Sickle Blood Cell Detection Based on Image Segmentation

Kholoud Alotaibi
South Dakota State University

Follow this and additional works at: http://openprairie.sdstate.edu/etd
Part of the Biomedical Commons, and the Computer Engineering Commons

Recommended Citation
http://openprairie.sdstate.edu/etd/1116

This Thesis - Open Access is brought to you for free and open access by Open PRAIRIE: Open Public Research Access Institutional Repository and Information Exchange. It has been accepted for inclusion in Theses and Dissertations by an authorized administrator of Open PRAIRIE: Open Public Research Access Institutional Repository and Information Exchange. For more information, please contact michael.biondo@sdstate.edu.
SICKLE BLOOD CELL DETECTION BASED ON IMAGE SEGMENTATION

BY

KHLOLOUD ALOTAIBI

A thesis submitted in partial fulfillment of the requirements for the

Master of Science

Major in Computer Science

South Dakota State University

2016
SICKLE BLOOD CELLS DETECTION BASED ON IMAGE SEGMENTATION

This dissertation is approved as a creditable and independent investigation by a candidate for the Master of Computer Science degree, and is acceptable for meeting the dissertation requirements for this degree. Acceptance of this does not imply that the conclusions reached by the candidate are necessarily the conclusions of the major department.

Sung Shin, Ph.D.
Thesis Advisor

Steven Hietpas, Ph.D.
Head, Department of Electrical Engineering & Computer Science

Dean, Graduate School
ACKNOWLEDGEMENTS

Firstly, I would like to express my sincere gratitude to my advisor Prof. Sung Shin for his continuous support, patience, motivation, and immense knowledge during my master studies and thesis. His guidance helped me through the entire process of the research and writing of this thesis. I could not have imagined having a better advisor and mentor for my master studies.

Besides my advisor, I would also like to thank my committee members, Prof. George Hamer and Prof. Yi Liu for serving as my committee members even during hardship. I also want to thank you for making my defense an enjoyable moment and for your brilliant comments and suggestions. Again, thank you.

A special thanks to my husband for his support, who left his job and family and was near me whenever I need him.

My sincere thanks also goes to my family—my parents, my brothers and my sisters—for supporting me spiritually throughout the writing of this thesis and my life in general.
TABLE OF CONTENT

LIST OF FIGURES ............................................................................................. vii

LIST OF TABLES ............................................................................................... xiii

ABSTRACT ............................................................................................................. xvi

CHAPTER 1: INTRODUCTION ............................................................................. 1

1.1 Sickle Cell Disease ................................................................................... 1

1.2 What Causes Sickle Cells Disease ........................................................... 1

1.3 Symptoms of Sickle Cells Disease .......................................................... 2

1.4 How is Sickle Disease Diagnosed? .......................................................... 2

1.5 The Problem ............................................................................................. 3

1.6 Expected Results ..................................................................................... 3

CHAPTER 2: LITERATURE REVIEW ................................................................. 4

2.1 Basic Techniques for Blood Cells Detection ............................................ 5

2.1.1 Image Acquisition .............................................................................. 5

2.1.2 Image Enhancement ........................................................................... 5

2.1.3 Image Segmentation .......................................................................... 5

2.1.4 Abnormal Blood Cells Detection ..................................................... 5

2.2 Pros .......................................................................................................... 7

2.3 Cons ......................................................................................................... 7

2.4 The Result After Running the Existing Algorithm ................................... 8

CHAPTER 3: THE NEW METHOD ...................................................................... 19

3.1 The Algorithm ......................................................................................... 19
3.2 Calculating Form Factor.................................................................20
3.3 Detecting the Sickle Cells...............................................................20
3.4 The Experiment..............................................................................21

CHAPTER 4: IMPLEMENTATION................................................................31

4.1 MATLAB Environment.................................................................31
4.2 Reading Image (Imread).................................................................31
4.3 Showing Image (Imshow)...............................................................31
4.4 Converting Colored Image to Gray (Rgb2gray)..............................32
4.5 Converting Gray Image to Binary (Gray Thresh and Im2bw)...........32
4.6 Filling Image Holes (Imfill)............................................................33
4.7 Clearing the Border (Imclearborder).............................................33
4.8 Deleting Small Objects (Bwareaopen)............................................34
4.9 Labeling the Objects (Bwlabel).....................................................34
4.10 Calculating Region Properties (Regionprops)...............................35

