(b). A schematic cross-sectional illustration of the macular region .............................. 2
Figure 2. An example of mouse flatmount RPE Image using a confocal microscope .................. 5
Figure 3. Typical image Cutboxes .......................................................................................... 7
Figure 4. Cutbox and CellProfiler analysis ........................................................................... 7
Figure 5. Extract the RGB vectors from the image .................................................................. 8
Figure 6. Screen shot of a CellProfiler Pipeline ...................................................................... 9
Figure 7. Grayscale image represented by a matrix .................................................................. 10
Figure 8. Black-and-white image represents by a matrix ....................................................... 11
Figure 9. RPE floatmount images that we tested our method on ............................................. 17
Figure 10. Output examples .................................................................................................... 20
Figure 11. Manual selection of the cutboxes .......................................................................... 21
Figure 12. Our clustering based, automated selection yields within one minute 130 good cutboxes from the same RPE flatmount image (Figure 2) ................................................................. 21
Figure 13. CellProfiler snapshot ............................................................................................ 23
1 INTRODUCTION

Age-related macular degeneration (AMD) is a neurodegenerative disease of the eye, considered a looming epidemic for the aging population [1, 2, 3]. AMD represents the late phase of age-related maculopathy resulting in a loss of vision in the center of the visual field (geographic atrophy) or blindness (choroidal neovascularization) [4]. Geographic atrophy is characterized by severe atrophy of the retinal pigment epithelial (RPE) and loss of overlying photoreceptors (see schematic illustration, Fig. 1(a)). Choroidal neovascularization grows through the break in Bruch's membrane from the underlying choriocapillaris and invades the sub-RPE, the sub-retinal space, or both (Fig. 1(b)). These abnormal vessels leak serum and blood that can induce fibrotic reaction known as a disciform scar. Only very recently new anti-angiogenics, e.g. anti-vascular endothelial growth factor, drugs have begun to show promises in treating the choroidal neovascularization [5, 6]. However, the long-term prognosis of this form of AMD is poor in many cases, especially once retina is committed. At present, there is no way to distinguish normal aging of the eye from AMD, and even less can we distinguish an eye that will progresses to new blood vessel growth from another that does not progress, before retina is impaired.

Figure 1(a). Schematic illustration of the eye anatomy. Responsible of the eye.
A key site of AMD pathology is the RPE, which plays a critical role in the maintenance of the outer retina. RPE is composed of a single layer of cells directly behind the photoreceptors and firmly attached to the underlying choroid (Fig. 1(b)). This layer has several functions including participation in the regeneration of retina in visual cycle and the phagocytosis of shed photoreceptor outer segments. Healthy RPE cells are critical for maintaining the structure of the retina, preserving normal photoreceptor function. Indeed, abnormal RPE cells contribute to disease mechanism and progression in numerous retinal diseases besides AMD, including Stargardt's dystrophy, Best's disease, and others [7].

We hypothesize that the morphology of RPE, a key site of AMD pathology, correlates with the age and disease status of the eye and can be quantified and modeled to predict the disease progression.
To test this hypothesis, first we need to establish the correlation between RPE morphology and age and disease status of the eye. Given a mouse RPE flatmount image (Figure 2), the present practice is to: 1) divide the flatmount image into zones of concentric rings of increasing radial distance to the center, where the optic nerves have been cut out; 2) manually make cutboxes (a cutbox is a rectangular region of an image) that contain about 100-150 cells; 3) differentiate ‘good’ cutboxes from ‘bad’ ones by a student eye-balling the images; 4) Finally, use the CellProfiler [8] to extract the numeric information from the selected ‘good’ cutboxes for further analysis. This image analysis process has been very time consuming, taking a student about one whole day to analyze one RPE flatmount image. The bottleneck lies on the manual selection of cutboxes.

