A FUZZY APPROACH FOR CELL COUNTING IN POORLY-ILLUMINATED IMAGES APPLIED TO A CELL-PHONE MICROSCOPE

A Thesis

Presented to the faculty of the Department of Electrical and Electronic Engineering

California State University, Sacramento

Submitted in partial satisfaction of the requirements for the degree of

MASTER OF SCIENCE

in

Electrical and Electronic Engineering

by

Mehdi Rahimzadeh Soumesaraei

SUMMER
2012
A FUZZY APPROACH FOR CELL COUNTING IN POORLY-ILLUMINATED IMAGES APPLIED TO A CELL-PHONE MICROSCOPE

A Thesis

by

Mehdi Rahimzadeh Soumesaraei

Approved by:

______________________________, Committee Chair
Warren D. Smith

______________________________, Second Reader
Stephen M. Lane

______________________________, Third Reader
V. Scott Gordon

______________________________
Date
Student: Mehdi Rahimzadeh Soumesaraei

I certify that this student has met the requirements for format contained in the University format manual, and that this thesis is suitable for shelving in the Library and credit is to be awarded for the thesis.

_____________________, Graduate Coordinator
Preetham B. Kumar

_____________________
Date

Department of Electrical and Electronic Engineering
Abstract

of

A FUZZY APPROACH FOR CELL COUNTING IN POORLY-ILLUMINATED IMAGES APPLIED TO A CELL-PHONE MICROSCOPE

by

Mehdi Rahimzadeh Soumesaraei

A blood cell count is a common diagnostic tool in medicine, and one way to obtain such a count is from an image of a blood smear. Researchers at the Center for Biophotonics Science and Technology (CBST) at the University of California, Davis have developed an attachment to convert a cell phone to a microscope. The images provided by this cell-phone microscope suffer from several artifacts, such as radial distortion and non-uniform illumination. It is desired to develop a software application for a smart phone to perform image processing and pattern recognition that can return an approximate blood count.

In this work, prototype software has been developed on a personal computer (PC) that performs the whole procedure of image processing and pattern recognition to provide an approximate red blood cell count. To do the red blood cell count, images that are taken of a blood sample by a smart phone are transferred to a PC for processing. Radial distortion correction and cropping the defocused area of the image are done as pre-processing steps in preparation for robust cell recognition. Adaptive multi-level segmentation is performed as the second step to transform the image to a fuzzy scene, followed by the red cell recognition step.

A fuzzy approach is taken for red cell recognition. The fuzzy approach presented in this work utilized fuzzy sets and not fuzzy logic. Adaptive image fuzzification and
fuzzy criterion functions proposed in this thesis have higher performance than conventional counting methods. The proposed approach is robust against fuzziness of the image due to the poor quality of a cell phone image, taken under non-laboratory conditions. The recognition process in this application is a blind search method that is independent of manual calibration and learning.

Most of this work has been dedicated to enhancing the algorithm of cell recognition even in poorly-illuminated images. This work focuses on red blood cell counting. However, the concept can be extended to other blood smear counting, such as white blood cells and platelets. This algorithm is tested on seven blood smear images, and the average values for precision and recall are 95.6 percent and 95.4 percent, respectively.

__________________________________, Committee Chair
Warren D. Smith

______________________________
Date
ACKNOWLEDGEMENTS

I wish to express my gratitude to my graduate advisor, Prof. Dr. Warren D. Smith who offered invaluable assistance, support, and guidance. This research would not have been possible without the guidance and the help of Prof. Dr. Stephen Lane and Assoc. Prof. Dr. Sebastian Wachsmann-Hogiu. Deepest gratitude is also due to the member of the supervisory committee, Prof. Dr. Scott Gordon. I also wish to express my gratitude to the graduate coordinator, Prof. Dr. Preetham Kumar, for his invaluable help and guidance during the years of my master’s degree.

Special thanks to all my dear friends who have always been helpful to me. Finally, I would like thank my family for their support and encouragement throughout my life.
TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Acknowledgments</th>
<th>vi</th>
</tr>
</thead>
<tbody>
<tr>
<td>List of Tables</td>
<td>ix</td>
</tr>
<tr>
<td>List of Figures</td>
<td>x</td>
</tr>
</tbody>
</table>

Chapter

1. INTRODUCTION ................................................................. 1
   1.1. Overview .............................................................. 1
   1.2. Blood Cell Count.................................................... 2
   1.3. Organization of the Remaining Chapters ......................... 3

2. BACKGROUND ........................................................................... 5
   2.1. Previous Work ............................................................ 5
   2.2. This Work ................................................................. 7

3. REVIEW OF THE LITERATURE.................................................. 9
   3.1. Radial Distortion Correction ....................................... 9
   3.2. Focus Criterion Functions .......................................... 11
      3.2.1. Gray Variance ..................................................... 13
      3.2.2. Tenengrad .......................................................... 14
      3.2.3. Sum Modified Laplacian ....................................... 15
      3.2.4. Sum Modulus Difference ...................................... 16
   3.3. Image Segmentation ................................................... 17
      3.3.1. Edge-Based Methods ............................................ 19
      3.3.2. Region-Based Methods ......................................... 20
         3.3.2.1. Otsu Method .................................................. 24
         3.3.2.2. Minimum Error Method ..................................... 28
         3.3.2.3. Fuzzy C-Mean Method ..................................... 32
   3.4. Image Fuzzification .................................................. 33
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.5. Object Recognition and Counting</td>
<td>42</td>
</tr>
<tr>
<td>3.5.1. Pixel-Counting Method</td>
<td>44</td>
</tr>
<tr>
<td>3.5.2. Masking Method</td>
<td>46</td>
</tr>
<tr>
<td>4. ALGORITHM DEVELOPMENT</td>
<td>50</td>
</tr>
<tr>
<td>4.1. Pre-processing</td>
<td>50</td>
</tr>
<tr>
<td>4.1.1. Radial Distortion Correction</td>
<td>50</td>
</tr>
<tr>
<td>4.1.2. Selecting the Most-Focused Region</td>
<td>52</td>
</tr>
<tr>
<td>4.2. Image Segmentation and Fuzzification</td>
<td>54</td>
</tr>
<tr>
<td>4.3. Cell Counting</td>
<td>60</td>
</tr>
<tr>
<td>4.3.1. Affinity Function</td>
<td>64</td>
</tr>
<tr>
<td>4.3.2. Homogeneity Function</td>
<td>66</td>
</tr>
<tr>
<td>4.3.3. Cell Recognition</td>
<td>68</td>
</tr>
<tr>
<td>5. RESULTS AND DISCUSSION</td>
<td>74</td>
</tr>
<tr>
<td>5.1. Preprocessing</td>
<td>74</td>
</tr>
<tr>
<td>5.2. Image Segmentation and Fuzzification</td>
<td>77</td>
</tr>
<tr>
<td>5.3. Cell Recognition</td>
<td>79</td>
</tr>
<tr>
<td>5.4. Performance</td>
<td>80</td>
</tr>
<tr>
<td>6. SUMMARY, CONCLUSIONS, AND RECOMMENDATIONS</td>
<td>86</td>
</tr>
<tr>
<td>6.1. Summary</td>
<td>86</td>
</tr>
<tr>
<td>6.2. Conclusions</td>
<td>86</td>
</tr>
<tr>
<td>6.3. Recommendations</td>
<td>87</td>
</tr>
<tr>
<td>Appendix 1. Matlab Codes</td>
<td>89</td>
</tr>
<tr>
<td>Appendix 2. The Cell Images</td>
<td>103</td>
</tr>
<tr>
<td>References</td>
<td>107</td>
</tr>
</tbody>
</table>
# LIST OF TABLES

<table>
<thead>
<tr>
<th>Tables</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Table 5.1</td>
<td>The comparison of the elapsed time for different focusing criterion functions</td>
</tr>
<tr>
<td>2. Table 5.2</td>
<td>The confusion matrix for the cell recognition algorithm</td>
</tr>
<tr>
<td>3. Table 5.3</td>
<td>Evaluation of the algorithm developed in this work</td>
</tr>
</tbody>
</table>
## LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figures</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Figure 2.1 The complex attachment developed at UCB to convert a cell phone to a microscope</td>
<td>6</td>
</tr>
<tr>
<td>2.</td>
<td>Figure 2.2 The lens-free attachment developed at UCLA for holography imaging of blood samples</td>
<td>6</td>
</tr>
<tr>
<td>3.</td>
<td>Figure 2.3 The system diagram of the cell-phone microscope developed at CBST</td>
<td>7</td>
</tr>
<tr>
<td>4.</td>
<td>Figure 2.4 An iPhone 2G cell phone equipped with a ball lens to convert it to a microscope, developed at CBST</td>
<td>8</td>
</tr>
<tr>
<td>5.</td>
<td>Figure 3.1 Representation of radial distortion on a rectangular wire mesh</td>
<td>9</td>
</tr>
<tr>
<td>6.</td>
<td>Figure 3.2 Gray-scale variation in digital images</td>
<td>19</td>
</tr>
<tr>
<td>7.</td>
<td>Figure 3.3 The result of assignment of different thresholds in image segmentation</td>
<td>22</td>
</tr>
<tr>
<td>8.</td>
<td>Figure 3.4 The impact of gray-scale contrast in image segmentation</td>
<td>24</td>
</tr>
<tr>
<td>9.</td>
<td>Figure 3.5 The segmented image of the scanned book (Figure 3.4.a) obtained by applying the threshold T = 215</td>
<td>25</td>
</tr>
<tr>
<td>10.</td>
<td>Figure 3.6 A plot of the criterion function in the ME method</td>
<td>30</td>
</tr>
<tr>
<td>11.</td>
<td>Figure 3.7 Breaking a histogram into two Gaussian distributions, representing the gray-scale distribution of the objects and background</td>
<td>31</td>
</tr>
<tr>
<td>12.</td>
<td>Figure 3.8 A chest x-ray image</td>
<td>34</td>
</tr>
<tr>
<td>13.</td>
<td>Figure 3.9 The result of segmentation using a threshold to illustrate the effect of inhomogeneity in the gray scale values of the objects</td>
<td>36</td>
</tr>
</tbody>
</table>
14. Figure 3.10 A linear fuzzy function that is extended from the minimum to the maximum gray scale values ............ 37

15. Figure 3.11 A fuzzy function that is linear in a limited range, called the fuzzy range .................................................. 39

16. Figure 3.12 An illustration of the fuzzy region in the histogram .......... 40

17. Figure 3.13 A step-wise fuzzy function .................................................. 42

18. Figure 3.14 An illustration of (a) broken and (b) connected blood cells after the segmentation process .................................. 44

19. Figure 3.15 A segmented image that contains 19 similar rounded objects ................................................................. 45

20. Figure 3.16 Illustration of the masking method ........................................ 47

21. Figure 3.17 Identifying cells that are not circular .............................. 48

22. Figure 3.18 Over-counting, when a small mask is applied .................. 49

23. Figure 3.19 A guard distance \((d)\) to provide minimum distance between the centers of the detected cells ...................... 49

24. Figure 4.1 An image of the pattern of parallel dark and bright straight bars captured by the cell-phone microscope developed at CBST .......................... 51

25. Figure 4.2 An image captured by the cell-phone microscope developed at CBST from a blood sample ...................... 55

26. Figure 4.3 Effect of uneven illumination ............................................. 56

27. Figure 4.4 The effect of illumination variation on the histogram ........ 57

28. Figure 4.5 Partitioning an image into nine sub-images and assigning a distinct threshold to each of them .................. 59

29. Figure 4.6 The threshold surface for the image shown in Figure 4.3 ......................... 60
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.7</td>
<td>A small portion of the red cell image captured by the cell-phone microscope developed at CBST</td>
</tr>
<tr>
<td>4.8</td>
<td>A link in a fuzzy scene</td>
</tr>
<tr>
<td>4.9</td>
<td>A one-pixel-thick ring mask on a fuzzy scene</td>
</tr>
<tr>
<td>4.10</td>
<td>A situation in which the mask spans across cells</td>
</tr>
<tr>
<td>4.11</td>
<td>The mask sweeping over the fuzzy scene</td>
</tr>
<tr>
<td>4.12</td>
<td>The feature space for cell recognition formed by the fuzzy functions, affinity and homogeneity</td>
</tr>
<tr>
<td>4.13</td>
<td>A flowchart of the algorithm that is developed in this thesis (part 1)</td>
</tr>
<tr>
<td>4.14</td>
<td>A flowchart of the algorithm that is developed in this thesis (part 2)</td>
</tr>
<tr>
<td>5.1</td>
<td>The distortion-corrected image of the bright and dark bars depicted in Figure 4.1</td>
</tr>
<tr>
<td>5.2</td>
<td>The output images of the focusing criterion functions using different methods</td>
</tr>
<tr>
<td>5.3</td>
<td>The fuzzified image of the blood cell image shown in Figure 5.2(d)</td>
</tr>
<tr>
<td>5.4</td>
<td>A guard distance as long as the diameter of the small masks</td>
</tr>
<tr>
<td>5.5</td>
<td>Finding the optimum affinity and homogeneity thresholds for the cell detection region in the feature space</td>
</tr>
<tr>
<td>A1.1</td>
<td>Blood cell image #1</td>
</tr>
<tr>
<td>A1.2</td>
<td>Blood cell image #2</td>
</tr>
<tr>
<td>A1.3</td>
<td>Blood cell image #3</td>
</tr>
<tr>
<td>A1.4</td>
<td>Blood cell image #4</td>
</tr>
<tr>
<td>A1.5</td>
<td>Blood cell image #5</td>
</tr>
</tbody>
</table>
48. Figure A1.6  Blood cell image #6 ................................................. 105
49. Figure A1.7  Blood cell image #7 ................................................. 106
Chapter 1

INTRODUCTION

1.1. Overview

Healthcare costs in the United States are growing and are projected to increase even more steeply. The number of elderly people is increasing, and the elderly typically require higher levels of healthcare [1]. It is desirable to reduce healthcare costs, while maintaining or even improving healthcare quality. Continuous medical monitoring to alert patients of problems can reduce healthcare costs, since it can prevent diseases from becoming severe and avoids hospitalization [2].