CHAPTER 5: ANALYSIS AND RESULT..................................................36

5.1 Algorithm Steps.............................................................................36

5.1.1 Step One: Reading the Image....................................................36
5.1.2 Step Two: Convert the Image into Grayscale..............................36
5.1.3 Step Three: Convert the Grayscale Image into Binary Image........37
5.1.4 Step Four: Filling the Holes.......................................................38
5.1.5 Step Five: Clearing the Border..................................................38
5.1.6 Step six: Deleting Small Objects...............................................39
5.1.7 Step Seven: Labeling………………………………………………………40
5.1.8 Step Eight: Calculating Region Properties…………………………40
5.1.9 Step Nine: Determine Center and Corner for the Label………………41
5.1.10 Step10: Calculating the Crescent Shape……………………………41
5.1.11 Step 11: Detecting the Sickle Cells…………………………………41
5.2 The Results…………………………………………………………………44
5.3 Summary……………………………………………………………………60

CHAPTER 6: COMPARING AND RESULTING…………………………………61
6.1 Summary……………………………………………………………………76

CHAPTER 7: CONCLUSION AND FUTURE WORK………………………77
7.1 Conclusion……………………………………………………………………77
7.2 Future Work…………………………………………………………………77

REFERENCES…………………………………………………………………78
LIST OF FIGURES

Figure 1.1: Sickle Cells and Normal Blood Cells [4] ........................................... 1

Figure 2.1: Block Diagram for Detecting Abnormal Cells ............................. 4

Figure 2.2: Flowchart Diagram for Detecting Abnormal Cells [2] .............. 6

Figure 2.3: The Results of Sample 1 after Running the Existing Algorithm .... 8

Figure 2.4: The Results of Sample 2 after Running the Existing Algorithm .... 9

Figure 2.5: The Results of Sample 3 after Running the Existing Algorithm .... 9

Figure 2.6: The Results of Sample 4 after Running the Existing Algorithm ...10

Figure 2.7: The Results of Sample 5 after Running the Existing Algorithm ....10

Figure 2.8: The Results of Sample 6 after Running the Existing Algorithm ...11

Figure 2.9: The Results of Sample 7 after Running the Existing Algorithm ....11

Figure 2.10: The Results of Sample 8 after Running the Existing Algorithm ....12

Figure 2.11: The Results of Sample 9 after Running the Existing Algorithm ....13

Figure 2.12: The Results of Sample 10 after Running the Existing Algorithm ...13

Figure 2.13: The Results of Sample 11 after Running the Existing Algorithm ...14

Figure 2.14: The Results of Sample 12 after Running the Existing Algorithm ...14

Figure 2.15: The Results of Sample 13 after Running the Existing Algorithm ...15

Figure 2.16: The Results of Sample 14 after Running the Existing algorithm ...15
Figure 2.17: The Results of Sample 15 after Running the Existing Algorithm