1.1 The Goal of the Project

This project is aim to automate the cutboxes selection by using a modern clustering method. Our approach can be summarized as follows. Given a RPE flatmount image, we : 1) convert it to a binary image; 2) divide the converted image into size of 200 pixels by 200 pixels sub-images containing approximately 100-150 cells; 3) define variability ($IQR^-$, $IQR$, $IQR^+$, median) as the features of the cutboxes; and 4) use a K-means clustering method to automatically differentiate ‘good’ cutboxes set from ‘bad’ ones. The output cutboxes will be used for further analysis using CellProfiler. This procedure usually takes less than one minute for one mouse eye flatmount. The processing time mainly depends on the sizes of the images, and how many output images we need to save on the disk.
2 METHOD

2.1 Data Preparation

2.1.1 Experimental Images (Flatmount RPE image)

The flatmount RPE images were obtained at John Nickerson’s Lab at the Emory Eye Center. The protocols for obtaining flatmount RPE images are briefly as follows.

The mouse eye was fixed with formalin for 10 minutes. Then on a microscope slide, any extra scleral tissue from eye including optic nerve is cut away. Four cuts were extended from puncture using 3 mm scissors from cornea back towards optic nerve; each section was unfurled to reveal and remove the lens. Then 4.5 ul of Zymed rabbit anti-ZO-1 antibody were added to 450 ul of antibody buffer, and the secondary was preblocked by adding 0.45 ul of oregon green conjugated anti-rabbit IgG (Invitrogen) to 450 ul of antibody buffer. The images were taken using confocal microscopy with 3 optical sections 5 um apart as the Z-stacks; each image is 1024x1024 pixels in size. These images are stitched together into a full flatmount (Figure 2) using Photoshop.

For human eyes, RPE flatmounts were stained with AF635-phalloidin to visualize RPE cell borders. In the confocal images: green florescence corresponded to phalloidin staining of actin cytoskeleton; red florescence corresponded to propidium iodide staining of nuclei; blue was autoiuorescence. The green channel was used for our analysis.

2.1.2 Basics of Image Processing and Analysis for Data Preparation
A typical example of the mouse eye RPE flatmount image is shown in Figure 2, which is a normal RGB image in jpeg format.

Figure 2. An example of mouse flatmount RPE image using a confocal microscope
For the ease of CellProfiler processing, the large flatmount image were cut into smaller boxes containing approximately 100 to 150 cells. These were called cutboxes. We define our cutbox size to be a square of size 200 by 200 pixels. From a typical mouse RPE flatmount image, over one thousand non-overlapping cutboxes can be produced. To automatically make the ‘good’ cutbox selection, the very first step is to find a method to distinguish the ‘good’ and ‘bad’ images. Since our goal is to get ‘good’ cutboxes for CellProfiler analysis, the ‘good’ ones should be like the top-left one in Figure 3. Necessary features for the ‘good’ cutboxes are 1) the numbers of cells in the ‘good’ cutboxes are almost the same, 2) no blurry or overlapped regions, and 3) no damaged region or with minimal damage. We use these three criteria to recognize ‘good’ images.

Automatic cutting of the flatmount image yields eight typical types of cutboxes, as listed in Figure 3. Clearly, not every cutbox can be used by CellProfiler to extract biological data. For example, the Cutbox-h in Figure 3, the net-like cell information part will be correctly recognized in CellProfiler. However, the blurry pattern in the upper part of the image will be recognized as some fuzzy information in CellProfiler. Cutbox-A of Figure 3 is the only cutbox format can be recognized correctly by CellProfiler. An example of CellProfiler analysis of a good image is shown in Figure 4. About one hundred good cutboxes can be obtained from an experimental image. Obviously, it is tedious to select them from nearly one thousand cutboxes manually, calling for automatic procedures for selecting useful cutboxes.

The key observation that inspired us or develops our algorithm for automatic selection of informative cutboxes is the distribution of green pixels in cutboxes of type A (Figure 3), which is dramatically different from that of the cutboxes of other types. Hence, this feature alone will allow us to differentiate useful cutboxes from all the other cutboxes.
Figure 3. Typical image Cutboxes. Cutbox-A contains a network structure of cell boundaries, which is the only ‘good’ cutbox pattern we want to choose. Cutbox-B, C and D contain no cell information and can be easily discarded using frequency calculation. Cutbox-E includes all blurry patterns. Cutbox-F contains three parts: no information, blurry region, and good region. Cutbox-G & H contain either two of the three: no information, blurry region or good region.