The development of portable biomedical devices can improve the performance of healthcare and reduce costs. Technology advances in the digital era can be utilized in many medical fields. Such advances have included lightweight, portable computers with high computational ability. Nowadays, smart phones, which are pocket-sized multi-purpose computers with data transmission ability, can be found everywhere. Smart phones can bring medical screening to every patient’s pocket when the phones are suitably equipped with medical sensors and analyzing software.

This thesis contributes toward the conversion of a cell phone to a clinical microscope for blood cell counting purposes. By affixing an appropriate attachment to a cell phone, it can capture images of the blood cells from a blood smear. The goal then is to process the blood cell image in order to count the number of blood cells. The image
processing software may be run on a Personal Computer (PC) that receives images from a cell phone. If a smart phone is equipped with the microscope-conversion attachment and suitable software, the entire process of image capture and analysis can take place on the phone.

1.2. Blood Cell Count

As one of the main medical diagnostic and monitoring tools, a Complete Blood Count (CBC) is a very common test performed in pathology laboratories. Physicians request a CBC to assess many symptoms or diseases. The result can reflect problems with fluid volume or loss of blood. A CBC test consists of White Blood Cell (WBC) evaluation, Red Blood Cell (RBC) evaluation, and Platelet evaluation. One of the tests in the RBC evaluation is RBC count [3].

This thesis focuses only on acquiring a RBC count using a cell-phone microscope. This test can help diagnose anemia and other conditions affecting red blood cells. The following are additional conditions for which an RBC count may be performed [4]:

- Alport syndrome
- Drug-induced immune hemolytic anemia
- Hemolytic anemia due to G6PD deficiency
- Hereditary anemias, such as thalassemia
- Idiopathic autoimmune hemolytic anemia
- Immune hemolytic anemia
- Macroglobulinemia of Waldenstrom
- Paroxysmal Nocturnal Hemoglobinuria (PNH)
- Primary myelofibrosis

In most of the diseases mentioned above, a clinical examination is needed for any further medical decisions. In other words, it is not claimed that the results provided by the cell-phone microscope can be substituted for the results of a clinical examination. The initial use of a cell-phone measurement is to alert the patient to obtain a clinical blood test to verify the cell phone’s results [5]. This alerting capability is especially beneficial to the patients who need nearly continuous RBC monitoring, such as cancer patients who are undergoing chemotherapy.

One form of anemia results from a reduction in the number of red blood cells. Since most cancer therapies destroy the cells that grow at a fast rate, and since red blood cells have relatively rapid growth rates, they are often affected [5]. This shortage in red blood cells causes a decrease of oxygen level in the blood, which results in tiredness. The cell-phone microscope device can help chemotherapy patients measure their RBC count several times a day, and thus alert them if it reaches a dangerous level.

1.3. Organization of the Remaining Chapters

Chapter 2 provides a background on cell-phone microscope development at the Center for Biophotonics Science and Technology (CBST) and elsewhere. Other competitor cell-phone microscopes, which were previously developed, are described,
followed by the description of the cell phone attachment that is used in this work.

Chapter 3 is a review of the image processing tasks that are essential for blood cell counting. The methods proposed for radial distortion correction, automatic searching for in-focus regions, image segmentation, and object counting are reviewed. Chapter 4 discusses the algorithm development for this project. The algorithm development includes the three modules of the algorithm: pre-processing, image segmentation, and object recognition. In Chapter 5, the results are provided and discussed. Chapter 6 is a summary of this work, along with conclusions and recommendations for future work.
Chapter 2

BACKGROUND

Utilizing cell phones that contain digital cameras as portable devices for medical imaging purposes is becoming attractive, since such cell phones are the most ubiquitous optical sensors [6]. An appropriate attachment that provides suitable magnification can convert the cell phone into a microscope. The magnified image then is captured by the digital camera in the cell phone.

Moreover, recent advances in smart phones allow them to run complex software. Thus, both medical imaging capability and image analyzing software can be combined in a single device. The rest of this chapter briefly reviews of the previous scholarly work to develop cell-phone microscopes, followed by the cell-phone microscope for this work.

2.1. Previous Work

In a project at the University of California at Berkeley (UCB), an attachment is designed to enable cell phones to capture and transmit high-resolution raw microscope images to expert readers. The attachment employs the eyepieces normally found in standard inexpensive microscopes. The target is illuminated by white and blue light-emitting diodes (LEDs) for bright-field and fluorescence microscopy, respectively [7]. Figure 2.1 depicts this complex UCB attachment.
Unlike the complex UCB attachment, a light, compact, and lens-free attachment is used for holographic microscopy at the University of California at Los Angeles (UCLA). Holographic imaging is accomplished by recording the interference of light scattered from an object with coherent reference light [8]. A simple LED is utilized to illuminate the sample. As light waves pass through blood cell samples toward the complementary metal-oxide semiconductor (CMOS) sensor, a hologram of the cells is created [9]. In the Figure 2.2 shows the attachment developed at UCLA.

![Figure 2.1. The complex attachment developed at UCB to convert a cell phone to a microscope [7].](image1)

![Figure 2.2. The lens-free attachment developed at UCLA for holography imaging of blood samples [9].](image2)
2.2. This Work

At the CBST, a one-millimeter diameter ball lens (Edmund Optics, Barrington, NJ) is utilized to obtain magnification comparable to that of commercial laboratory microscopes. The ball lens is mounted inside a small ring of black rubber, and the rubber then is attached to the cell phone by means of double-sided tape. A white-light LED also is applied to provide the required illumination for imaging [6]. Figure 2.3 illustrates the position of light source, blood sample, and ball lens in front of the cell phone.

![Diagram of cell phone microscope](image)

**Cell phone microscope**

Figure 2.3. The system diagram of the cell-phone microscope developed at CBST [6].

An iPhone 2G smart phone (Apple Inc., Cupertino, CA) was converted to a microscope and loaded with image analyzing software for pre- and post-image processing tasks. This phone possesses a 2-megapixel CMOS sensor panel (Micron Technology, Inc., Boise, ID), which provides images with 1200 pixels by 1600 pixels resolution. The resulting cell-phone microscope utilized in this project is depicted in Figure 2.4.
Figure 2.4. An iPhone 2G cell phone equipped with a ball lens to convert it to a microscope, developed at CBST.

The advantage of the CBST approach is that the 1-mm ball lens simplifies the conversion of a cell phone to a microscope. However, the image provided by CBST’s approach suffers from optical aberrations. The ball lens focuses only the central portion of the field-of-view. Moreover, the image suffers from radial distortion. The usable field-of-view without post image processing is 150 $\mu m \times 150 \mu m$, which is extended to 350 $\mu m \times 350 \mu m$, by applying post processing [6].
3.1. Radial Distortion Correction

As cited in Chapter 2, a spherical lens causes radial distortion of the image [6]. In optics, distortion occurs when the magnification is not uniform over the entire field-of-view. There are two types of the radial distortion: Pincushion and Barrel. Figure 3.1 shows these distortions. Barrel distortion arises when the magnification decreases toward the edge of the field (Figure 3.1.b), while in pincushion distortion, there is greater magnification at the borders (Figure 3.1.c) [10].

![Figure 3.1. Representation of radial distortion on a rectangular wire mesh. (a) Undistorted, (b) Barrel distortion, (c) Pin-cushion distortion [10].](image)

Geometric distortion is a significant problem in digital image processing, especially when it comes to measurements in scientific image analysis [11]. Among several geometric distortions, radial distortion is predominant [12]. Achieving a formula that maps a point in the distorted image into the corresponding point in the original image
is desired. For correction of radial distortion, several mathematical models are proposed to perform such mapping.

The polynomial model (PM) is proposed for transferring a point in the distorted image back to its original position in the undistorted image. Let \((x_d, y_d)\) be the Cartesian coordinates of a point in a distorted image that is associated with \((x_u, y_u)\) in the undistorted scene; then, the PM for this transform is

\[
r_u = r_d (1 + \lambda_1 r_d^2 + \lambda_2 r_d^4 + \cdots),
\]

where \(r_d\) and \(r_u\) are radial distances of the distorted point \((x_d, y_d)\) and undistorted point \((x_u, y_u)\) from the distortion center, respectively. The distortion constants are noted as \(\lambda_s\), for \(s = 1, 2, \ldots\). As an alternative to the PM model, the division model (DM) for mapping from the distorted image to the undistorted one is described as

\[
r_u = \frac{r_d}{1 + \lambda_1 r_d^2 + \lambda_2 r_d^4 + \cdots}.
\]

The DM is preferred to PM, since it can express high distortions at much lower order. In practice, a second-order model is utilized [11].

To apply second-order models of the proposed types for distortion correction, the center of distortion and the distortion constant are required. As shown in Figure 3.1, usually an image of a pattern of straight lines is utilized to measure the distortion type
and level. This pattern is helpful for calculating the constant factors in the mapping formulas. In the distorted patterns in Figure 3.1, the distortion centers are at the geometric centers of the patterns. The coordinates of the distortion center are important, because the magnification of the lens is rotationally invariant with respect to the distortion center [13].

Some approaches are proposed to obtain distortion parameters. A straight line in an undistorted image,

\[ y = kx + b, \]  

becomes an arc,

\[ ay'^2 + bx'^2 + a'y' + b'x' + c = 0, \]

in the distorted scene, which is a portion of a circle. Circle-fitting techniques, such as direct least squares (DL), and Levenberg-Marquardt (LM), usually are utilized to estimate the distortion parameters [11].

3.2. Focus Criterion Functions

In optics, focus refers to the convergence of light rays coming from a point on the object to a point on the image. In photography, it is desirable that the image forms where the incoming light from the target is focused [14]. In commercial digital cameras, the lens
can be adjusted manually or automatically to capture the image of the targeted object on the sensor array of the camera. In a defocused image, the scene is blurred, and boundaries of objects in the scene are not accurately distinguishable.

Because of the shape of the spherical lens utilized in this work, only a portion of the image is focused [6] and consequently suitable for cell recognition. This section presents a review of methods for automatic focus adjustment. These methods are used to find the regions of the image that are sufficiently focused.

Several methods are proposed in the literature, and some already are used in commercial cameras as auto-focusing functions. Nowadays, most digital cameras have auto-focusing. This technology provides the best focus adjustment for the users by simply clicking the shutter button half-way. This function in digital cameras allows unskilled users to take photos without any manual adjustments.

Various functions are proposed to evaluate the level of sharpness and contrast in digital images. These functions return an index of focus from the input image. An image is focused when the edges of objects in it are sharp. Unfocused images are vague and not easily distinguishable, analogous to applying an averaging (low-pass) filter, which reduces the high frequency components of the image. The proportion of high frequency to low frequency components in an image can be employed as a measure of sharpness and consequently a focus index. This index can be provided by the Fourier Transform (FT) of an image [15]. This index can be utilized for focus adjustment by finding the index’s maximum.
Other approaches also are proposed to measure the sharpness of images at different lens positions. These methods propose criterion functions to evaluate the image’s contrast. The digital images are inputs to these criterion functions, and the criterion functions return values to be applied as focus indices. Such functions must comply with the following criteria [15]:

1. Reaching the global minimum or maximum at the utmost focused position

2. Monotonic on all sides of the global minimum or maximum.

Both of these criteria imply that the function must have no local minima or maxima. This property allows the algorithm to continue searching to reach a minimum or maximum. Then, the lens can be set to that point to capture the most focused image. Some of the criterion functions proposed in the literature now are reviewed.

3.2.1. Gray Variance (GV) [14]

By dividing the histogram of a gray-scale image by the total number of its pixels, the normalized histogram is achieved with the properties of a probability mass function (PMF). The images that are achieved at different lens positions are assumed to be samples of a random process. Thus, every outcome of this random process is a random experiment with an exclusive PMF. One of the significant properties of a PMF that provides a measure of distribution is variance. The variance shows how widely a PMF is distributed.
Focused images are expected to have the highest possible gray-scale variance in comparison with defocused ones. Sharp edges provide significant variation in gray levels between neighboring pixels. These significant variations ultimately lead to greater variance in the histogram. Theoretically, the gray variance (GV) operator returns the global maximum at the most focused position and behaves monotonically before and after reaching that point.

3.2.2. Tenengrad (TEN) [15]

The variations of gray-scale values between neighboring pixels can be calculated and accumulated to return a focus index. A gradient-based operator called Tenengrad (TEN) is proposed to measure such variations in images by applying a gradient operator. The focus index is the sum of its magnitudes,

$$\sum |\nabla f(x,y)| = \sum \sqrt{f_x^2 + f_y^2},$$  \hspace{1cm} (5)

where $f(x,y)$ is a 2D function, and $f_x$ and $f_y$ are gradients of the 2D function $f$ in the $x$ and $y$ directions, respectively. The vertical and horizontal gradients are achieved by convolving the image with the Sobel gradient operators,

$$i_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}, \quad i_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix},$$  \hspace{1cm} (6)
resulting in

$$S(x, y) = \sqrt{(i_x * f(x, y))^2 + (i_y * f(x, y))^2}, \quad (7)$$

where $S(x, y)$ is a matrix consisting of the square root of the addition of squared partial gradients. Since the Sobel operator is a $3 \times 3$ matrix, then for an image that is an $n \times n$ matrix, the size of $S$ is $(n + 2) \times (n + 2)$. To come up with a single value as the focus index, all the elements of $S$ are accumulated. Therefore, the TEN focus index is

$$\text{TEN} = \sum_x \sum_y S(x, y). \quad (8)$$

3.2.3. Sum Modified Laplacian (SML) [15]

The Laplacian is the second-order derivative of a 2D function. The Laplacian operator returns the maximum magnitude when it is applied on a local maximum or minimum. Thus, it is expected that the Laplacian can be utilized to provide a measure of difference between neighboring pixels of an image. The representation of the Laplacian in Cartesian coordinates is

$$\nabla^2 f(x, y) = \frac{\partial^2 f(x, y)}{\partial x^2} + \frac{\partial^2 f(x, y)}{\partial y^2}. \quad (9)$$
However, only the magnitude of the Laplacian is needed in this application, which is

\[ |\nabla^2 f(x, y)| = \left| \frac{\partial^2 f(x, y)}{\partial x^2} \right| + \left| \frac{\partial^2 f(x, y)}{\partial y^2} \right|. \tag{10} \]

The sum modified Laplacian (SML) criterion function calculates the magnitude of a modified Laplacian on the image \( f(x, y) \). The \( x \) and \( y \) components of this function are

\[ \left| \frac{\partial^2 f(x, y)}{\partial x^2} \right| = |2f(x, y) - f(x + 1, y) - f(x - 1, y)| \tag{11} \]

and

\[ \left| \frac{\partial^2 f(x, y)}{\partial y^2} \right| = |2f(x, y) - f(x, y - 1) - f(x, y + 1)|, \tag{12} \]

respectively. The focus index is calculated by accumulating the elements of the resulting matrix, which is

\[ SML = \sum_x \sum_y \left( \left| \frac{\partial^2 f(x, y)}{\partial x^2} \right| + \left| \frac{\partial^2 f(x, y)}{\partial y^2} \right| \right)^2. \tag{13} \]

3.2.4. Sum Modulus Difference (SMD) [15]

The sum modulus difference (SMD) is the sum of differences of the gray scale values between neighboring pixels along the vertical and horizontal directions. These
summations of the directional variations are

\[
SMD_x = \sum_\limits{x} \sum_\limits{y} |f(x,y) - f(x,y - 1)|
\]  \hspace{1cm} (14)

for the vertical direction and

\[
SMD_y = \sum_\limits{x} \sum_\limits{y} |f(x,y) - f(x + 1,y)|
\]  \hspace{1cm} (15)

for the horizontal direction. The SMD focus index is calculated by simply adding the both directional variations,

\[
SMD = SMD_x + SMD_y.
\]  \hspace{1cm} (16)

3.3. Image Segmentation

Segmentation means labeling pixels in an image based on their values and positions. The generic definition of image segmentation is provided in [16]:

“Segmentation subdivides an image to its constituent regions or objects. The level to which the subdivision is carried depends on the problem being solved. That is, segmentation should stop when the object of interest in an application has been isolated.”