Figure 2.18: The Results of Sample 16 after Running the Existing Algorithm

Figure 2.19: The Results of Sample 17 after Running the Existing Algorithm

Figure 2.20: The Results of Sample 18 after Running the Existing Algorithm

Figure 2.21: The Results of Sample 19 after Running the Existing Algorithm

Figure 2.21: The Results of Sample 20 after Running the Existing Algorithm

Figure 3.1: The Diagram of Sickle Cell Detection

Figure 3.2: Calculating the Form Factor

Figure 3.3: The Expected Results of Sample 1 after Running the Algorithm

Figure 3.4: The Expected Results of Sample 2 after Running the Algorithm

Figure 3.5: The Expected Results of Sample 3 after Running the Algorithm

Figure 3.6: The Expected Results of Sample 4 after Running the Algorithm

Figure 3.7: The Expected Results of Sample 5 after Running the Algorithm

Figure 3.8: The Expected Results of Sample 6 after Running the Algorithm

Figure 3.9: The Expected Results of Sample 7 after Running the Algorithm

Figure 3.10: The Expected Results of Sample 8 after Running the Algorithm

Figure 3.11: The Expected Results of Sample 9 after Running the Algorithm

Figure 3.12: The Expected Results of Sample 10 after Running the Algorithm
Figure 3.13: The Expected Results of Sample 11 after Running the Algorithm………26
Figure 3.14: The Expected Results of Sample 12 after Running the Algorithm………26
Figure 3.15: The Expected Results of Sample 13 after Running the Algorithm………27
Figure 3.16: The Expected Results of Sample 14 after Running the Algorithm………27
Figure 3.17: The Expected Results of Sample 15 after Running the Algorithm………28
Figure 3.18: The Expected Results of Sample 16 after Running the Algorithm………28
Figure 3.19: The Expected Results of Sample 17 after Running the Algorithm………28
Figure 3.20: The Expected Results of Sample 18 after Running the Algorithm………29
Figure 3.21: The Expected Results of Sample 19 after Running the Algorithm………29
Figure 3.22: The Expected Results of Sample 20 after Running the Algorithm………30
Figure 5.1: The Results after Reading SS16.jpg Image File…………………………36
Figure 5.2: The Results after Converting the Image into Grayscale…………………37
Figure 5.3: The Results after Converting the Gray Image into Binary Image…………37
Figure 5.4: The Results after Filling the Holes in the Binary Image…………………38
Figure 5.5: The Results after Clearing all the Objects Touching the Borders…………39
Figure 5.6: The Results after Deleting all Objects that have an Area < 150…………39
Figure 5.7: The Results after labeling the Objects in the Image……………………40
Figure 5.8: The Results after Detecting Sickle Cells in Sample 1……………………42
Figure 5.9: The Results of Detecting Sickle Cells in Sample 2..........................44
Figure 5.10: The Results of Detecting Sickle Cells in Sample 3..........................44
Figure 5.11: The Results of Detecting Sickle Cells in Sample 4..........................45
Figure 5.12: The Results of Detecting Sickle Cells in Sample 5.........................46
Figure 5.13: The Results of Detecting Sickle Cells in Sample 6..........................47
Figure 5.14: The Results of Detecting Sickle Cells in Sample 7.........................48
Figure 5.15: The Results of Detecting Sickle Cells in Sample 8.........................49
Figure 5.16: The Results of Detecting Sickle Cells in Sample 9.........................50
Figure 5.17: The Results of Detecting Sickle Cells in Sample 10.......................51
Figure 5.18: The Results of Detecting Sickle Cells in Sample 11.......................52
Figure 5.19: The Results of Detecting Sickle Cells in Sample 12.......................53
Figure 5.20: The Results of Detecting Sickle Cells in Sample 13.......................54
Figure 5.21: The Results of Detecting Sickle Cells in Sample 14.......................54
Figure 5.22: The Results of Detecting Sickle Cells in Sample 15.......................55
Figure 5.23: The Results of Detecting Sickle Cells in Sample 16.......................56
Figure 5.24: The Results of Detecting Sickle Cells in Sample 17.......................57
Figure 5.25: The Results of Detecting Sickle Cells in Sample 18.......................57
Figure 5.26: The Results of Detecting Sickle Cells in Sample 19.......................58
Figure 5.27: The Results of Detecting Sickle Cells in Sample 20