Figure 4. Cutbox and CellProfiler Analysis. The left graph is the original cutbox format fed into CellProfiler. The right one is the one analyzed by some CellProfiler pipeline. Each colored polygon is a cell identified by the software. For a ‘good’ cutbox, both of the images should contain the similar structures.
To prepare the data for further analysis, the RGB microscopy images have to be transformed. We take two steps for image transformation: RGB image to grayscale, and grayscale to black-and-white. There are several reasons that we only keep the green channel. Figure 5 shows the red, green and blue channel separation from a cutbox image. We can easily decide that only green channel provides the network pattern of cell boundaries. The red channel contains the cell nuclei, which cannot be easily used to identify cells, because a large portion of the cells have multiple nuclei. The blue channel is the background auto-flourescence, which does not contain any useful cell structural information. This is the first reason we select the only green channel information and transfer it into grayscale images.

Figure 5. Extract the RGB vectors from the image. The very left graph is the original cutbox. We separate the Red, Green, and Blue channel information from it and get the graphs in the middle. Only the green channel can be translated into a network pattern of cell boundaries in grayscale.
Another important reason to choose the green channel is that CellProfiler analysis of cell patterns only relies on the green channel. Figure 6 shows a CellProfiler pipeline (the sequence of processing commands), where only the green channel information was used for cell identification. Therefore, we choose the green channel.

Figure 6. Screen shot of a CellProfiler Pipeline. Each task or module of the pipeline for our project is shown on the left part of the software. When open the third module- ImageMath, we see that only the original green channel information is kept and used in the following analysis.

To further simplify the analysis, we made another transformation from grayscale image to black-and-white image. For a specific part of the grayscale image shown in Figure 7, the matrix below represents the grayscale image; each element of the matrix contains an integer value in the range from 0 to 255. If we highlighted all pixels’ value greater than 100, the shape of the cell edge can be seen in the matrix. To identify the network pattern, we only need to decide whether the information represents an
edge or not. Therefore, binary information is good enough. Hence, we simplify the matrix representation of a grayscale image by replacing a pixel value greater than 100 by 1 and that lower than 100 by 0. As a result, our grayscale image is changed into black-and-white image which only includes 0’s (black) and 1’s (white) information. Figure 8 illustrates that the black-and-white image transformed from grayscale image in Figure 7. The threshold for this conversion was the default value of 100 in Matlab [9] image processing toolbox. The binary data structure simplifies our frequency and distribution calculation.

![Grayscale image represented by a matrix. When highlighting all pixel value larger than 100, we could get the cytoskeleton structure from the matrix itself.](image-url)
Figure 8. Black-and-white image represents by a matrix. This black-and-white image is transferred from the grayscale image in Figure 7. All 1’s information can describe the cell edge structure.

2.2 Algorithm

To find out how the white pixels are distributed in a cutbox, we divide a cutbox into nBlocks blocks of size nBlockWidth by nBlockHeight (we use 200*200 in this project). (Where nBlocks, nBlockWidth and nBlockHeight are integers) Each block is a small image with nBlockWidth by nBlockHeight (=nPixelsPerBlock) pixels.
For block $i$ ($1 \leq i \leq n$Blocks), let $X_i$ be the number of white pixels. The values of $\{X_i\}_{i=1}^{nBlocks}$ roughly reflect the distribution of white pixels in a cutbox. In statistics, there are two widely used measures for variability of data set. The first one is the sample variance, which is defined as

$$S^2 = \frac{\sum_{i=1}^{n} X_i^2 - (\sum_{i=1}^{n} X_i)^2/n}{n-1}, \tag{1}$$

where $n$ is the sample size and it is equal to nBlocks in our case. The sample variance is very sensitive to outliers. To eliminate or reduce the influence of outliers, we decide to use the other measure, which is the inter-quartile range (IQR). Let $Q_1$ and $Q_3$ be the first and third quartiles of the values of set $\{X_i\}_{i=1}^{n}$, respectively. The interquartile range of $\{X_i\}_{i=1}^{nBlocks}$ is defined as