Numerous methods of segmentation have been proposed from the early years of computer vision. However, it is still one of the most difficult tasks in image processing
In this work, since extracting the cells from the background is desired for cell recognition, a review of some image segmentation methods is presented.

Objects in an image are recognizable because they have different gray scale values or colors. Figure 3.2 illustrates this contrast between an object and its background. In Figure 3.2(a), a gray-scale image containing a clock is shown. Figure 3.2(b) depicts the values of the pixels of a portion of the image, showing the gray scale contrast between the object (clock) and background. As shown in Figure 3.2(b), there is a narrow variation in the gray scale values of the pixels that belong to either background or object, which is considered as noise. But, a significant contrast exists between the two regions associated with the clock and the background. Thus, there are two main factors that contribute to image segmentation: similarity of the pixels of an object and discontinuity at the boundary of the object [16].

Two different approaches for image segmentation are in common use: edge-based and region-based methods. The edge-based approach attempts to extract the boundaries of the objects or regions, since the edge is recognizable by application of a differential operator. The Sobel, Robert, Prewitt, Laplacian, and Canny operators are mostly used for edge extraction in image processing applications [17]. If the interest is to extract the whole region of an object, a region-based method is applied. A region-based approach can use any of a variety of strategies for pixel classification. Upcoming sections discuss some edge-based and region-based methods.
Figure 3.2. Gray-scale variation in digital images. (a) A gray-scale image [18] that contains some objects. (b) Pixel-wise representation of a small portion of the image in (a). This image provides the pixel values to point out the abrupt change in gray scale values at the boundaries.

3.3.1. Edge-Based Methods

An edge-based operator is utilized to locate abrupt changes in the values of neighboring pixels. The first derivative of a graph returns its slope at every point of the graph, which is proportional to the sharpness of the change. Moving in the vertical or horizontal direction in an image results in a 1D graph. In image processing, a 2D gradient-based operator is applied to detect un-smooth transitions in the pixels’ gray scale values.

The Sobel, Roberts, and Prewitt algorithms are first-order gradient-based operators that respond to discontinuities in gray-scale images. The weights that these
operators apply for each direction, vertical, horizontal, and diagonal, set them apart. In all of these methods, if the calculated slope is higher than a threshold, the operator detects an edge. The major weakness of these gradient-based methods is their sensitivity to noise in images. As a result, the edge-extracted images do not produce continuous lines as the boundaries of the object for a noisy image.

The Laplacian operator applies a second-order gradient on images to detect the local maxima or minima as boundaries. This method is highly responsive to corners and lines. This operator, denoted in (9), was previously (Section 3.2.3) introduced as a way to measure the contrast in images.

To overcome the disadvantages with the previous operators, Canny proposed an operator to maximize the signal-to-noise ratio (SNR) of the segmented image. Canny’s operator has a lower possibility of being confused by non-edge points.

3.3.2. Region-Based Methods

Given that the objects and regions in an image are recognizable based upon contrast, it is possible to segment images objects by appropriately classifying pixels in different gray-scale ranges. The gray scale values of pixels representing an object or region are expected to be bounded in certain ranges. Region-based methods try to find these appropriate gray-scale ranges to segment images into its content objects or regions.

A straightforward strategy to specify the gray-scale ranges for different objects or regions in the images is thresholding. By setting a threshold, the full range of gray scale
values is partitioned into two ranges. By setting more than one threshold, the gray-scale range can be partitioned into more than two ranges. Those pixels having gray scale values within a certain range are assigned to one class. Assume that the gray scale value of the $i$-th pixel in an image is represented by $x_i$. A threshold ($T$) segments the image into two separate classes of pixels ($C_0$ and $C_1$) based on their gray scale values,

$$x_i \in \begin{cases} C_0 & x \leq T \\ C_1 & x > T \end{cases}.$$  \hspace{1cm} (17)

Different threshold values cause different classification results. Thus, finding an optimum threshold value is desired. If $k$ types of objects or regions have exclusive ranges of gray scale values, then there are $k - 1$ optimum thresholds to partition the pixels that belong to the different objects or regions. As a convention, in bi-level segmentation ($k = 2$), the pixels of two classes are represented by the highest (bright) and lowest (dark) possible gray scale values.

Figure 3.3 illustrates how applying different threshold values affects the result of segmentation. As seen in Figure 3.3(a), the majority of pixels that represent the tank are darker than its background. Therefore, an optimum threshold value should be in between the ranges of gray scale values of the background and tank. Figures 3.3(b), (c), and (d) show under-thresholding, over-thresholding, and optimum thresholding, respectively. Here, the threshold values that result in under-thresholding and over-thresholding are chosen arbitrarily greater and smaller than the optimum threshold value. The optimum
threshold can be achieved by applying the Otsu method, which is described further in this chapter.

Determining the optimal level for thresholding is the major task in image segmentation. If the image is composed of distinct ranges of gray scale values, associated with the objects and background, the histogram includes two main separate peaks [19]. The optimum threshold should be chosen in the valley between the main peaks.

![Figure 3.3](image-url)

Figure 3.3. The result of assignment of different thresholds in image segmentation. (a) A gray-scale image, depicting a tank [18]. The tank is darker than its surroundings, and the gray scale range of the image is extended from 0 (darkest) to 255 (brightest). (b) Under-thresholding, T = 25, (c) Over-thresholding, T = 153. (d) Optimal segmentation, T = 76.
In images with high contrast between the objects and background, picking a threshold for optimal segmentation is not difficult. Figure 3.4 illustrates how the high contrast between letters and background separates the main peaks in the histogram. Figure 3.4(a) shows a scanned page of a book, followed by the corresponding histogram. The main peaks, associated with the letters (objects) and background, are separated and easily recognizable in the histogram. An appropriate threshold can be determined by picking a gray-scale value in the valley between the main peaks. Figure 3.5 shows the result of thresholding, where the assigned threshold is 215. Thresholding is a part of the letter recognition process that is utilized for converting printed documents to digital media. In the above example of letter extraction, the threshold was assigned by visual inspection. In other words, by looking at the histogram, a threshold between the main peaks was picked. But in most applications, automatic optimal thresholding is desired, since the computer must perform the recognition by itself.

Several approaches are proposed toward the automatic assignment of thresholds that are classified into global and local thresholding. Global thresholding assigns a threshold for a gray-scale image based on the information provided by the histogram of the whole scene, while the local approach considers only a small region of the image to classify the pixels of that small part [19].
The first official step in the great movement to build and operate railroads, having cars drawn by engines, was an Act of the British Parliament in 1821 for the construction of the Stockton & Darlington railway.

Figure 3.4. The impact of gray-scale contrast in image segmentation. (a) A gray-scale image obtained by a digital scanner [20]; the contrast between the letters and background is important. (b) The histogram of the image in (a). Two main peaks, around 85 and 235, are the approximate statistical means for the letters’ and background gray-scale ranges, respectively.

3.3.2.1. Otsu Method

Otsu [21] proposed an iterative method to assign the optimum threshold based on the histogram. In this method, the histogram is converted to a probability mass function.
The first official step in the great movement to build and operate railroads, having cars drawn by engines, was an Act of the British Parliament in 1821 for the construction of the Stockton & Darlington railway.

Figure 3.5. The segmented image of the scanned book (Figure 3.4.a) obtained by applying the threshold $T = 215$.

(PMF) by dividing by the total number of pixels in the image. Suppose there are $L$ possible gray scale values in an image, and each pixel can take a gray scale value from 0 to $L - 1$. Also, let $N$ be the total number of pixels in the image and $n_i$ be the frequency in the histogram for the $i$-th gray scale level. Then,

$$n_1 + n_2 + \cdots + n_L = N,$$

and thus,

$$p_i = \frac{n_i}{N}, \quad p_i \geq 0, \quad \sum_{i=0}^{L-1} p_i = 1,$$

where $p_i$ is the $i$-th component of the normalized frequency in the PMF of the image.

Now suppose that $T$ is the threshold value. Every pixel with a gray scale value smaller
than or equal to $T$ is classified in class $C_0$ and the rest in class $C_1$. Thus, the probabilities of being classified into $C_0$ or $C_1$ are

$$\omega_0(T) = \sum_{i=0}^{T} p_i,$$  \hspace{1cm} (20)

and

$$\omega_1(T) = \sum_{i=T+1}^{L} p_i = 1 - \omega_0(T),$$  \hspace{1cm} (21)

respectively.

The parts of the PMF, associated with each of the two classes can be interpreted as partial PMFs by dividing each part by the probability of that class. These partial PMFs, denoted $P_0$ and $P_1$ for $C_0$ and $C_1$, respectively, are

$$P_0 = \frac{p_i}{\omega_0}, \quad 0 \leq i \leq T,$$  \hspace{1cm} (22)

and

$$P_1 = \frac{p_i}{\omega_1}, \quad T < i \leq L.$$  \hspace{1cm} (23)
The statistical moments of these partial PMFs are

\[ \mu_0 = \sum_{i=0}^{T} i p_0 = \sum_{i=0}^{T} \frac{i p_i}{\omega_0}, \]

(24)

\[ \mu_1 = \sum_{i=T+1}^{L-1} i p_1 = \sum_{i=T+1}^{L-1} \frac{i p_i}{\omega_1}, \]

(25)

\[ \sigma_0^2 = \sum_{i=1}^{T} (i - \mu_0)^2 p_0 = \sum_{i=1}^{T} \frac{(i - \mu_0)^2 p_i}{\omega_0}, \]

(26)

\[ \sigma_1^2 = \sum_{i=T+1}^{L-1} (i - \mu_1)^2 p_1 = \sum_{i=T+1}^{L-1} \frac{(i - \mu_1)^2 p_i}{\omega_1}, \]

(27)

where \( \mu \) and \( \sigma^2 \) are the statistical mean and variance.

Otsu has proposed between-class variance \( \sigma_B^2 \) and within-class variance \( \sigma_W^2 \), respectively, as

\[ \sigma_B^2 = \omega_0 (\mu_0 - \mu_{\text{Total}})^2 + \omega_1 (\mu_1 - \mu_{\text{Total}})^2 = \omega_0 \omega_1 (\mu_1 - \mu_0)^2, \]

(28)

and

\[ \sigma_W^2 = \omega_0 \sigma_0^2 + \omega_1 \sigma_1^2, \]

(29)
where $\mu_{\text{total}}$ is the statistical average of the whole PMF, calculated as

$$\mu_T = \sum_{i=0}^{L} i p_i .$$

Since the maximum between-class and minimum within-class variances are desired, a measure of separation is defined as

$$\eta(T) = \frac{\sigma_B^2}{\sigma_W^2} .$$

The iterative algorithm is to calculate $\eta(T)$ for every gray scale value and return the optimum threshold when $\eta$ has its maximum value, denoted as [21]

$$\text{Threshold (Otsu method)} = \arg \max_{\tau \in \{0, \ldots, L-1\}} \eta(T) .$$

3.3.2.2. Minimum Error Method [22]

If the histogram is composed of two main peaks, associated with objects and background, the threshold can be assigned at the deepest point between them. This idea has driven a method called minimum error (ME) thresholding. In the ME method, two main peaks are approximated by two Gaussian distributions, with each Gaussian distribution scaled to the probability of its class. Then, the optimum threshold is where the two weighted Gaussian distributions meet. Theoretically, this point is the optimum threshold, since it results the minimum error of classification.
In (22) and (23) and following, two partial PMFs are introduced, and their statistical moments are calculated. Using the calculated mean and variances, the two partial PMFs are approximated by two Gaussian distributions. These new PMFs, denoted $P'_0$ and $P'_1$ for the $C_0$ and $C_1$ classes, respectively, are defined as

$$P'_0 = \frac{1}{\sqrt{2\pi\sigma_0}} \exp\left(-\frac{(i - \mu_0)^2}{2\sigma_0^2}\right),$$  \hspace{1cm} (33)$$

and

$$P'_1 = \frac{1}{\sqrt{2\pi\sigma_1}} \exp\left(-\frac{(i - \mu_1)^2}{2\sigma_1^2}\right).$$  \hspace{1cm} (34)$$

To find the cross-over point, it is necessary to rescale the Gaussian partial PMFs according to their actual weights, which are $\omega_0$ and $\omega_1$. Then, the $i$ that satisfies the equation

$$\omega_0 P'_0 = \omega_1 P'_1$$  \hspace{1cm} (35)$$
is the cross-over point.