Figure 6.1: The Differences between the Algorithms for Sample 1

Figure 6.2: The Differences between the Algorithms for Sample 2

Figure 6.3: The Differences between the Algorithms for Sample 3

Figure 6.4: The Differences between the Algorithms for Sample 4

Figure 6.5: The Differences between the Algorithms for Sample 5

Figure 6.6: The Differences between the Algorithms for Sample 6

Figure 6.7: The Differences between the Algorithms for Sample 7

Figure 6.8: The Differences between the Algorithms for Sample 8

Figure 6.9: The Differences between the Algorithms for Sample 9

Figure 6.10: The Differences between the Algorithms for Sample 10

Figure 6.11: The Differences between the Algorithms for Sample 11

Figure 6.12: The Differences between the Algorithms for Sample 12

Figure 6.13: The Differences between the Algorithms for Sample 13

Figure 6.14: The Differences between the Algorithms for Sample 14

Figure 6.15: The Differences between the Algorithms for Sample 15

Figure 6.16: The Differences between the Algorithms for Sample 16

Figure 6.17: The Differences between the Algorithms for Sample 17
Figure 6.18: The Differences between the Algorithms for Sample 18..................74

Figure 6.19: The Differences between the Algorithms for Sample 19..................74

Figure 6.20: The Differences between the Algorithms for Sample 20..................75
LIST OF TABLES

Table 5.1: The Value of Form Factor for Sample 1 .............................................43
Table 5.2: The Value of Form Factor for Sample 3 .............................................45
Table 5.3: The Value of Form Factor for Sample 4 .............................................45
Table 5.4: The Value of Form Factor for Sample 5 .............................................46
Table 5.5: The Value of Form Factor for Sample 6 .............................................47
Table 5.6: The Value of Form Factor for Sample 7 .............................................48
Table 5.7: The Value of Form Factor for Sample 8 .............................................49
Table 5.8: The Value of Form Factor for Sample 9 .............................................50
Table 5.9: The Value of Form Factor for Sample 10 .........................................51
Table 5.10: The Value of Form Factor for Sample 11 .......................................52
Table 5.11: The Value of Form Factor for Sample 12 .......................................53
Table 5.12: The Value of Form Factor for Sample 14 .......................................55
Table 5.13: The Value of Form Factor for Sample 15 .......................................56
Table 5.14: The Value of Form Factor for Sample 19 .......................................58
Table 5.15: The Value of Form Factor for Sample 20 .......................................59
Table 5.16: Summarization the Results of 20 Samples .....................................60
Table 6.1: The Difference in Accuracy on Sample 1 .........................................61
Table 6.2: The Difference in Accuracy on Sample 2 .........................................62
<table>
<thead>
<tr>
<th>Table Number</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.3</td>
<td>The Difference in Accuracy on Sample 3</td>
<td>63</td>
</tr>
<tr>
<td>6.4</td>
<td>The Difference in Accuracy on Sample 4</td>
<td>63</td>
</tr>
<tr>
<td>6.5</td>
<td>The Difference in Accuracy on Sample 5</td>
<td>64</td>
</tr>
<tr>
<td>6.6</td>
<td>The Difference in Accuracy on Sample 6</td>
<td>65</td>
</tr>
<tr>
<td>6.7</td>
<td>The Difference in Accuracy on Sample 7</td>
<td>66</td>
</tr>
<tr>
<td>6.8</td>
<td>The Difference in Accuracy on Sample 8</td>
<td>67</td>
</tr>
<tr>
<td>6.9</td>
<td>The Difference in Accuracy on Sample 9</td>
<td>67</td>
</tr>
<tr>
<td>6.10</td>
<td>The Difference in Accuracy on Sample 10</td>
<td>68</td>
</tr>
<tr>
<td>6.11</td>
<td>The Difference in Accuracy on Sample 11</td>
<td>69</td>
</tr>
<tr>
<td>6.12</td>
<td>The Difference in Accuracy on Sample 12</td>
<td>70</td>
</tr>
<tr>
<td>6.13</td>
<td>The Difference in Accuracy on Sample 13</td>
<td>70</td>
</tr>
<tr>
<td>6.14</td>
<td>The Difference in Accuracy on Sample 14</td>
<td>71</td>
</tr>
<tr>
<td>6.15</td>
<td>The Difference in Accuracy on Sample 15</td>
<td>72</td>
</tr>
<tr>
<td>6.16</td>
<td>The Difference in Accuracy on Sample 16</td>
<td>72</td>
</tr>
<tr>
<td>6.17</td>
<td>The Difference in Accuracy on Sample 17</td>
<td>73</td>
</tr>
<tr>
<td>6.18</td>
<td>The Difference in Accuracy on Sample 18</td>
<td>74</td>
</tr>
<tr>
<td>6.19</td>
<td>The Difference in Accuracy on Sample 19</td>
<td>75</td>
</tr>
<tr>
<td>6.20</td>
<td>The Difference in Accuracy on Sample 20</td>
<td>75</td>
</tr>
</tbody>
</table>
Table 6.21: Summarization the Accuracy on 20 Samples ..........................76
ABSTRACT