$$IQR = Q_3 - Q_1. \tag{2}$$

While IQR is very resistant to outliers, it only uses approximate half of available data values, and the values that are higher than $Q_1$ and $Q_3$ are simply thrown away. To recover some more information, we use two additional values, $Q_1 - P_{05}$ and $P_{95} - Q_3$, to describe the variability, where $P_{05}$ is the 5th percentile and $P_{95}$ is the 95th percentile. Note that values higher than $P_{95}$ or lower than $P_{05}$ are discarded, since they are likely to be outliers. Therefore the variability of the values of the set $\{X_i\}_{i=1}^{n}$ is represented by a triplet $(IQR^-, IQR, IQR^+)$, where

$$IQR^- = Q_1 - P_{05} \text{ and } IQR^+ = P_{95} - Q_3. \tag{3}$$

Hereafter, we shall call the variability of the set $\{X_i\}_{i=1}^{nBlocks}$ of a cutbox reflected by the triplet $(IQR^-, IQR, IQR^+)$ the **variability of the cutbox.** We expect that values of the triplet $(IQR^-, IQR, IQR^+)$
are significantly smaller for good cutboxes than those for bad cutboxes. (The pure-white and pure-black cutboxes of course have smaller variability. However, it is trivial to distinguish them and they will be discarded before we apply a K-mean clustering method to select the ‘good’ cutboxes.)

It is tempting to set a cutoff value for each component of the triplet \((IQR^-, IQR, IQR^+)\), and declare a cutbox as good if the value of each component of the triplet \((IQR^-, IQR, IQR^+)\) less than the corresponding cutoff value for the cutbox. However, this threshold is practically impossible to be set since the cutoff values vary from image to image. Hence we resort to the clustering methods, which are widely used to group objects according to the features of the objects under study. For example, in genetics, clustering methods often contribute to identify genes with similar biological function(s). With correctly selected clustering methods, objects with similar features tend to be assigned to the same group or cluster.

To use the triplets \((IQR^-, IQR, IQR^+)\) as features to identify good cutboxes, we face a problem: the triplet \((IQR^-, IQR, IQR^+)\) is location-invariant. That is, adding any constant to each \(X_i\) \((1 \leq i \leq n\text{Blocks})\) does not change the values of the components of the triplet \((IQR^-, IQR, IQR^+)\). Thus it is possible that two cutboxes with totally different shapes have similar variabilities. To guard against this case, it is necessary to add one more feature. Here is our observation. If two cutboxes have similar variabilities and their centers are close to each other, then it is unlikely that they belong to different cutbox categories. In statistics, both the mean and the median can be used to describe the center or the location of a data set. The former is sensitive to outliers while the latter is almost immune from the effect of outliers. Thus, we use the median as the fourth feature in the process for selecting ‘good’ cutboxes.

Out of the many possible clustering algorithms and software available, we choose the popular K-means clustering method [10] for its simplicity. The basic steps of K-means with the number K of clusters is given in advance, are
1. Select a set of \( K \) points as cluster seeds. This is often done by randomly selecting \( K \) objects.
2. Assign each individual object to the cluster of whose centroid is nearest.
3. Repeat step 2 until no further changes occur in the cluster composition.

Like many other clustering methods, the K-means method requires a distance measure to describe the dissimilarity of two different objects. The Euclidean distance is usually used. However, we prefer the more outlier-resistant city-block distance, which is defined as follows. Let \( f^i = (f^i_1, f^i_2, ..., f^i_d) \) \((i = 1, 2, \ldots, d)\) be the feature vectors for objects 1 and 2, respectively, the city block distance is given by

\[
d(f^1, f^2) = \sum_{j=1}^{d} |f^1_j - f^2_j|. \tag{4}
\]

Using this distance, the centroid of a cluster is defined as the point in the feature space whose components are the medians of the corresponding components of the feature vectors of the objects in the cluster.