By manipulation and applying the natural logarithm to both sides of (35), the cross-over point is achieved, which is a function of the threshold $T$. Since $T$ is a variable in the range of gray scale values, a criterion function,

$$J(T) = 1 + 2[\omega_0(T) \log \sigma_0(T) + \omega_1(T) \log \sigma_1(T)] - 2[\omega_0(T) \log \omega_0(T) + \omega_1(T) \log \omega_1(T)],$$  \hspace{1cm} (36)$$
is proposed to find the threshold with the minimum classification error. The optimum threshold is calculated as [22]

$$\text{Threshold (ME method)} = \arg \min_{T \in \{0,\ldots,L-1\}} J(T). \quad (37)$$

A graph of the criterion function over all the gray scale values and the minimum point for an image are depicted in Figure 3.6.

![Figure 3.6](image)

Figure 3.6. A plot of the criterion function in the ME method. This function reaches its minimum for the gray scale value 119. In some gray-scale regions, the function has no value, since the histogram is zero in those regions, and consequently, the criterion function cannot be defined in those regions.
Since only the first and second order moments are utilized to simulate the partial PMFs, the actual histogram is not necessarily equal to the sum of the weighted partial PMFs. Figure 3.7 depicts such a mismatch. In Figure 3.7, two partial Gaussian PMFs are shown along with the original histogram. Where two weighted partial PMFs meet is calculated by the ME method to be the optimum threshold.

![Histogram diagram](image)

Figure 3.7. Breaking a histogram into two Gaussian distributions, representing the gray-scale distribution of the objects and background. The sum of the two Gaussian models does not exactly equal the original.
3.3.2.3. Fuzzy C-Mean Method

Nowadays, fuzzy theory is widely accepted in many classification problems. Fuzzy methods are applied for image segmentation to cluster the gray scale values into different classes. To do so, a fuzzy criterion function is needed to measure the dependence (membership value) of a gray scale value to each class. Many criterion functions are proposed for fuzzy-based image segmentation. To comply with the fuzzy subset theory, the sum of the membership degrees associated with the different classes must be one. A pixel is classified to the class that has the highest membership value for its gray scale value.

The fuzzy C-mean [23] classification algorithm is used not only in image segmentation but also in many other classification applications as an unsupervised learning method. In image segmentation, the input data are the histogram values for the image. The goal is to classify the \( L \) gray scale values into \( c \) classes. For each class, there is a center point, which is analogous to the statistical mean in the Otsu and ME methods.

The fuzzy C-mean algorithm starts with \( c \) arbitrary centers for classes and goes through the training process to assign a membership value to each gray scale value associated with each class. Using a criterion function, the classification error is calculated, and it results in another set of centers of classes. The membership value of the
The \( k \)-th pixel belonging to the \( i \)-th class \( u_{ik} \) is calculated as

\[
u_{ik} = \left( \sum_{j=1}^{c} \left( \frac{D_{ik}}{D_{jk}} \right)^{\frac{2}{m-1}} \right)^{-1} \quad \forall i, k, \quad m > 1,
\]

where \( D_{ik} \) is the Euclidean distance between the gray scale value of the \( k \)-th pixel and the center gray scale value of the \( i \)-th class, and \( m \) is a constant that determines the fuzziness of clustering. Once all membership values are calculated, new class centers are calculated as

\[
u_i = \frac{\sum_{k=1}^{N} u_{ik}^{m} x_k}{\sum_{k=1}^{N} u_{ik}^{m}} \quad \forall i,
\]

where \( \nu_i \) is the new center of the \( i \)-th class, \( x_k \) is the gray scale value of the \( k \)-th pixel, and \( N \) is the total number of pixels. For the first iteration, the centers of classes can be taken arbitrarily, since they converge to their optimum values. The main advantage of the clustering methods over thresholding is the ability to classify the pixels into more than two classes [24].

### 3.4. Image Fuzzification

As discussed in Section 3.3, higher contrast between objects and background in an image helps in robust segmentation when using a thresholding method. An appropriate
gap between the peaks of the objects and background in an image’s histogram leads to better performance in segmentation.

Sometimes gray scale values of the body of an object are not homogenous. This inhomogeneity causes lower contrast between the objects and background and consequently misclassification, especially in biomedical images. There are many factors in biomedical image acquisition methods that cause inhomogeneity in the image of an object such as a human’s organ. In general, projections of a 3D scene onto a 2D image have some inaccuracies, because all depth details are compressed into a single value. An X-ray image is an embodiment of this drawback, since its key feature is overlaying the body structures at all depth levels onto a 2D image [25]. Inhomogeneous ribs in the X-ray image are shown in Figure 3.8.

![Figure 3.8](image_url)

Figure 3.8. A chest x-ray image. The gray scale values of the ribs are not homogeneously distributed, which makes it difficult for the recognition algorithm to extract the ribs completely (Courtesy: Medline Plus, National institute of Health (NIH)).
A medical expert who deals with medical images learns about the artifacts and inaccuracies in biomedical imaging modalities in order to make accurate diagnostic decisions. The images captured by the cell-phone microscope specifically suffer from poor quality; however, they can be utilized by an expert, or even a non-expert user, to count the total number of cells.

Figure 3.9(b) shows a small portion of an image of blood cells. Cell inhomogeneity is obvious in Figure 3.9(a). This effect causes misclassification in the segmented image in Figure 3.9(b). The difference between the two images is the number of gray scale values used to represent them. More details are distinguishable in Figure 3.9(a), since it is a 256-value gray scale image (gray image), while in the other, only two scale values (binary image) are used. A good compromise between gray and binary images is achieved by image fuzzification.

In this work, image fuzzification is applied using a step-wise fuzzy function to provide more than two classes for the segmented image. Image fuzzification benefits from a higher number of gray scale values than binary images have.

In fuzzy sets, each member has a membership degree. Here, a gray image \( P \) is considered to be a set of pixels; so, each pixel \( p \) is a member of this set. A fuzzy subset \( A \) is defined as a set of membership values assigned to the pixels of image \( P \),

\[
A = \{ \mu_A(p) | p \in P \},
\]
where $\mu_A(p)$ is the membership value assigned to pixel $p$. Fuzzy function $\mu_A$ maps every pixel in the image to a value in the range from 0 to 1 [26],

$$\mu_A: P \to [0, 1].$$

In this work, the fuzzy membership that each pixel can take is directly associated with its probability of being classified as belonging to an object.

![Figure 3.9](image.png)

Figure 3.9. The result of segmentation using a threshold to illustrate the effect of inhomogeneity of the gray scale values of the objects. Parts of the cells in (a) are classified as background in (b), and also parts of the background are classified as cells.

The most crucial part of the image fuzzification is assigning the appropriate membership values to the pixels. In the blood cell images, a darker pixel is more likely to be classified as a part of a cell. Therefore, a straightforward strategy for designing the fuzzy function is to subtract the gray scale value from 255 and then to divide the result by 255, giving a value in the range of 0 to 1. This function assigns the highest membership value (1) to the darkest pixel (0) and the lowest value (0) to the brightest one (255), as
depicted in Figure 3.10. In this work, a fuzzy function is the relation between the pixel’s gray scale value and its membership value.

![Graph of fuzzy function](image)

**Figure 3.10.** A linear fuzzy function that is extended from the minimum to the maximum gray scale values. As the gray scale value increases, the membership value decreases, since it is less probable to be in the gray-scale range of the blood cells.

The main drawback of the fuzzy function plotted in Figure 3.10 is that the information provided by the histogram is not taken into account. Notice that the histogram is not stretched over all gray scale values; only a limited range of gray scale values appear in images. Assume $x_d$ and $x_b$ are the gray scale values of the darkest and brightest pixels, respectively. Also assume that pixels with gray level $x_d$ belong to a cell
(membership value equal to 1) and pixels with gray level $x_b$ belong to background
(membership value equal to 0). In other words, let $x_d$ and $x_b$ define the boundaries of the
fuzzy region. So, the fuzzy function becomes

$$
\mu_A = \begin{cases} 
1 & x \leq x_d \\
\frac{x - x_d}{x_b - x_d} & x_d < x < x_b \\
0 & x \geq x_b 
\end{cases}
$$

where $x$ is the gray scale value of a pixel, and $\mu_A$ is the membership degree. Figure 3.11
shows such a function.

The range from $x_d$ to $x_b$ is the fuzzy region, and the membership value associated
with this region is linearly decreasing. It is not a necessity for the fuzzy function to be
linear in the fuzzy range. The fuzzy function can assign any value from 0 to 1; however,
it should be monotonically non-increasing as gray level increases.

Besides the linearity or non-linearity of the fuzzy function in the fuzzy region,
determining the boundaries of the fuzzy region is also crucial. The lower bound of the
fuzzy region is the gray scale value below which any darker pixel certainly belongs to a
cell. The upper bound is the gray level above which any lighter pixel is background. In
Figure 3.12, the main peaks of objects and background in the histogram are shown. A
fuzzy region is defined where these two main peaks overlap. There is a chance of
misclassification for the pixels with a gray level in this region. Values $x_d$ and $x_b$ are
assigned as the boundaries of this overlapped region.
Figure 3.1. A fuzzy function that is linear in a limited range, called the fuzzy range. Gray scale values smaller and greater than the fuzzy range have the membership values 1 and 0, respectively. The upper, $x_b$, and lower, $x_d$, boundaries of the fuzzy region are indicated.

In this work, a strategy similar to the one used for Gaussian distributions is utilized to determine the fuzzy region. The optimum threshold is within the fuzzy region. In this thesis, it is proposed to assign a portion of the objects and background classes to the fuzzy region. Suppose $l_1$ is the range of gray levels to be classified as objects,

$$l_1 = T - x_1,$$  \hspace{1cm} (43)
Figure 3.12. An illustration of the fuzzy region in the histogram. The fuzzy region is where the objects and background distributions overlap. The probability of misclassification is highest in this region.

where $T$ is the threshold, and $x_1$ is the least gray level in the image. Similarly, $l_2$ is the range of gray levels for the background class,

$$l_2 = x_2 - T.$$  \hspace{1cm} (44)

where $x_2$ is the greatest gray level in the image.
A fuzzy ratio \( m \) is defined to assign a portion of the gray-scale range of the objects and background to the fuzzy region, where

\[
0 < m < 1.
\]  

(45)

Therefore, by assigning a fuzzy ratio, the boundaries of the fuzzy region are known as

\[
g_d = T - m \times l_1,
\]

(46)

and

\[
g_b = T + m \times l_2.
\]

(47)

The fuzzy membership allocation to the fuzzy region is not necessarily linear. For simplicity and to reduce the complexity of the algorithm, a step-wise function is applied to allocate fuzzy membership in the fuzzy region. The membership value allocated for the objects in the fuzzy region \( x_d < x < T \) is two-thirds, and the membership value allocated for the background in the fuzzy region \( T < x < x_b \) is one-third. Remember that the fuzzy function maps a gray level to a fuzzy membership value to show its likelihood of belonging to an object. Such a function is plotted in Figure 3.13.
Figure 3.13. A step-wise fuzzy function. The fuzzy membership in the fuzzy region is 2/3 for gray scale values less than the threshold and 1/3 for those more than the threshold.

3.5. Object Recognition and Counting

The previous sections provide the theoretical concepts for preparing the image for cell recognition and counting. Once the image is segmented optimally, every object must be extracted and its features evaluated to determine if it is a desired object. This process is object recognition and has many applications from biomedical image analysis to automatic military target finding.
Suppose one of the segmented objects has only 10 pixels, while the desired object must consist of approximately 30 pixels. So, a reliable counting algorithm must disqualify the small object for counting. A robust counting algorithm is supposed to find the best candidates from the extracted objects. Many approaches for object recognition are proposed, which are categorized into one of two groups:

- Contour-based methods
- Region-based methods

In the contour-based approach, only the boundaries of the shape are used for recognition, while in the region-based approach, the features that specify an object are considered as the distinct factors [27]. Many techniques are proposed in the literature for each approach. In this work, the objective is counting the number of cells as accurately as possible. Since most of the cells are rounded and have approximately the same size, some criteria, such as cell area, can be considered to enhance the robustness of the algorithm.

It is hoped that extracted objects in a segmented image are individual objects that are ready to be counted. However, as mentioned earlier in this section, some extracted regions may not be appropriate candidates and should be disregarded. In this work, due to the poor image quality, an object may be segmented as two separate objects. In such cases, it is desired to merge the separated regions of the object to count them as a single object. Figure 3.14(a) shows such broken and separated objects.

Overlapped objects, on the other hand, cause two or more objects to merge together and be considered as a single object. In this work, it is seen that some blood cells
partially overlap each other, and consequently are counted as one. Such problems highly affect the ultimate result of the counting algorithm. Figure 3.14(b) depicts a segmented image that has overlapped and connected blood cells. Following is the explanation of two methods proposed for cell counting.

Figure 3.14. An illustration of (a) broken and (b) connected blood cells after the segmentation process

3.5.1. Pixel-Counting Method

A straightforward technique for counting similar objects in a segmented image is to calculate the total number of pixels that are classified as objects. If the objects to be counted are similar in terms of shape and size, then the average area of each object is known. By dividing the number of pixels classified as objects by the known average area of one object, the total number of objects is achieved.

Figure 3.15 shows a segmented image containing 19 similar objects, distributed randomly. This size of this image is reduced in Figure3.15. The objects are similarly
rounded, each with an area approximately equal to 360 pixels at original size of the image. The total number of black pixels in the image is 6525 pixels. So,

\[
\text{Number of objects} = \frac{6525}{360} = 18.125
\]  

(48)

Figure 3.15. A segmented image that contains 19 similar rounded objects. Some of the objects overlap others.

is the approximate number of objects depicted in the image. The total number of objects can only be a positive integer. The calculated number can be rounded to either 18 or 19.

This simple method provides an accurate estimation of the total number of objects in the image, only if the following criteria are met:

- The objects have the same area
- No objects overlap.

In this work, this method is not considered as a reliable method to count the blood cells for three reasons. First, the red blood cells are not necessarily the same size and tend
to have a donut shape. Such shape differences affect the total number of pixels that are classified as cells. The second factor is the poor image quality, which causes segmenting bigger cells in the center of the image and smaller ones in peripheral regions.