SICKLE BLOOD CELL DETECTION BASED ON IMAGE SEGMENTATION
KHOLOUD ALOTAIBI
2016

Red blood cells have a vital role in human health. Red blood cells have a circular shape and a concave surface and exchange the gasses between the inside and outside of the body. However, at times, these normally round cells become sickle shaped, which is an indication of sickle cell disease.

This paper introduces a unique approach to detect sickle blood cells in blood samples using image segmentation and shape detection. This method is based on calculating the max axis and min axis of the cell. The form factor is computed using these properties to determine whether the cell is sickle or not. This method is 90 percent more accurate than the existing method.
Chapter One: Introduction

1.1. Sickle cell disease:

Sickle cell disease is an inherited blood disorder. Red blood cells contain hemoglobin which carries the oxygen in the blood. They are round and flexible which enables them to travel through small blood vessels to deliver oxygen to all parts of the body. These cells normally live about 120 days [1].

Sickle cell disease causes red blood cells to become crescent shaped, which causes them to break apart easily. This ultimately causes anemia because the cells only live 10-20 days [1].

![Figure 1.1: Sickle Cells and Normal Blood Cells](image)

1.2. What Causes Sickle Cell Disease?

Sickle cell disease is caused by a genetic abnormality in the hemoglobin gene. When oxygen is released from sickle hemoglobin, it sticks together and forms long rods, which damage and change the shape of the red blood cells [1].
1.3. Symptoms of Sickle Cell Disease:

Patients with sickle cell disease may develop severe pain in the chest, back, arms, legs, and abdomen. Pain can occur anywhere in the body. Sickle red blood cells in the lungs can cause severe illness with chest pain, fever, and difficulty breathing. Sickle cell disease can also cause permanent damage to the brain, heart, kidneys, liver, spleen, and bones. The severity and symptoms vary greatly from person to person, even within the same family.

Symptoms of sickle cell crisis include

- Severe pain
- Anemia
- Chest pain and difficulty breathing
- Strokes
- Joint pain and arthritis and bone infarctions
- Blockage of blood flow in the spleen or liver
- Severe infection [1]

1.4. How is Sickle Disease Diagnosed?

Sickle-shaped red blood cells can be seen when a blood sample is examined under a microscope. However, sickle cell disease is diagnosed by a blood test called hemoglobin electrophoresis, which measures the amount of the abnormal sickle hemoglobin. The amount of sickle hemoglobin determines whether the person is a carrier (sickle cell trait) or has sickle cell disease. Prenatal testing for sickle cell disease is possible by examining the DNA of fetal cells obtained by chorionic villus sampling or amniocentesis. Testing
newborns for sickle cell disease is required by law in all 50 states. Early detection and treatment reduces the risk of serious infections and other complications. [1].

1.5. The Problem:

In human health, blood health is very important, and the early diagnosis of diseases is necessary to prevent death and the worsening of disease.

With the development of information technology, image-processing technology is becoming an essential and effective tool in scientific research. It is especially widely used and effective in the field of biomedical engineering. [6]

Finding a faster, more efficient methods to diagnose this kind of disease by using image segmentation so that sickle cells can be detected in blood samples is an essential part of detecting this disease. The goal of this thesis is to demonstrate how image-processing technology can be useful in the detection of sickle cell disease.