Let \( \{1, 2, \ldots, m\} \) be the labels of \( m \) objects under study. A set of clusters \( \{H_j\}_{j=1}^{K} \) are said to constitute a partition of the objects \( \{1, 2, \ldots, m\} \) if they are mutually exclusive and any object belongs to one of them. Let \( f^i \) be the feature vector of object I and \( f^c_j \) the centroid of the cluster \( H_j \). From the point view of mathematical optimization, the K-means method tries to find a partition \( \{H_j\}_{j=1}^{K} \) to minimize the following quantity

\[
Q(H_1, H_2, ..., H_K) = \sum_{j=1}^{K} \sum_{i \in H_j} d(f^i, f^c_j). \tag{5}
\]
The clusters that are finally selected depend on the choice of the seed. If the seed is illchosen, then $Q(H_1, H_2, \ldots, H_k)$ will not achieve its global minimum. In other words, the K-means method does not give the correct solution. For our image study, the solution is to apply the K-means method with 40 to 200 different seeds and take the partition of clusters with the smallest value of $Q(H_1, H_2, \ldots, H_k)$. This treatment in theory does not guarantee the global minimum of $Q(H_1, H_2, \ldots, H_k)$. However, in practice it often gives satisfactory results.

Another important issue for K-means method is the selection of number K of clusters. However, this issue does not concern us because our aim is to identify the good cutboxes set by using the features $(IQR^-, IQR, IQR^+, \text{median})$. From our limited experience, after excluding the all-white, the all-black and the half-white and half-black cutboxes, by letting $K=5$, the K-means method will assign almost all the good cutboxes to a cluster if the quality of the image under investigation is reasonable high. It is unnecessary to exclude the all-white/black cutboxes before applying the K-means method. If these cutboxes are also included in the clustering process, then a good choice for $K$ is 7 or 8.

After the K-means method assigns all the cutboxes to different clusters, our next step is to find out which cluster has the desired cutboxes. Thus, for each cluster, we compute the median and $Q_3$ for each component of the triplet $(IQR^-, IQR, IQR^+)$, which will be termed cluster median and cluster $Q_3$ for the component, respectively. And we choose the cluster with the smallest median for each component of the triplet $(IQR^-, IQR, IQR^+)$. If the quality of an image is not too low, we shall find that there is at least one cluster whose cluster medians for the components of the triplet $(IQR^-, IQR, IQR^+)$ are significantly low. For any un-wanted cluster, the cluster median for at least one component of the triplet $(IQR^-, IQR, IQR^+)$ is significant high.
The cluster with the smallest cluster medians contains almost all the good cutboxes. However, it also has many undesired cutboxes. Those cutboxes often have relatively high values for at least one component of the triplet $(IQR^-, IQR, IQR^+)$. Thus, we eliminate a cutbox from our list if at least one of its components is higher than the cluster $Q_3$. The cutoff value is chosen purely out of our experience. Different cutoff points may be used. Based on our numerical experiments, we find that the use of $Q_3$ as the cutoff point excludes almost all the unwanted cutboxes while keeps most useful ones.

Occasionally, there may be two clusters, both of which have plenty of good cutboxes. For this situation, we should consider combining the two clusters and compute the $Q_3$ for the combined cluster, and apply the $Q_3$ rule to eliminate the un-wanted cutboxes. Another choice is to use a different value for $K$ and run the K-mean procedure again.
3 RESULTS

We implemented both image processing and image selection in Matlab [9]. We processed and analyzed five sample images (show in figure 9). The first one MAC-8145-rd10-100.jpg is accompanied by the manual selection results. Our program automatically searches for all jpeg format images in the folder where the program located.

Figure 9. RPE floatmount images that we tested our method on. The first four images are whole mouse RPE flatmount image. The last one is a portion of a human RPE near macula. Our algorithm and application work fine on all of them.
Table 1 Data after K-means Clustering