Overlapping is the third reason to avoid using this method. These three factors reduce the reliability of this method, and therefore a more complicated algorithm is required in this work.

3.5.2. Masking Method

To overcome the flaws of the pixel-counting method, the masking method is proposed. By applying a mask similar to the shape of the objects, and sweeping it through the image, the objects can be detected and then counted. This algorithm returns the coordinates of the objects in the image. Figure 3.16 depicts a simple scheme of how this method works.

A mask is depicted in Figure 3.16(a). A binary (segmented) image containing an L-shaped object is depicted in Figure 3.16(b). Every square represents a pixel that can take only one of two values, 1 or 0, which are shown as black and white, respectively. In the masking method, the L-shaped mask lies over the binary image in all possible positions and orientations to examine if it matches objects in the image’s content.

In this method, the size and shape of the objects still must be known. However, the problem with overlapping is reduced. Because the computer program checks every
possible L-shaped configuration of the pixels in the image, it can detect the overlapping objects separately.

Figure 3.16. Illustration of the masking method. A prototype of the object to be counted is constructed as a mask (a) and is swept over the image that contains such an object (b) to count the objects.

In this work, not all of the objects have the same size and shape. Therefore, a strategy is needed to deal with the variation of the size and shape of the red blood cells. So, it is proposed to perform the masking method for a set of shapes and sizes that a red blood cell can have. Then a method to filter out the false-detected cells is provided to avoid over-counting.

Suppose an average shape for the red blood cells is an $r$-radius filled circle. A set of such circles with smaller and larger radii is taken as a mask set. For example, two circular masks with radii $(r - 1)$ and $(r + 1)$ pixels are two alternatives for the $r$-radius mask. All of the masks in the mask set are applied to identify the cells of different sizes.
and those which suffer from shape aberration. Figure 3.17 shows how this strategy can identify an oval-shaped cell. Suppose the circle shown in Figure 3.17(a) is a circular mask with the average size, and the circle in Figure 3.17(b) is a smaller size mask. Figure 3.17 shows how the smaller size mask can be fitted inside an oval-shaped cell. Therefore, applying a set of masks of different sizes instead of only an average-sized mask can detect the cells with changes in size and shape.

![Figure 3.17](image)

Figure 3.17. Identifying cells that are not circular. (a) The $r$-radius circular mask, which is the average size of the cells. (b) The $(r - 1)$-radius circular mask, which is the small alternative mask of cells. (c) The $(r - 1)$-radius circular mask fitted in an oval-shaped cell.

A problem that arises with the above strategy is over-counting. For instance, a smaller mask can match a bigger cell in several positions. Figure 3.18 shows two small circles embedded in a bigger circle. To avoid over-counting, a minimum Euclidean distance between two detected objects can be set. Though such a guard distance avoids over-counting, it also reduces the capability of the algorithm to detect overlapped cells, which is a valuable feature of the masking method.
Big guard values guarantee counting a single cell only once; small guard values enable the algorithm to count highly-overlapped cells. In practice, the optimum guard distance can be determined during a calibration process of the algorithm. The guard distance (d) shown in Figure 3.19 allows the counting of cells that are overlapped by no more than the amount depicted.

Figure 3.19. A guard distance (d) to provide a minimum distance between the centers of the detected cells. Thus, the cells that are overlapped such that their centers are closer to each other than the guard distance are not counted separately.
ALGORITHM DEVELOPMENT

In this chapter, the whole procedure, from preparing the raw image that is captured by the cell phone microscope developed at CBST to counting the red blood cells in the image as correctly as possible, is reviewed. The cell counting procedure consists of methods proposed in the literature, modified for better performance for this particular cell-phone microscope. This procedure is divided into three sections: 1. Preprocessing, 2. Image segmentation and fuzzification, and 3. Cell detection and counting.

The untouched image captured by the CBST cell phone microscope is a true-color (24-bit RGB) image. It is first reduced to a 256-level gray image, since the color information is subject to change with the amount of stain used in preparing the blood sample and the illumination type. In the gray image, cells are observed as dark filled circles or donuts in a bright background.

4.1. Pre-processing

4.1.1. Radial Distortion Correction

As mentioned in the second chapter, utilizing a spherical lens to attain more magnification causes pincushion distortion. This artifact must be corrected prior to
further processing. Using a PM or DM method, the image can be corrected by applying the appropriate distortion constant and center. As explained in Section 3.1, a distorted image of straight lines is needed to provide the correction parameters. For this purpose, a pattern of parallel dark and bright straight bars is utilized. Figure 4.1 shows an image of this pattern captured by the cell-phone microscope.

![Figure 4.1](image)

Figure 4.1. An image of the pattern of parallel dark and bright straight bars captured by the cell-phone microscope developed at CBST. The metric measurements of this image, compared with those for the original pattern, help in calculating the distortion characteristics, such as the distortion center and constant, of the imaging system.

Chapter 3 reviewed methods for calculating the distortion center and constant. In this work, the distortion constant is estimated through trying a wide range of correction
parameter values and verifying the performance by visual inspection. Once the camera is calibrated, the distortion parameters do not change.

4.1.2. Selecting the Most-Focused Region

In addition to radial distortion, the ball lens causes de-focused regions in the periphery of the image. Since the objects in the fuzzy de-focused regions are poorly distinguishable, those regions need to be disregarded. In Figure 4.2, the de-focused regions are apparent in peripheral regions of the image. Prior to further processing, the algorithm must search for the most-focused regions and crop the others.

The focus criterion functions studied in Section 3.2 provide a quantitative measure to compare the contrast of images captured of a scene. The auto-focusing systems in digital cameras slide the lens back and forth and applying those focus criterion functions. In other words, the criterion function is applied on images at different lens positions to find the maximum. In this work, the image already is captured, and finding the best focused region within the image is desired. For this reason, the focus index of different regions of the image must be calculated.

One approach for searching for the best focused region is to choose, say, a $256 \times 256$ (pixels) sized-window and to slide it over the image to see where it returns the maximum focus index. Though this approach is straightforward, it is time-consuming. The original size of the image is $1200 \times 1600$ pixels, and for a $256 \times 256$ pixel window, sliding in 10 pixel steps over the image results in 12565 regions of the image to be
processed. Thus, an intelligent search strategy is required to save time. Because the spherical ball lens causes rotationally invariant distortion, the most focused region must be near the center of the image. Therefore, in this work, the search is started at the center of the image.

Instead of a global search algorithm (e.g., a sliding window), a surface-based backtracking algorithm is applied, starting at the center of the image [28]. This algorithm is to find the most-focused region by assuming that there are no local maxima. That is, it is expected that the criterion function returns the maximum value at the most-focused region, and the value returned monotonically decreases when the algorithm moves from that point.

The algorithm consists of two stages of backtracking search. The difference between the stages is the step size in moving the sliding window. Once the window reaches a region that returns the maximum value of a criterion function, a finer step size search tries to tune the position of the window in order to reach the best performance. This strategy saves time compared with global, blind search methods. The details of this algorithm are as follows:

1. Put the window at the center of the image, denoting that position as the “currentState,” and calculate the criterion function for that region.
2. Slide the window “step” pixels upward, denoting the new location as “newState.” If the criterion function returns a value greater than the one at the “currentState,” then “currentState” ← “newState,” and go to (2).
3. Do (2) for downward.

4. Do (2) for the right direction.

5. Do (2) for the left direction.

6. Return the “\text{currentState}” as the location of the window that is the most-focused region.

4.2. Image Segmentation and Fuzzification

Chapter 3 reviewed techniques for image segmentation. In this project, the region-based segmentation methods are preferred, rather than the edge-based ones. The edges in the poorly-illuminated cell-phone microscope images are not sharp enough to be good candidates for the edge-based methods. Region-based segmentation techniques classify the pixels of the image into classes. The algorithm developed in this work is able to segment the cell images using all the methods described in Chapter 3, such as Otsu, Minimum Error (ME), and Fuzzy C-Mean.

As was discussed, it is better to fuzzify the image into more than two levels because of the inhomogeneity of the cells. Fuzzification in this work does not involve fuzzy logic; fuzzification is employed for gray level partitioning. Also, distinct ranges of gray scale values cannot be assigned to classify the pixels into classes, because the image is not uniformly illuminated, and therefore the dynamic range of gray scale values changes from one region in the image to another. Figure 4.3 shows the illumination variation between two different regions of the image. The two selected regions, labeled 1 and 2, have different gray scale distributions.
Figure 4.2. An image captured by the cell-phone microscope developed at CBST from a blood sample. The cells in the marginal regions look fuzzy and are not easy to distinguish. The ball lens employed in this microscope is only able to focus on a limited region of the scene.

Figure 4.4 illustrates how the illumination variations can affect the histogram of the image at different regions. The two histograms in the figure are associated with
regions 1 and 2 in the image in Figure 4.3. The histogram associated with low-contrast region 1 is jammed, compared with the histogram of the region 2. Therefore, a single threshold cannot be applied for classification for these two regions.

Figure 4.3. Effect of uneven illumination. Uneven illumination prevents using a global threshold. Two boxes, labeled 1 and 2, are selected at different regions of this image of the cells. The difference in illumination between these two regions is clearly visible in this image.

Several approaches are available to deal with uneven illumination in digital images. One method tries to simulate the distribution of the illumination in the image using a quadratic polynomial [29]. This method is not employed in this work, since it
assumes that the center of the image has the highest level of illumination, whereas the region of the image with the highest level of illumination may appear away from the center.

Figure 4.4. The effect of illumination variation on the histogram. The histogram of region 2 is jammed and shifted to smaller gray scale values. A global threshold cannot separate the cells from background in the all regions of this image.

Another proposed approach is to separate the illumination and reflection factors using a logarithmic operator [16]. This method assumes that the reflectance factor of every object is a constant, and it is what differentiates the gray level of objects from background in an image. Thus, the gray level of an object in an image is proportional to
both the reflectance factor of that object and the light intensity that impinges on the object. This method divides the gray levels into reflectance and illumination factors, and then tries to remove the illumination factor. The images in this work are rather noisy, since they are poorly-illuminated. The above method separates the reflectance factors using a high-pass filter. In this work, although the light is transmitted through the cells, this method still is applicable. But application of the above method has the undesirable effect of increasing the level of noise in the processed image.

Another approach is employed in this work, which is robust to the uneven illumination of the images and leads to an algorithm of low complexity. This method is known as adaptive thresholding, which means applying different thresholds to sub-images instead of using a single global threshold. In this method, the entire image is partitioned into sub-images, and a threshold is assigned for each sub-image. Different thresholds in the neighboring sub-images cause inconsistencies in the segmented image. The cells that fall at the border of the two sub-images face two different thresholds. To overcome such problems, the thresholds are interpolated to construct a threshold surface of the size of the image. The result is a smooth threshold surface, which assigns a distinct threshold for each pixel of the image.

Suppose that the image is partitioned into sub-images, and for every sub-image, a threshold is given. Then, the threshold of each sub-image is assigned to the center of that region, as depicted in Figure 4.5. The interpolation provides the threshold value for each pixel of the image. Figure 4.6 shows the smooth threshold surface for the cell image in
Figure 4.3. As shown in Figure 4.6, the regions with greater thresholds are associated with the regions of the cell image in Figure 4.3 that are brighter. Similar approaches are taken in this work for upper and lower boundaries of the fuzzy region. Thus, three threshold surfaces are constructed to fuzzify the image based upon the fuzzy function depicted in the Figure 3.13.

<table>
<thead>
<tr>
<th>$T_1$</th>
<th>$T_2$</th>
<th>$T_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T_4$</td>
<td>$T_5$</td>
<td>$T_6$</td>
</tr>
<tr>
<td>$T_7$</td>
<td>$T_8$</td>
<td>$T_9$</td>
</tr>
</tbody>
</table>

Figure 4.5. Partitioning an image into nine sub-images and assigning a distinct threshold to each of them. The centers of the sub-images are where the thresholds are assigned for interpolation.

The number of sub-images must be selected to make sure that there are enough cells in every sub-image. The illumination variation, on the other hand, must be negligible in every sub-image. Moreover, for a robust algorithm, unusual scenarios also must be considered. Suppose there is not any cell in a sub-image. Then, the threshold assigned for that sub-image becomes misleading. In this algorithm, a function is employed to compare the contrast of every sub-image with those of the others. This function checks if the contrast from a sub-image to its neighbor changes abruptly. If the contrast of a sub-image is far below the average of the contrast of its neighbors, then the
average of the neighboring thresholds is utilized as the threshold in that sub-image.

Applying this function improves the performance of the algorithm.

Figure 4.6. The threshold surface for the image shown in Figure 4.3. The image shown in Figure 4.3 is partitioned into 25 sub-images, and using the threshold of each sub-image, the threshold surface is constructed.

4.3. Cell Counting

Once the image is segmented and fuzzified, it is time to do the most significant part of the algorithm, which is counting. All of the processes reviewed so far prepare the raw image for the cell recognition part required for cell counting. Among the recognition
methods reviewed in Chapter 3, the masking method is applied in this work. Here, a fuzzy approach is proposed to modify the masking method for robust cell recognition in the poorly-illuminated images captured by the cell-phone microscope. The reasons for choosing the masking method are as follows:

1. Masks can be created based upon the shape of the object to be counted.
2. Object of different sizes can be recognized in this method by applying different sizes of masks.
3. By applying this method, overlapped cells also can be recognized separately.

It is observed in Figure 4.7 that some of the red blood cells are more donut-shaped, while others are more like solid filled circles. A donut-shaped mask can be fitted to both shapes of red blood cells. The mask developed in this work is a one-pixel-thick circle, which is employed to detect the red blood cells. As mentioned in Section 3.5, the mask is created in a range of different sizes in order to detect objects of different sizes. In this work, the algorithm can be tuned to desired mask sizes. This size range is an input option of the computer program that performs the blood cell counting.