1.6. Expected result:

After running the algorithm, this test detects the sickle blood cells that do not touch the borders. Moreover, the result is more accurate than existing algorithm because it differentiates between the connected normal cells that can appear as sickle cells with the previous algorithm.
Chapter Two: Literature Review

There have been many studies about sickle blood cell detection. One of them is the “Detection of Abnormal Blood Cells Using Image Processing Technique” by Malher Bhatt and Shashi Probha. In this study, the authors try to detect abnormal blood cells based on form factor and using image segmentation. The form factor they used calculates the circularity shape of the cell, and if the cell not circular, it is counted as an abnormal cell.

Another study, “Image Processing Based Abnormal Blood Cell Detection” by Deepika N. Patil and Uday P. Khot specified the threshold of each type of abnormal blood cell.

Figure 2.1: Block Diagram for Detecting Abnormal Blood Cells

Figure 2.1: Block Diagram for Detecting Abnormal Cells
2.1 Basic Techniques for Blood Cells Detection

Image processing is one branch of computer science that focuses on performing image processing in order to enhance the images or extract certain types of information that is laid out with predetermined criteria [11].

2.1.1. Image Acquisition

The digital microscope is interfaced to a computer. The microscopic images are obtained as digital images. The resolution of the digital image depends on the type of digital microscope used [2].

2.1.2. Image Enhancement

For better segmentation of the blood cells, the imported image has to be enhanced. This improves the quality of the image by enhancing the details [2].

2.1.3. Image Segmentation

Image segmentation is the process of partitioning a digital image into multiple segments (sets of pixels, also known as super pixels). The goal of segmentation is to simplify and/or change the representation of an image into something that is more meaningful and easier to analyze. Image segmentation is typically used to locate objects and boundaries (lines, curves, etc.) in images. More precisely, image segmentation is the process of assigning a label to every pixel in an image such that pixels with the same label share certain characteristics. [19].

2.1.4. Abnormal Blood Cell Detection

Form factor threshold is fixed for different abnormal cells. Based on the form factor, the abnormal cells are detected. [2]
Malher and Probha started with reading the image and then converted it to a grayscale image. They then used edge detection to detect the abnormal cells.

Edge detection is an image segmentation technique that determines the presence of an edge or line in an image and outlines the edge in an appropriate way. The main purpose of edge detection is to simplify the image data in order to minimize the amount of data to be processed [9].

Edge detection is followed by a labeling stage and then form factor calculation [2].

Form Factor = 4*Pi*area/(perimeter*perimeter) [3][12].

Figure 2.2: Flowchart Diagram for Detecting Abnormal Cells [2]
If a form factor is between 0.6 and 1, the cell is counted as a normal cell, but if the form factor is less than 0.6, the cell is counted as an abnormal cell [2]. The form factor of the sickle blood cell is less than 0.5 [3].

2.2. Pros:

This approach is fast, simple and detects the abnormal cells by calculating the circularity of the cell.

2.3. Cons:

When normal cells are connected to each other, the shape of the connected cells will not be circular, so they will count as an abnormal cell.
2.4. The Results after Running the Existing Algorithm:

The following figures show how the existing method detects the sickle cells and its performance.

Figure 2.3: The Results of Sample 1 after Running the Existing Algorithm

Figure (2.3) shows that the existing algorithm detects only one sickle cell instead of three sickle cells while it detects one normal cell as a sickle cell.
Figure (2.4) shows that the existing algorithm was unable to detect any sickle cells while it detects two normal cells as sickle cells. It is expected to detect one sickle cell.

Figure (2.5) shows that the existing algorithm is unable to detect any sickle cells while it is expected to detect six sickle cells.
Figure (2.6) shows that the existing algorithm is unable to detect any sickle cells while it detects one normal cell as a sickle cell.

Figure 2.6: The Results of Sample 4 after Running the Existing Algorithm

Figure 2.7: The Results of Sample 5 after Running the Existing Algorithm
Figure (2.7) shows that the existing algorithm detects only one sickle cell instead of six sickle cells while it detects one normal cell as a sickle cell.

Figure (2.8) shows that the existing algorithm detects two sickle cells instead of seven sickle cells while it detects three normal cells as sickle cells.