<table>
<thead>
<tr>
<th>Feature</th>
<th>Centroid</th>
<th>P05</th>
<th>P25</th>
<th>Median</th>
<th>P75</th>
<th>P95</th>
</tr>
</thead>
<tbody>
<tr>
<td>L2 - Cluster # 1, size = 27</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P25-P05</td>
<td>121.8</td>
<td>74.0</td>
<td>92.0</td>
<td>122.5</td>
<td>148.5</td>
<td>182.0</td>
</tr>
<tr>
<td>IQR</td>
<td>290.3</td>
<td>217.5</td>
<td>258.0</td>
<td>283.0</td>
<td>311.5</td>
<td>383.0</td>
</tr>
<tr>
<td>P95-P75</td>
<td>32.5</td>
<td>0.0</td>
<td>12.0</td>
<td>30.0</td>
<td>41.5</td>
<td>89.5</td>
</tr>
<tr>
<td>Median</td>
<td>3.1</td>
<td>1.0</td>
<td>1.0</td>
<td>2.0</td>
<td>4.0</td>
<td>14.0</td>
</tr>
<tr>
<td>L2 - Cluster # 2, size = 183</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P25-P05</td>
<td>27.8</td>
<td>13.5</td>
<td>17.0</td>
<td>22.5</td>
<td>30.0</td>
<td>64.0</td>
</tr>
<tr>
<td>IQR</td>
<td>43.4</td>
<td>22.0</td>
<td>27.0</td>
<td>31.5</td>
<td>44.5</td>
<td>118.5</td>
</tr>
<tr>
<td>P95-P75</td>
<td>29.3</td>
<td>2.0</td>
<td>17.5</td>
<td>23.0</td>
<td>33.0</td>
<td>77.5</td>
</tr>
<tr>
<td>Median</td>
<td>266.3</td>
<td>70.0</td>
<td>261.0</td>
<td>289.0</td>
<td>312.0</td>
<td>342.0</td>
</tr>
<tr>
<td>L2 - Cluster # 3, size = 16</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P25-P05</td>
<td>30.2</td>
<td>8.0</td>
<td>13.0</td>
<td>20.8</td>
<td>35.0</td>
<td>90.5</td>
</tr>
<tr>
<td>IQR</td>
<td>145.3</td>
<td>56.5</td>
<td>90.0</td>
<td>124.8</td>
<td>220.0</td>
<td>250.5</td>
</tr>
<tr>
<td>P95-P75</td>
<td>197.9</td>
<td>55.5</td>
<td>93.5</td>
<td>217.5</td>
<td>284.0</td>
<td>342.0</td>
</tr>
<tr>
<td>Median</td>
<td>2.1</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>9.0</td>
</tr>
<tr>
<td>L2 - Cluster # 4, size = 23</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P25-P05</td>
<td>29.5</td>
<td>4.5</td>
<td>14.0</td>
<td>25.5</td>
<td>44.5</td>
<td>65.5</td>
</tr>
<tr>
<td>IQR</td>
<td>404.0</td>
<td>326.5</td>
<td>349.5</td>
<td>422.5</td>
<td>451.5</td>
<td>462.5</td>
</tr>
<tr>
<td>P95-P75</td>
<td>29.1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>62.0</td>
<td>114.0</td>
</tr>
<tr>
<td>Median</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>L2 - Cluster # 5, size = 21</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P25-P05</td>
<td>218.0</td>
<td>112.5</td>
<td>178.5</td>
<td>208.5</td>
<td>277.0</td>
<td>299.0</td>
</tr>
<tr>
<td>IQR</td>
<td>125.5</td>
<td>45.0</td>
<td>94.5</td>
<td>135.5</td>
<td>152.5</td>
<td>198.0</td>
</tr>
<tr>
<td>P95-P75</td>
<td>28.5</td>
<td>0.5</td>
<td>8.0</td>
<td>18.0</td>
<td>43.0</td>
<td>77.0</td>
</tr>
<tr>
<td>Median</td>
<td>25.0</td>
<td>1.0</td>
<td>1.0</td>
<td>4.0</td>
<td>25.0</td>
<td>89.0</td>
</tr>
</tbody>
</table>
The first step of our project is to preprocess the flatmount image. The program will divide the image into cutboxes by size of 200*200 pixels. And then transfer the green channel information from the original RPE flatmount image. The black-and-white image will be generated by the grayscale image from the green channel data. After this, we need to calculate the varibilities of the cutboxes. And then we use k-means clustering to group the data according to the features we defined in our algorithm. Table 1 shows the data we generate for image MAC-8145-rd10-100.jpg. Listed in Table 1, the features ($IQR^-, IQR, IQR^+, median$) vary significantly between different clusters, and can easily distinguish the good ones out of the rest. Based on our assumption, cluster 2 is selected as the ‘good’ cutboxes in this particular example.