Red blood cells for a healthy human range in size from about 4 to 6 μm [30]. The distance between the lens and the imaging sensor array in the cell phone is constant. Therefore, the blood sample must be placed at a fixed distance from the cell phone’s lens to be captured as a focused image. This fixed distance in the optical system provides a fixed magnification for the cell-phone microscope. Therefore, the final range of the blood cell sizes in the image should be predictable, and the predicted range of sizes can be applied to tune the algorithm.
In the masking method, the mask is swept over the segmented image. Whenever the mask covers an object, that object is counted. In such a case, a segmented image must be a binary image, with its pixels having only the logic values zero or one, represented by white and black pixels in images. But, in this work, as described in Section 3.4, due to inhomogeneity of the gray levels in small regions of the image, the pixels are fuzzified to more than two levels. A fuzzy function, depicted in Figure 3.13, is applied to map the image into a fuzzy scene with four fixed membership values. Therefore, when the mask sweeps over the fuzzy scene, each pixel of the mask returns a fuzzy membership value, instead of a simple zero or one.

As described in Section 3.4, some biomedical images suffer from inhomogeneity and need post-processing. For instance, in Magnetic Resonance Angiography, the blood vessels are not homogeneous in gray levels, due to magnetic field inhomogeneity, motion, or the other artifacts. Post-processing is needed to connect the blood vessels in the image [31]. In [32], the concept of “Fuzzy Connectedness” is presented as a framework for finding the level of connectedness between the regions in an inhomogeneous image. In a fuzzy scene, a link is a series of adjacent pixels. The criterion
functions that are proposed in [32] measure the connectivity of a link in a fuzzy scene. If this connectivity exceeds a threshold, the regions that are connected through that link are considered as parts of a single object. In Figure 4.8, a link in a fuzzy scene is depicted as a dashed line between two points, A and B. If the connectedness index of this link derived from the fuzzy connectedness criterion functions exceeds the threshold, points A and B are in the same object.

Suppose that \( c \) is a link that consists of a sequence of adjacent pixels in a fuzzy scene, such as

\[
c: < c_1, c_2, c_3, ..., c_n >,
\]

(49)

where, \( n \) is the number of the pixels in the link. Each of these pixels is associated with a membership value in the fuzzy scene. In this work, the one-pixel-width circular mask, which is utilized to detect the red blood cells, is assumed as a link in a fuzzy scene. If the fuzzy connectivity of such a link is above a specified threshold, then the link is considered to be on a red blood cell in the image. Figure 4.9 shows how a ring mask is placed on a fuzzy scene. The membership values of the pixels in the fuzzy scene that the mask is placed on form a sequence of membership values, such as

\[
l = (m_2, m_3, m_4, m_{12}, m_{20}, m_{27}, m_{34}, m_{40}, m_{46}, m_{45}, m_{44}, m_{36}, m_{28}, m_{21}, m_{14}, m_8),
\]

(50)

where, \( l \) is the membership sequence of the link, and the \( m_i \)'s are the membership values of the pixels in the link. In this work, two fuzzy connectedness criterion functions are chosen to evaluate the connectivity of the fuzzy links.
4.3.1. Affinity Function

In Section 3.4, it is mentioned that the membership value of a pixel in the fuzzy scene is proportional to the probability of belonging to a red blood cell. Therefore, a straightforward criterion function can be the sum of the membership values in the link; the higher the sum, the more probable that the link defines a cell.

The affinity function returns a fuzzy degree in the range of 0 to 1 that is proportional to the probability of detection of a cell. The ring mask is highly probable to be on a cell when its pixels return 1 as the membership value. In this work, the affinity
function is the average of the membership values of the link in the mask. Thus, the
affinity function \( \mu_a \) of a link, \( l \), is

\[
\mu_a(l) = \frac{1}{n} \sum_{i=1}^{n} m_i \quad \text{for } i = 1, \ldots, n.
\]  \hspace{1cm} (51)

If all membership degrees are equal to 1, the function returns 1, as well. This value is the
maximum that the function can return.

---

Figure 4.9. A one-pixel-thick ring mask on a fuzzy scene. The gray squares represent
pixels of the mask, and the white ones are other pixels. When the mask is placed on a
fuzzy scene, each of its pixels returns a fuzzy membership value.
4.3.2. Homogeneity Function

Besides the affinity function, which is a straightforward approach for cell detection, other factors also need to be considered. Notice that a link can be detected as a cell when the sum of its membership values is greater than a specified threshold. However, a link with a high affinity index may be placed on multiple cells, instead of one. In Figure 4.10, such a scenario is depicted, as the ring mask, dashed circle, is placed on three cells. Most of the pixels of the mask return high membership values, since they belong to cells. Thus, an interceptor mechanism is required to avoid detecting cells in such scenarios. The main difference in the fuzzy membership sequence of a mask when placed on multiple cells versus a single cell is the abrupt change in the membership values where the link jumps from a cell to the background or vice versa.

The homogeneity function [33] is defined to return the maximum value when there is no variation in the sequence of the membership values and the minimum when it has the highest possible variations. The discontinuity of transition from one cell to another is the key point that is used for the designation of the homogeneity function. The sequence of the membership values of a link is analogous to a discrete time signal. The signal is a constant ("dc") when all values are equal, and it is has the highest frequency when the values alternate from high to low. In this application, since the membership values are limited to zero, one-third, two-third, and one, the signal has the highest frequency when it is

\[ (0, 1, 0, 1, 0, 1, \ldots). \]
Figure 4.10. A situation in which the mask spans across cells. In such a case, most of the mask’s pixels are on the cells, and they return high membership values.

Thus, the homogeneity function should return one for a dc signal and zero for the signal represented in (52). The homogeneity function is defined as

\[ \mu_h(l) = 1 - \frac{1}{2n} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha(c_i, c_j) |m_i - m_j|, \quad \text{for } i, j = 1, ..., n \text{ and } i \neq j, \]  

(53)
where $\alpha(c_i, c_j)$ is the adjacency function defined as

$$\alpha(c_i, c_j) = \begin{cases} 1, & |i - j| = 1 \text{ or } n - 1 \\ 0, & \text{Otherwise.} \end{cases}$$

The 2 in the factor $\frac{1}{2n}$ in (56) is because the values in the summation are calculated twice, when $i$ and $j$ are substituted for each other.

### 4.3.3. Cell Recognition

When the mask is placed at a location on the fuzzy scene, the affinity and homogeneity fuzzy functions provide two indices for that position. The outputs of the fuzzy functions are in the range of zero to one, where one is associated with the highest probability of detecting a cell. Also, as mentioned in Section 3.3, due to the inhomogeneity in the gray levels of the cells, it is anticipated that, for some of the cells, the fuzzy functions return values less than one. But, these values should be sufficiently close to one to consider those coordinates of the image as the locations of cells.

Therefore, two thresholds for the two fuzzy functions are set for detecting the cells in a fuzzy scene. Suppose that $\eta_a$ and $\eta_h$ are the threshold values for the output of the affinity and homogeneity functions, respectively. When the link of the mask $(l)$ at location $i$ of the image returns $\mu_a(l_i)$ and $\mu_h(l_i)$ as the outputs of the fuzzy functions, the conditions that must be met to detect a cell are

$$\mu_a(l_i) > \eta_a \text{ and } \mu_h(l_i) > \eta_h.$$ 

(55)
In pattern recognition, the functions that evaluate a pattern form a feature space. The output values of the fuzzy functions are the features in this work. Figure 4.11 shows the mask placed at coordinates \((x_0, y_0)\) of the fuzzy scene. The sequence of the membership values of the mask is the input to the fuzzy functions. The output values of the fuzzy functions are the coordinates of a point in the feature space. The feature space is shown in Figure 4.12, where the horizontal and vertical axes are the outputs of the affinity and homogeneity functions, respectively.

![Figure 4.11. The mask sweeping over the fuzzy scene. The mask returns two fuzzy function values corresponding to each pair of coordinates of the image](image)

The affinity and homogeneity thresholds form a region in the feature space, which is the cell detection area, depicted as the gray region in Figure 4.12. If the fuzzy function values associated with the location of the mask in the image form a point in the cell detection area, then the location of the mask is considered as the location of a cell.
Although, the feature space is shown as Cartesian, \( \mu_a \) and \( \mu_h \) are not necessarily statistically independent. Therefore, assigning thresholds to form a rectangular detection area may not necessarily result in the best cell detection. The cell detection area may have other shapes, rather than rectangular. However, in this work, other possible cell detection areas are not considered. Setting the appropriate values for the thresholds in the feature space is a crucial task, which is done manually in this project. The feature space thresholds are adjusted for an image to detect as many cells as possible, with least error, and then the adjusted algorithm is applied on other images to measure the performance of the algorithm.

![Cell Detection Area](image)

**Figure 4.12.** The feature space for cell recognition formed by the fuzzy functions, affinity and homogeneity. The affinity and homogeneity thresholds form a region within the feature space for cell detection.
The flowchart of the algorithm developed in this work is depicted in Figures 4.13-14. It includes the whole procedure of image processing and pattern recognition that is reviewed in this chapter. The computer program, implemented in MATLAB, of this algorithm is presented in Appendix 1.
Figure 4.13. A flowchart of the algorithm that is developed in this thesis (part 1)
Figure 4.14. A flowchart of the algorithm that is developed in this thesis (part 2)
Chapter 5

RESULTS AND DISCUSSION

In this chapter, the results of each module of the counting algorithm are presented and discussed. As reviewed in Chapter 4, the counting algorithm consists of the three modules, preprocessing, segmentation and fuzzification, and cell recognition. At the end, the performance of the counting algorithm is evaluated.

5.1. Preprocessing

The distorted image of the bright and dark bars, shown in Figure 4.1, is utilized for finding the distortion parameters. The center of the image is assumed to be the distortion center. The distortion constant then is achieved by trying different values for both the PM and DM models. The performance of the distortion correction is visually evaluated. Figure 5.1 shows the distortion corrected version of the image shown in Figure 4.1 using the PM model. Once the distortion model of a cell-phone microscope is known, it can be applied on all of the images captured by it.

Selecting the most focused region of the image is the other part of the image preprocessing in this work. The algorithm developed in this work is capable of applying each of the four criterion functions presented in Chapter 3. For one input image, each of
the four methods results in a different output image. Figure 5.2 shows the output images for these four criterion functions.

Figure 5.1. The distortion-corrected image of the bright and dark bars depicted in Figure 4.1. The PM model is used for this correction, where the distortion center is the center of the image, and $\lambda$ (distortion constant) = 0.001.

In addition to performance, the speed of an algorithm is important if it is to be implemented on cell phones. Table 5.1 shows the elapsed time of each of the four criterion functions. To obtain these results, the methods were employed in the search algorithm described in 4.1.2 written in MATLAB and run on a system with the following
characteristics: Intel Core 2 Duo Processor, 2-GB memory, Microsoft Windows 7 64-bit OS, and MATLAB™ 7.9 (R2009b). The table shows that the TEN method was fastest.

Figure 5.2. The output images of the focusing criterion functions using different methods. (a) Gray Variance (GV). (b) Sum Modulus Difference (SMD). (c) Sum Modified Laplacian (SML). (d) Tenengrad (TEN).
Table 5.1. The comparison of the elapsed time for different focusing criterion functions

<table>
<thead>
<tr>
<th>Criterion Function</th>
<th>Elapsed Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gray Variance (GV)</td>
<td>0.500</td>
</tr>
<tr>
<td>Sum Modulus Difference (SMD)</td>
<td>1.650</td>
</tr>
<tr>
<td>Sum Modified Laplacian (SML)</td>
<td>26.500</td>
</tr>
<tr>
<td>Tenengrad (TEN)</td>
<td>0.274</td>
</tr>
</tbody>
</table>

5.2. Image Segmentation and Fuzzification

Three methods of image segmentation, Otsu, ME, and Fuzzy C-Mean, are reviewed in Chapter 3. The adaptive thresholding technique partitions the image into several sub-images and segments each of the sub-images separately. The thresholds are interpolated to produce a smooth threshold surface. Then, the image is fuzzified by applying a fuzzy ratio \((m)\). Figure 5.3 shows a fuzzified blood cell image. The segmentation method is the Otsu method; the number of sub-images is 25, and the fuzzy ratio is 0.15.

As shown in Figure 5.3, the pixels that belong to cells have the highest membership values and are the brightest pixels. On the other hand, the background is mostly black, which is associated with the lowest membership value. Recall that the membership value of a pixel is associated with the probability of being classified as part of a cell.

Two quantitative variables are employed in this part of the algorithm. One is the number of sub-images used in the adaptive thresholding, and the other is the fuzzy ratio \((m)\) used for fuzzification. As discussed in 4.3, because of the fixed magnification and limited range of the blood cells sizes, the size of cells in the image does not vary
much. The image is partitioned into five equal segments horizontally and vertically resulting in 25 sub-images. Thus, every sub-image has at least one or two cells, and the illumination variation within each sub-image is negligible.

Figure 5.3. The fuzzified image of the blood cell image shown in Figure 5.2(d). The side bar shows the gray scale associated with the fuzzy membership values. This image is segmented using adaptive thresholding, where the thresholds are chosen using the Otsu method. The image is partitioned into 25 sub-images to find the thresholds for each sub-image separately. The fuzzy ratio that is used to fuzzify the image is 0.15.

The fuzzy ratio \( m \) defines the fuzzy region of the gray scale, as described in Section 3.4. There is not a particular criterion that defines the optimum value of this variable. In this work, the fuzzy ratio is tuned based on the counting performance. It was found that a ratio of between 0.1 to 0.2 resulted in the best performance.
5.3. Cell Recognition

Section 4.3 described the details of the masking method utilized in this work. In this part of the algorithm, several variables must be tuned. First, the range of the mask size must be defined. Then, the guard distance between two detected cells and the affinity and homogeneity thresholds must be tuned. As discussed in Section 4.3, for a particular cell-phone microscope, a limited range for the mask size can be defined. In this work, the minimum and maximum radii for the mask were selected to be six and nine pixels, respectively. By applying masks in this size range, all of the red blood cells can be detected, if they are well segmented, and roundly shaped.

As described in Section 3.5, the guard distance avoids over-counting by limiting the distance between two detected cells. On the other hand, it can prevent the detection of overlapped cells. As a compromise, the diameter of the smallest mask, or 12 pixels, is taken as the guard distance. In Figure 5.4, the two inner circles represent small masks, while the two other circles represent big masks. The guard distance \( d \) is the diameter of the inner circles. Thus, some of the partially overlapped big cells can be detected. The guard distance is an input variable of the algorithm and can be tuned during calibration.