Figure 2.8: The Results of Sample 6 after Running the Existing Algorithm

Figure 2.9: The Results of Sample 7 after Running the Existing Algorithm
Figure (2.9) shows that the existing algorithm detects one sickle cell instead of two sickle cells while it detects one normal cell as a sickle cell.

Figure (2.10) shows that the existing algorithm detects only one sickle cell instead of five sickle cells while it detects two normal cells as sickle cells.
Figure (2.8) shows that the existing algorithm detects three sickle cells instead of four sickle cells.

Figure (2.12) shows that the existing algorithm detects only one sickle cell while it is expected to detect five sickle cells.
Figure (2.13) shows that the existing algorithm detects four sickle cells while it is expected to detect six sickle cells.

Figure 2.13: The Results of Sample 11 after Running the Existing Algorithm

Figure 2.14: The Results of Sample 12 after Running the Existing Algorithm
Figure (2.14) shows that the existing algorithm detects four sickle cells instead of five sickle cells while it detects three normal cells as sickle cells.

Figure 2.15: The Results of Sample 13 after Running the Existing Algorithm

Figure (2.15) shows that the existing algorithm is unable to detect any sickle cells while it is expected to detect one sickle cell.

Figure 2.16: The Results of Sample 14 after Running the Existing Algorithm
Figure (2.16) shows that the existing algorithm detects one sickle cell instead of four sickle cells while it detects three normal cells as sickle cells.

Figure 2.17: The Results of Sample 15 after Running the Existing Algorithm

Figure (2.17) shows that the existing algorithm detects only one sickle cell instead of four sickle cells.

Figure 2.18: The Results of Sample 16 after Running the Existing Algorithm
Figure (2.18) shows that the existing algorithm detects only one sickle cell that is supposed to be detected.

Figure 2.19: The Results of Sample 17 after Running the Existing Algorithm

Figure (2.19) shows that the existing algorithm is able to detect one sickle cell that is supposed to be detected. However, it detects three normal cells as sickle cells.

Figure 2.20: The Results of Sample 18 after Running the Existing Algorithm
Figure (2.20) shows that the existing algorithm detects the correct sickle cell but also detects one normal cell as a sickle cell.

Figure (2.21): The Results of Sample 19 after Running the Existing Algorithm

Figure (2.21) shows that the existing algorithm detects three sickle cells instead of six sickle cells.

Figure (2.22): The Results of Sample 20 after Running the Existing Algorithm

Figure (2.22) shows that the existing algorithm detects one sickle cell instead of two sickle cells while it detects one normal cell as a sickle cell.
Chapter Three: The New Method

3.1 The Algorithm:

Because of the problem of connected cells, the new algorithm focuses on detecting only the sickle-shaped cells.

Figure 3.1: The Diagram of Sickle Cell Detection
Figure (3.1) shows the steps of a new algorithm as follows: After the image is read, the image is converted to grayscale using "rgb2gray". Then the image is converted into black and white format for calculating the threshold using "graythresh." After that, the holes are filled using "imfill." Next, the borders are cleared and small objects are removed using "imclearborder" and "bwareaopen" respectively. Next, objects are labeled and the form factor calculated to detect the sickle cells.

3.2 Calculating Form Factor:

To calculate the form factor, first, compute the major axis length and the minor axis length of the cell. Then calculate the difference between the major and minor axes. Finally, calculate the form factor using the following formula:

\[
\text{Form Factor} = \frac{\text{Major axis length}}{\text{difference between major and minor.}}
\]

3.3. Detecting the Sickle Cells:

By testing for the following conditions, sickle cells can be detected: if the form factor is less than 1.78 then the cell will be counted as sickle cell.
3.4 The Experiment:

The algorithm is run on 20 images with different size cells to show it can work with any size image. It is expected to get the same results that appear in the following figures based on the form factor.

Figure 3.3: The Expected Results of Sample 1 after Running the Algorithm

Figure (3.3) shows that sample one contains three sickle cells that are expected to be detected after running the two algorithms.

Figure 3.4: The Expected Results of Sample 2 after Running the Algorithm
Figure (3.4) shows that sample two contains one sickle cell that is expected to be detected after running the two algorithms.