After the process, the ‘good’ cutboxes are automatically stored into a folder with the original image name. And figure 11 shows the examples of the ‘good’ cutboxes selected by our program. The cutbox location is recorded in the name of the image description. Therefore, the analyst who uses CellProfiler could know the location of the cutbox.
Figure 10. Output examples. The output cutboxes set will have the same prefix ‘cutbox_’. And the cutbox name also contains the position information of the top-left pixel in the original RPE flatmount image.

The old manual selection of cutboxes would take a student almost a whole day (10 hours) for one flatmount image. And the number of selected cutboxes is 51 for the Figure 11. To process the same image, our method only takes about one minute, and the yield is more than 2 fold larger: 130 cutboxes (Figure 12).
Figure 11. Manual selection of the cutboxes generated 51 cutboxes from the flatmount image of Figure 2. The process time is in the order of 10 hours.

Figure 12. Our clustering based, automated selection yields within one minute 130 good cutboxes from the same RPE flatmount image (Figure 2).
To facilitate image analysis of RPE morphology study, we have developed a combined image processing and K-mean clustering tool to automatically select for good image cutboxes. Our program can 1) automatically detect the .jpg format images located within the same folder of the application; 2) divide the images into size of 200*200 cutboxes; 3) select the ‘good’ cutboxes by using K-means clustering method; 4) save the ‘good’ cutboxes into a folder with the original image name; and 5) the output images with the original image location of the top left pixel. Our program improves the current manual method in both time and number of selections. The average processing time using our method is less than one minute, a significant improvement to many hours for manual approaches. In addition, our method results in several folds more good cutboxes than manual selection.

There are several directions we can take to further improve the tool. Right now, our program only can automatically detect the jpeg format images in the folder. And then makes the ‘good’ cutboxes selected and stored in jpeg format. However, the flatmount RPE images obtained at John Nickerson’s Lab at the Emory Eye Center are usually stored in .tif format. Based on their convenience, it is better to change the default format to .tif file later. Our current method to make the image division is from the very top-left pixel of the image. And cut the image into non-overlapping cutboxes with the size of 200 by 200 pixels. Though we can get enough ‘good’ cutboxes set, those cutboxes are not the best ones. That is, some good features of the cell structure may be lost in our way of cutting the image. We may develop a better way to divide the image in the future. Some good suggestion is to detect the center of the image first (this center may not be the center of the whole image, but the center of the whole eye), and then divide the image from the center point.

A really feasible and possible improvement for this project is to store the black-and-white cutboxes we make as the targets instead of the original RGB ones. Then make them processed in CellProfiler
directly. By observing the pipeline processes they use in CellProfiler (Figure 13), the steps are really similar as we analysis the image. Our image pre-processing step will transfer the RGB flatmount image into grayscale only keep the green channel information. The second and third steps of the pipeline in CellProfiler did the same thing. Thus, we can save two processes in CellProfiler: ColorToGray, and ImageMath for every cutbox analysis. We expect this method to be an indispensible addition to the RPE analysis.

Figure 13. CellProfiler snapshot. The second and third processes of the CellProfiler pipeline are almost same as what we make the image transformation. Both of them are aim to get the grayscale image and only keep the green channel information.
REFERENCES


APPENDIX

Matlab code:

close all;
clear all;clc

img_filename=dir('*.jpg');
for img_num=1: length(img_filename)
    A=imread(img_filename(img_num).name);
    num=10;
    num1=10;
    a=200;
    b=200;
    [x y z]=size(A);
    row=ceil(x/a);
    col=ceil(y/b);
    aa=a/num;
    bb=b/num1;
    x_coordinate=0;
    y_coordinate=0;
    block=zeros(row,col);
    bnum=1;
    G=A(:,:,2);
    total=0;
y=[];
X=[];
csv_x=[];

for i=1:row-1
    for j=1:col-1
        m=a*(i-1)+1;
        n=a*i;
        p=b*(j-1)+1;
        q=b*j;
        GG=G(m:n,p:q);
        level=graythresh(GG);
        GGG=im2bw(GG,level);
        AG=reshape(GGG,1,a*b);
        AG=double(AG);
        x=tabulate(AG);
        F=[];
        E=[];
        bbnum=1;
        count=0;
        if x(1,3)<98  %clear the all whites or all blacks areas
            for iii=1:num
                for jji=1:num1
                    mm=aa*(iii-1)+1;
                    nn=aa*iii;
                    pp=bb*(jjj-1)+1;
                    ...
qq=bb*jjj;