But, the most important part of the cell recognition process is finding the appropriate values for the affinity and homogeneity thresholds. As shown in Figure 4.12, these two thresholds form a cell recognition region in the feature space. Unfortunately, there is not a mathematical way to achieve the optimum values for these thresholds. However, some machine learning techniques, such as neural networks, can be used to
find values for these thresholds to minimize error in cell detection. In this work, the thresholds were changed while the performance was observed. The values of the affinity and homogeneity thresholds that gave the best performance were employed in the algorithm.

![Figure 5.4. A guard distance as long as the diameter of the small masks. This guard distance allows detecting some big partially overlapped cells.](image)

5.4. Performance

In preparation for presenting the performance of the algorithm developed in this work, a brief review of performance measurements in pattern recognition is presented. The aim in pattern recognition is to automatically find particular patterns or cases in a population of different patterns or cases. The outcomes of such a process are true or false predictions. In this work, the aim is finding the red blood cells. Table 5.2 shows the scenarios that can happen in finding the cells in the image. This table is called the confusion matrix.
Table 5.2. The confusion matrix for the cell recognition algorithm

<table>
<thead>
<tr>
<th></th>
<th>A cell is detected</th>
<th>No cell is detected</th>
</tr>
</thead>
<tbody>
<tr>
<td>A cell exists</td>
<td>a. The number of truly detected cells by the algorithm</td>
<td>b. The number of true cells that are not detected by the algorithm</td>
</tr>
<tr>
<td>No cell exists</td>
<td>c. The number of falsely detected cells by the algorithm</td>
<td>d. The number of mask coordinates on the image where no cell exists, and the algorithm also do not consider them as cells</td>
</tr>
</tbody>
</table>

To evaluate the performance of such predictions, some measures are defined. The parameters used in such measures are the components of the confusion matrix, which are indicated by a, b, c, and d in Table 5.2. The first measure is accuracy, defined as

$$Accuracy = \frac{a + d}{a + b + c + d}.$$  (56)

The accuracy is the proportion of the number of cases that are correctly predicted to the total number of cases that are examined. In this work, the component d in the confusion matrix is the number of coordinates in the image at which there are not cells and the masks do not detect cells. In practice, d outnumbers the other components of the confusion matrix, because there are only about 110 to 160 cells in a 256 × 256 pixel image. Therefore, whatever a, b, and c are, the accuracy is always more than 99%, so accuracy is not a useful measure of the performance of the algorithm.

In this work, two other measures are used to evaluate the performance of the counting algorithm, and they are
Recall is the proportion of the number of the correctly detected cells to the total actual number of cells. Precision is the proportion of the number of correctly detected cells to the sum of correctly and falsely detected cells [34].

Section 5.3 discussed that the optimum values for affinity and homogeneity thresholds were found by changing them and watching the performance. The recall and precision measurements cannot be employed in this matter, since the computer cannot determine automatically which cells are correctly or falsely detected or not detected at all. The truly and falsely detected cells are identified by visual inspection, once the algorithm renders the blood cell image with a mark on the identified cells. This process of manually counting the truly and falsely detected cells must be accomplished for many algorithm-rendered cell images for different pairs of the affinity and homogeneity thresholds. This process takes much time and is not practical during this work. An alternative performance measure is utilized to find the optimum affinity and homogeneity thresholds in this work.

A blood cell image containing a known number of cells is used to find the optimum thresholds for the cell recognition part of the algorithm. The number of cells in the image is manually counted previously. In this new performance measure, the counting
error is the proportion of the difference between the number of algorithm-counted cells, regardless of true or false results, to the true number of cells.

\[ \text{Counting Error} = \frac{(a + c) - (a + b)}{a + b}. \] (59)

In (59), the algorithm is providing the \((a + c)\) value, and \((a + b)\) is the true number of cells that is counted manually. Recall that this counting error is employed only for finding the optimum pair of the affinity and homogeneity thresholds, and the performance of the algorithm is evaluated using precision and recall measures, as introduced earlier.

In Figure 5.5, the affinity and homogeneity thresholds with 0% and ±5% counting error are traced. Point A, for instance, in Figure 5.5 has a large error margin. In this work, the affinity and homogeneity threshold values at point A are chosen.

Once all of the parameters of the algorithm are set, several images of the blood cells are used to evaluate its performance. These images are captured using the cell-phone microscope developed at CBST from a healthy subject’s blood samples, and the actual numbers of cells in each of them is counted manually. Table 5.3 provides the a, b, and c confusion matrix values for this algorithm for seven images, shown in Appendix 2.
Figure 5.5. Finding the optimum affinity and homogeneity thresholds for the cell detection region in the feature space. Point A is selected because it has large margins with the error lines ($0.7 < \eta_a < 0.85; 0.86 < \eta_h < 0.93$).

Table 5.3. Evaluation of the algorithm developed in this work

<table>
<thead>
<tr>
<th></th>
<th>Correctly-detected cells (a)</th>
<th>Cells not detected (b)</th>
<th>Falsely-detected cells (c)</th>
<th>Recall (%)</th>
<th>Precision (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Image #1</td>
<td>106</td>
<td>26</td>
<td>5</td>
<td>80.3</td>
<td>95.5</td>
</tr>
<tr>
<td>Image #2</td>
<td>128</td>
<td>10</td>
<td>5</td>
<td>92.8</td>
<td>96.2</td>
</tr>
<tr>
<td>Image #3</td>
<td>126</td>
<td>3</td>
<td>5</td>
<td>97.7</td>
<td>96.2</td>
</tr>
<tr>
<td>Image #4</td>
<td>133</td>
<td>2</td>
<td>5</td>
<td>98.6</td>
<td>96.4</td>
</tr>
<tr>
<td>Image #5</td>
<td>135</td>
<td>0</td>
<td>6</td>
<td>100</td>
<td>95.7</td>
</tr>
<tr>
<td>Image #6</td>
<td>126</td>
<td>0</td>
<td>7</td>
<td>100</td>
<td>94.7</td>
</tr>
<tr>
<td>Image #7</td>
<td>139</td>
<td>2</td>
<td>8</td>
<td>98.6</td>
<td>94.6</td>
</tr>
<tr>
<td>Average</td>
<td>128</td>
<td>6.14</td>
<td>5.86</td>
<td>95.4</td>
<td>95.6</td>
</tr>
</tbody>
</table>
The data in Table 5.3 show average recall and precision values of 95.4, and 95.5 percent, respectively. Thus, 95 percent of the cells were detected by the algorithm, while 95 percent of the detected cells were actually cells. Thus the error of this counting algorithm is about 5 percent.

Manual cell counting is performed using microscopes and hemocytometers. In manual counting, the precision is highly dependent on the operator. Among the vision-based automatic cell counting algorithms [35], CellC is chosen to compare with the performance achieved in this work. The CellC algorithm does not indicate the location of each cell. Thus, recall and precision cannot be derived for the results provided by it. Therefore, the counting error (59) is used as the performance measure for comparison. The average counting error for the seven blood images in Appendix 2 counted by CellC is almost 26 percent, which is far worse than for the algorithm developed in this work. Compared with CellC, the algorithm developed in this work has the advantage of compatibility with poor illumination in images and also indicating the location of detected cells for calibration purposes.
6.1. Summary

This thesis provides a procedure for preparing and analyzing the raw image captured by the cell-phone microscope developed at CBST and proposes and evaluates a fuzzy method for cell recognition. Each hurdle encountered in cell counting is discussed, and one or several methods are proposed to overcome each hurdle. The primary objective of this work is providing an approximate count of red blood cells. The algorithm developed in this work demonstrated recall and precision values of approximately 95 percent. The algorithm is implemented in MATLAB, and the codes are available in Appendix 2.

6.2. Conclusions

The fuzzy approach that is proposed in this work performs better than a commonly used method for vision-based automatic cell counting. Fuzzification is employed to overcome inhomogeneity in the images, and fuzzy logic is not involved in this work. The main advantage of the algorithm proposed in this thesis is its robustness to poor illumination that causes gray scale inhomogeneity. Another advantage of the algorithm developed in this work is that it indicates the recognized cells in the image. This capability helps investigators find the missed cells and correct the final count.
6.3. Recommendations

This thesis is dedicated to accomplishing the whole procedure of image processing and pattern recognition to provide a count of red blood cells. As this algorithm consists of different modules to perform different operations, the amount of time allocated to develop each module was limited. Each part of the image processing and pattern recognition process can be investigated further to produce better final performance.

In this work, distortion correction is done by choosing the distortion constant manually. Finding this constant automatically is desirable. To select the most-focused region of the image, different criterion functions result in different regions. The performance of these criterion functions needs to be further evaluated. The fuzzification function that maps the gray-scale image to a fuzzy scene is also another crucial part of the algorithm. In this work, a step-wise function is utilized for fuzzification that can be replaced by many other functions. Studying the effect of the fuzzification function on the final performance of the algorithm is needed in order to find and apply the optimum function.

Cell recognition needs much consideration, since it directly affects the cell count. The fuzzy affinity and homogeneity functions can be improved or replaced by other functions. Also, the feature space may be expanded by adding more fuzzy functions. Finding the cell region in the feature space can be determined by machine learning methods, such as neural networks. In this work, the optimum thresholds for affinity and homogeneity functions are determined manually. An automatic way to determine these
thresholds is desirable. The guard distance helps the algorithm to avoid over-counting, although it limits the ability of the algorithm to detect overlapped cells. Thus, a method to replace the guard distance is desirable, if it allows counting overlapped cells. The watershed method is a good candidate to find overlapped cells [16].

Finally, the performance of this algorithm is evaluated on only seven blood images. Therefore, more blood images are needed for evaluating the performance of this algorithm. Implementation of this algorithm on smart phones also is desirable.
Appendix 1

MATLAB CODES

The m-files that drive every stage of the proposed algorithm are presented in this appendix. The entire algorithm is implemented in ‘redCellCounting.m’, in which the parameters are defined, and it calls functions associated with every stage of the algorithm. The cell recognition algorithm is composed of five subroutines to return the coordinates of the detected cells. The sixth subroutine (cellIndicator.m) takes the image and the locations of detected cells and returns the image with each detected cell indicated by a white spot.

1. Distortion correction: DistCorr.m

```matlab
function A_corrected=DistCorr(A,lambda)
% This program is dedicated to correct the input image %which is suffering from radial distortion.
% A: input distorted image, must be gray
% lambda: distortion constant,
% for pincushion distortion lambda is positive and for %barrel distortion it is negative.
[w,l] = size(A);  
[x,y] = meshgrid(1:1,l:1:w);

midx=round(size(A,2)/2);
midy=round(size(A,1)/2);

xt=x(:)-midx;
yt=y(:)-midy;

[theta, r]=cart2pol(xt,yt);
s=r+lambda*r.^2;
[ut,vt]=pol2cart(theta,s);

u = reshape(ut,size(x)) + midx;
v = reshape(vt,size(y)) + midy;
```

```
tmap_A = cat(3,u,v);
resamp = makeresampler('linear','bound');
A_corrected = tformarray(A,[],resamp,[2 1],[1 2],[],tmap_A,1);

2. Selecting the most-focused region: FocusCrop.m

function B=FocusCrop(A,a,b,S)
% This function is supposed to crop the most focused region of image
% with size of a*b
% S is the method of criterion function which can be one of %GV, SMD, SML and TEN. If no method is specified the %default method used is GV

%Finding the center of the image and crop it by a*b size
Size=zeros(2,1); % Size vector
Size(1,1)=a;
Size(2,1)=b;
[m,n]=size(A);
CenterRow=floor(m/2);
CenterCol=floor(n/2);
tic;
if (nargin<4)
    S='GV';
end
s1=strcmp(S,'GV');
s2=strcmp(S,'SMD');
s3=strcmp(S,'SML');
s4=strcmp(S,'TEN');

UpperLeft=zeros(2,1); %Upper left point vector
UpperLeft(1,1)=CenterRow-floor(a/2);
UpperLeft(2,1)=CenterCol-floor(b/2);
UpperLeftNew=UpperLeft;

% initial coordination and focal measure
B=imcrop(A,[UpperLeft(2,1) UpperLeft(1,1) Size(2,1)-1 Size(1,1)-1]);
if (s1)
    FocLast=Variance(B);
else if (s2)
    FocLast=SMD(B);
else if (s3)
    FocLast=SML(B);
else if (s4)
    FocLast=TEN(B);
end
% 2-D Local backtracking search
for i=1:2
    True=1;
    switch i
        case 1
            Step=Size/10;
        case 2
            Step=Size/100;
    end
    while (True)
        for k=1:4
            if ((LastMove==3 && k==1) || (LastMove==4 && k==2) || ... (LastMove==1 && k==3) || (LastMove==2 && k==4))
                continue
            end
            switch k
                case 1
                    UpperLeftNew=UpperLeft;
                    UpperLeftNew(2,1)=UpperLeft(2,1)+Step(2,1);
                    B=imcrop(A,[UpperLeftNew(2,1) UpperLeftNew(1,1) Size(2,1)-1 Size(1,1)-1]);
                case 2
                    UpperLeftNew=UpperLeft;
                    UpperLeftNew(1,1)=UpperLeft(1,1)-Step(1,1);
                    B=imcrop(A,[UpperLeftNew(2,1) UpperLeftNew(1,1) Size(2,1)-1 Size(1,1)-1]);
                case 3
                    UpperLeftNew=UpperLeft;
                    UpperLeftNew(2,1)=UpperLeft(2,1)-Step(2,1);
                    B=imcrop(A,[UpperLeftNew(2,1) UpperLeftNew(1,1) Size(2,1)-1 Size(1,1)-1]);
                case 4
                    UpperLeftNew=UpperLeft;
                    UpperLeftNew(1,1)=UpperLeft(1,1)+Step(1,1);
                    B=imcrop(A,[UpperLeftNew(2,1) UpperLeftNew(1,1) Size(2,1)-1 Size(1,1)-1]);
            end
            if (s1)
                FocCurrent=Variance(B);
            elseif (s2)
                FocCurrent=SMD(B);
            end
        end
    end
end

LastMove=0;
else if (s3)
    FocCurrent=SML(B);
else if (s4)
    FocCurrent=TEN(B);
end
end

if (FocCurrent>FocLast)
    LastMove=k;
    FocLast=FocCurrent;
    UpperLeft=UpperLeftNew;
    True=1;
    break;
end
True=0;
end
end

B=imcrop(A,[UpperLeft(2,1) UpperLeft(1,1) Size(2,1)-1 Size(1,1)-1]);toc;
figure,imshow(B);
function sigma=Variance(A)
% This function takes a rect-matrix with values between 0 to 255. It
% returns
% the variance of the values in that matrix.
a=reshape(A,[],1);
x=0:255;
b=hist(a,x);
b=b/sum(b);
sigma=Var(b,x);
end

function v=Var(a,x)
% This function computes the expectation value of a random variable.
% "a"
% is the the vector corresponding with PDF of the random variable. x is
% vector representing discrete value of the random variable x. So "a"
% and
% "x" should be in the same length.
l=length(a);
m=0;m2=0;
for i=1:l
    m=x(i)*a(i)+m;
    m2=x(i)*x(i)*a(i)+m2;
end
v=m2-m*m;
function s=SMD(A)
% This function returns the Sum Modified Difference criterion function.
% It returns the higher value when the input image is highly focused
[m n]=size(A);
x=0;
y=0;
for i=1:m
    for j=2:n
        x=A(i,j)-A(i,j-1);
    end
end
for j=1:n
    for i=1:m-1
        y=A(i,j)-A(i+1,j);
    end
end
s=x+y;

function s=TEN(B)
% This function returns the Tenengrad criterion function.
% It returns the higher value when the input image is highly focused
B=double(B);
Cx=[-1 0 1;-2 0 2;-1 0 1];
Cy=[1 2 1;0 0 0;-1 -2 -1];
Ax=conv2(B,Cx);
Ay=conv2(B,Cy);

s=sum(sum((Ax.^2+Ay.^2)));

function s=SML(B)
% This function returns the Sum Modulus Laplacian criterion function.
% It returns the higher value when the input image is highly focused
s=0;
for i=2:(size(B,1)-1)
    for j=2:(size(B,2)-1)
        s=abs(2*B(i,j)-B(i-1,j)-B(i+1,j)+abs(2*B(i,j)-B(i,j-1)-B(i,j+1))+s;
    end
end

3. Image fuzzification: SegThreshInt.m

function Y=SegThreshInt(X,d,fuzzyPercent,method)
% X is gray image and d is number of segments in each x and y
% direction. Consequently the total number of segments is d^2. This
% function returns Y which is 4-level gary image, by thresholding and
% fuzzification.
% fuzzyPercent: the amount of region around the threshold that is fuzzy.
% Normally it is .1 to .2 for %10 and %20 fuzzification
% method: is the method of thresholding could be one of Otsu, MP, HCA,
% ICV
% or ME.

[m,n]=size(X);
Y=zeros(size(X));

s1=strcmp(method,'Otsu');
s2=strcmp(method,'MP');
s3=strcmp(method,'HCA');
s4=strcmp(method,'ICV');
s5=strcmp(method,'ME');

%Composing the cell of segmented matrix
x=ones(1,d);
x=x*floor(m/d);
x(d)=x(d)+mod(m,d);

y=ones(1,d);
y=y*floor(n/d);
y(d)=y(d)+mod(n,d);

C=mat2cell(X,x,y);
T=zeros(d);U=T;L=T;
NotEmpty=zeros(d);

% Finding the threshold for every sub-image and putting them in a d*d
% matrix, T
for i=1:d
    for j=1:d
        A=C{i,j};
        if (s1)
            t=graythresh(A);
        else if (s2)
            t=MP(A);
        else if (s3)
            t=HCA(A);
        else if (s4)
            t=ICV(A);
        else if (s5)
            t=ME(A);
        end
        end
    end
end
U(i,j) = (upperFuzzyReiman(A, t*255, fuzzyPercent))/255;
L(i,j) = (lowerFuzzyReiman(A, t*255, fuzzyPercent))/255;
T(i,j) = t;
NotEmpty(i,j) = GV(A);
end
end

% Checking if the accepted GV level is met at each segment or not and if it
% is not met, finding the threshold based on average value of neighbors.
Th = 0.3*mean(mean(NotEmpty));
for i=1:d
    for j=1:d
        if ((NotEmpty(i,j)<Th) && s1)
            T(i,j) = Mean(T, 1, i, j, NotEmpty, Th);
        end
    end
end

% 2D interpolation of thresholds to smoothe the
B = IntThresh(X, T);
Bu = IntThresh(X, U);
Bl = IntThresh(X, L);
for i=1:m
    for j=1:n
        if (X(i,j)<=(Bl(i,j)*255))
            Y(i,j) = 1;
        else if (X(i,j)<=B(i,j)*255)
            Y(i,j) = 2/3;
            else if (X(i,j)<=(Bu(i,j)*255))
                Y(i,j) = 1/3;
            else
                Y(i,j) = 0;
            end
        end
    end
end
figure, imshow(Y);

%**********************************************************
function sigma = GV(A)
% This function takes a rect-matrix with values between 0 to 255. It returns
% the variance of the values in that matrix
a = reshape(A,[],1);
x = 0:255;
b = hist(a,x);
b = b/sum(b);
sigma = Var(b,x);
%**********************************************************
function v=Var(a,x)
% This function computes the expectation value of a random variable.
% "a"
% is the the vector corresponding with PDF of the random variable. x is
% vector representing discrete value of the random variable x. So "a" and
% "x" should be in the same length.
% l=length(a);
l=1:length(a);
m=0;m2=0;
for i=1:l
    m=x(i)*a(i)+m;
    m2=x(i)*x(i)*a(i)+m2;
end
v=m2
%**********************************************************
function m=Mean(A,k,a,b,NE,Th)
% Calculates the Mean of neighboring segments if the segment do not meet
% the criteria for contrast

[m,n]=size(A);
s=0;
N=0;
for l=1:k
    for i=a-l:a+l
        for j=b-l:b+l
            if ((i~=a || j~=b) & (NE(i,j)>=Th))
                s=A(i,j)+s;
                N=N+1;
            end
        end
    end
end
m=s/N;
%**********************************************************
function A=IntThresh(X,T)
% return a surface for the image X where T is the %corresponding matrix
% of thresholds for each segment

[l,w]=size(X);
d=size(T,1);

m=floor(l/d);
n=floor(w/d);

x=(m/2):m:(m*d);
y=(n/2):n:(n*d);
x=[1 x(1:d) 1];
y=[1 y(1:d) w];

T=padarray(T,[1 1],'replicate','both');

[X,Y]=meshgrid(x,y);
[ XI,YI]=meshgrid(l:1:l,1:w);

A=interp2(X,Y,T, XI,YI);

%**********************************************************
function b=upperFuzzyReiman(I,t,v)
% Specifies the upper bound of fuzzy region
x=0:255;
h=hist(I(:),x);
a=0;

for i=256:-1:1
    if h(i) == 0
        continue;
    else
        a=i-1;
        break;
    end
end

b=t+floor((a-t)*v);

%**********************************************************
function b=lowerFuzzyReiman(I,t,v)
% Specifies the upper bound of fuzzy region
x=0:255;
h=hist(I(:),x);
a=0;

for i=1:256
    if h(i) == 0
        continue;
    else
        a=i-1;
        break;
    end
end

b=t-floor((t-a)*v);

4. Sweeping the cell image with the ring masks: CellFuzzy.m

function [Y1,Y2]=CellFuzzy_6(X,upperRadius,lowerRadius,homoConstant)
% This function sweeps the image by a ring mask and return the % corresponding values of Affinity and Homogeneity functions. %
% Y1: The matrix with the same size as image, every pixel returns the value % of affinity function corresponding with the location of the center of the mask % Y2: Same as Y1 for Homogeniety function % X: the fuzzy image % upperRadius and Lower Radius: Highest and lowest radius of masks used in % sweeping % homoconstant: The exponential constant (k) used in homogeneity function

[m,n]=size(X);
X=double(X);
Y1=zeros(size(X));Y2=Y1;

% Sweeping with all masks
for p=upperRadius:-1:lowerRadius
    Z=HoleFigureConnected(p);
    l=size(Z,1);
    tic;
    for i=(1+p):(m-p)
        for j=(1+p):(n-p)
            x=zeros(l,1);
            for k=1:l
                x(k)=X(i+Z(k,1),j+Z(k,2));
            end
            Y1(i,j)=sum(x)/length(x);
            Y2(i,j)=homogeneity(x,homoConstant);
        end
    end
toc;
end

%***************************************************************************************%function
Z=HoleFigureConnected(n)
% Making connected ring to be used as the mask
theta=pi/(2*n);
Z1=zeros(2*n,2);
r=n;
alpha=0;
k=1;
P=0;
while (alpha<=(pi/2))
    if (P)
        P=0;
    end

theta=pi/(2*n);
Z1=zeros(2*n,2);
r=n;
alpha=0;
k=1;
P=0;
while (alpha<=(pi/2))
    if (P)
        P=0;
    end

x=round(r*cos(alpha));
y=round(r*sin(alpha));
if (k==1)
    x_last=x;
    y_last=y;
end
if (abs(x_last-x)>1)
    F=1;
    x=x_last-1;
end
if (abs(y-y_last)>1)
    F=1;
    y=y_last+1;
end
Z1(k,1)=x;
Z1(k,2)=y;
if (~F)
    alpha=alpha+theta;
end
x_last=x;
y_last=y;
k=k+1;
end
l=k-1;
Z=zeros(4*l,2);
for i=1:l
    Z(i+0*1,1)=Z1(i,1);
    Z(i+1*1,1)=-1*Z1(i,2);
    Z(i+2*1,1)=-1*Z1(i,1);
    Z(i+3*1,1)=Z1(i,2);
    Z(i+0*1,2)=Z1(i,2);
    Z(i+1*1,2)=Z1(i,1);
    Z(i+2*1,2)=-1*Z1(i,2);
    Z(i+3*1,2)=-1*Z1(i,1);
end
%***********************************************************************
function h=homogeneity(x,exponentialConstant)
% The homogeneity function which accept the membership degrees in a vector
% x and its exponential constant as exponentialConstant

l=length(x);
s=0;
for i=2:l
    s=abs(x(i)-x(i-1))+s;
end
s=abs(x(1)-x(l))+s;
s=s/l;
h=exp(-1*exponentialConstant*s);

5. Finding cells: cellFinder.m

function Z=cellFinder(A,H,etaA,etaH,diameter)
% This finds cells using the thresholds defined for both affinity and
% homogeneity functions and guard distance
% A: Affinity function
% H: Homogeneity function
% etaA: Affinity threshold between [0,1]
% etaH: Homogeneity threshold between [0,1]
% diameter: the guard distance

[m,n]=size(A);
Z=zeros(m*n,2);
l=1;

for i=1:m
    for j=1:n
        if (A(i,j) >= etaA) && (H(i,j) >= etaH)
            Z(l,:)=[i j];
            l=l+1;
        end
    end
end
Z=Z(1:nnz(Z(:,1)),:);
Z=euclideanDistanceRemover(Z,diameter);

function A=euclideanDistanceRemover(B,n)
% It removes the coordinations
% within the guard distance
m=size(B,1);

for i=2:m
    x=B(i,:);
    for j=1:i-1
        y=B(j,:);
        d=sqrt((x(1)-y(1))^2 + (x(2)-y(2))^2);
        if d < n
            B(i,:)=[0 0];
            break;
        end
    end
end

end
A=reshape(nonzeros(B),.5*nnz(B),2);

6. To indicate cells by white spots: cellIndicator.m

function Y=cellIndicator(X,Z)
% Indicate the points which their coordination is come in Z with red dots

[m,n]=size(X);

Y_1=uint8(zeros(m,n));
Y_2=Y_1;
Y_3=Y_1;
K=[1 0; -1 0; 0 1; 0 -1; 0 0];

for i=1:m
    for j=1:n
        if inTheList(Z,[i j])
            for k=1:5
                Y_1(i+K(k,1),j+K(k,2))=uint8(255);
                Y_2(i+K(k,1),j+K(k,2))=uint8(0);
                Y_3(i+K(k,1),j+K(k,2))=uint8(0);
            end
        else if ~((inTheList(Z, [i-1 j])) || (inTheList(Z, [i+1 j])) ||
                (inTheList(Z, [i j-1])) || (inTheList(Z, [i j+1])))
            Y_1(i,j)=uint8(X(i,j));
            Y_2(i,j)=Y_1(i,j);
            Y_3(i,j)=Y_1(i,j);
        end
    end
end

Y=cat(3,Y_1,Y_2,Y_3);

******************************************************************************
function p=inTheList(X,x)
% checks if the called coordination is in the list of cells or not
l=size(X,1);
p=0;
for i=1:l
    if (x==X(i,:))
        p=1;
        break;
    end
end

7. The main function that adjusts the parameters and calls the subroutines:

redCellCounting.m

function n=redCellCounting(imageName)
% Return and indicates the cells inside the image

lambda=.0005;
cropMethod='SMD';
cropWidth=256;
cropHeight=256;

segmentationMethod='Otsu';
segmentationWindows=3;
segmentationFuzzyPercent=0.15;

maskUpperRadius=9;
maskLowerRadius=6;
homogenityConstant=4;
guardDistance=2*maskLowerRadius;

etaA=.65;
etaH=.75;

I=imread(imageName);
I=rgb2gray(I);

I_1=DistCorr(I,lambda);
I_2=FocusCrop(I_1,cropHeight,cropWidth,cropMethod);
I_3=SegThreshInt(I_2,segmentationWindows,segmentationFuzzyPercent,segmentationMethod);
[A,
H]=CellFuzzy_6(I_3,maskUpperRadius,maskLowerRadius,homogenityConstant);
Z=cellFinder(A,H,etaA,etaH,guardDistance);
Y=cellIndicator(I_2,Z);
n=size(Z,1);
figure, imshow(Y);
Appendix 2

THE CELL IMAGES

The white spots on the images indicate the locations of the detected cells. The cells with more than one spot are over-counted.

Figure A1.1. Blood cell image #1.

Figure A1.2. Blood cell image #2.
Figure A1.3. Blood cell image #3.

Figure A1.4. Blood cell image #4.
Figure A1.5. Blood cell image #5.

Figure A1.6. Blood cell image #6.
Figure A1.7. Blood cell image #7.
REFERENCES