G4=GGG(mm:nn,pp:qq);

AGG=reshape(G4,1,aa*bb);

AGG=double(AGG);

xx=tabulate(AGG);

F=[F xx(1,2)];

bbnum=bbnum+1;

end

end

end

R=iqr(F); % R=Q3-Q1

y=quantile(F,[.05 .25 .50 .75 .95]);

X=[bnum y(2) y(3) y(2)-y(1) R y(5)-y(4)];

if isnan(R)==0 && R~=0 && (y(5)-y(4))~=0 % R is not NaN and not 0
    csv_x=[csv_x;X];
end

bnum=bnum+1;

end

end

[r c]=size(csv_x);

new=[];

for i=1:r
if csv_x(i,2)==400 && csv_x(i,3)==400
    new=[new; csv_x(i,:)];
end
end

new_x=new(:,[2 3 5]);

kmeans_x=kmeans(new_x,5,'distance','cityblock','emptyaction','drop','replicates',40);%city block

new=[new kmeans_x];

s1=[];

s2=[];

s3=[];

s4=[];

s5=[];

[row col]=size(new);

for i=1:row
    if new(i,7)==1
        s1=[s1; new(i,4:6)];
    elseif new(i,7)==2
        s2=[s2; new(i,4:6)];
    elseif new(i,7)==3
        s3=[s3; new(i,4:6)];
    elseif new(i,7)==4
        s4=[s4; new(i,4:6)];
    elseif new(i,7)==5
        s5=[s5; new(i,4:6)];
    end
end
s5=[s5; new(i,4:6)];
end

diff

mini_array=[mean(s1(:,2)) mean(s2(:,2)) mean(s3(:,2)) mean(s4(:,2)) mean(s5(:,2))];
mini_x=min(mini_array);
bound=quantile(s1,[.50 .60 .70 .75 .80]);% upper confidence bound 50% 60% 70% 75% & 80%
cluster_num=1; % decide which cluster it is
if mini_x==mean(s2(:,2))
    cluster_num=2;bound=quantile(s2,[.50 .60 .70 .75 .80]);
elseif mini_x==mean(s3(:,2))
    cluster_num=3;bound=quantile(s3,[.50 .60 .70 .75 .80]);
elseif mini_x==mean(s4(:,2))
    cluster_num=4;bound=quantile(s4,[.50 .60 .70 .75 .80]);
elseif mini_x==mean(s5(:,2))
    cluster_num=5;bound=quantile(s5,[.50 .60 .70 .75 .80]);
end

[row col]=size(new);
final=[];

img_foldername=strtok(img_filename(img_num).name,'.');
folder_name=[img_foldername,'_cutboxsets'];
mkdir(folder_name);
pt=3;% use 70% as default
for i=1:row
if new(i,7)==cluster_num && new(i,4)<=bound(pt,1) && new(i,5)<=bound(pt,2) && new(i,6)<=bound(pt,3) % for the img which is blur
    final=[final; new(i,:)];
    end
end

cutbox_num=final(:,1);
[c_num cc_num]=size(cutbox_num);
[x y z]=size(A);
row=ceil(x/a);
col=ceil(y/b);
final_count=0;

for c_i=1:c_num
    bnum=1;
    for i=1:row-1
        for j=1:col-1
            m=a*(i-1)+1;
            n=a*i;
            p=b*(j-1)+1;
            q=b*j;
            B=A(m:n,p:q,:);
            if bnum==cutbox_num(c_i)
                imwrite(B,[folder_name, 'cutbox_x',num2str(p),'_y',num2str(m),'.jpg'],jpg');
                final_count=final_count+1;
            end
        end
    end
end
bnum=1;

end

bnum=bnum+1;

end

end

total_cuts=bnum-1;

final_count;

figure;

imshow(A);

title('original image');

end