Figure 3.5: The Expected Results of Sample 3 after Running the Algorithm

Figure (3.5) shows that sample one contains six sickle cells that are expected to be detected after running the two algorithms.

Figure 3.6: The Expected Results of Sample 4 after Running the Algorithm
Figure (3.6) shows that sample three contains two sickle cells that are expected to be detected after running the two algorithms.

Figure 3.7: The Expected Results of Sample 5 after Running the Algorithm

Figure (3.7) shows that sample five contains six sickle cells that are expected to be detected after running the two algorithms.

Figure 3.8: The Expected Results of Sample 6 after Running the Algorithm
Figure (3.8) shows that sample six contains seven sickle cells that are expected to be detected after running the two algorithms.

Figure 3.9: The Expected Results of Sample 7 after Running the Algorithm

Figure (3.9) shows that sample seven contains two sickle cells that are expected to be detected after running the two algorithms.

Figure 3.10: The Expected Results of Sample 8 after Running the Algorithm
Figure (3.10) shows that sample eight contains five sickle cells that are expected to be detected after running the two algorithms.

Figure (3.11) shows that sample nine contains four sickle cells that are expected to be detected after running the two algorithms.

Figure (3.12) shows that sample 10 contains five sickle cells that are expected to be detected after running the two algorithms.
Figure (3.13) shows that sample 11 contains six sickle cells that are expected to be detected after running the two algorithms.

Figure (3.14) shows that sample 12 contains five sickle cells that are expected to be detected after running the two algorithms.
Figure 3.15: The Expected Results of Sample 13 after Running the Algorithm

Figure (3.15) shows that sample 13 contains one sickle cell that is expected to be detected after running the two algorithms.

Figure 3.16: The Expected Results of Sample 14 after Running the Algorithm

Figure (3.16) shows that sample 14 contains four sickle cells that are expected to be detected after running the two algorithms.
Figure (3.17) shows that sample 15 contains four sickle cells that are expected to be detected after running the two algorithms.

Figure (3.18) shows that sample 16 contains one sickle cell that is expected to be detected after running the two algorithms.

Figure (3.19): The Expected Results of Sample 17 after Running the Algorithm
Figure (3.19) shows that sample 17 contains one sickle cell that is expected to be detected after running the two algorithms.

Figure (3.20) shows that sample 18 contains one sickle cell that is expected to be detected after running the two algorithms.

Figure (3.21) shows that sample 19 contains six sickle cells that are expected to be detected after running the two algorithms.
Figure (3.22) shows that sample 20 contains two sickle cells that are expected to be detected after running the two algorithms.
Chapter Four: Implementation

4.1. MATLAB environment:

To implement the new algorithm, the "MATLAB" tool is used. MATLAB has an image-processing toolbox, which contains all functions that are used to analyze the image such as reading, enhancement, converting from one image type to another, segmentation, labeling and more.

The functions that used to run the new algorithm are described below.

4.2. Reading Image (Imread):

MATLAB deals with the image as a two dimensional matrix. Imread reads the image into MATLAB’s environment as a matrix.

The basic syntax:

\[ \text{Image} = \text{imread}('file-name'); \]

MATLAB reads the image file “file-name” into an array image. Imread can read many different formats such as JPG, PNG, GIF and TIF [5].

4.3. Showing Image (Imshow):

Imshow displays images on the MATLAB desktop [5].

The basic syntax:

\[ \text{Imshow} (I) \]

MATLAB displays the image I in a graphics figure, where I is a grayscale, RGB (truecolor), or binary image. For binary images, imshow displays pixels with the value 0 (zero) as black and 1 as white [5].
4.4. Converting Colored Image to Gary (rgb2gray):

There are four types of image:

1- RGB image
2- Gray – Scale images
3- Binary image “Black and white”
4- Indexed images

Rgb2gray converts the image from RGB into a gray-scale image [5].

Basic syntax:

\[ I = \text{rgb2gray}(\text{RGB}); \]

MATLAB converts the truecolor image RGB to the grayscale intensity image I. The rgb2gray function converts RGB images to grayscale by eliminating the hue and saturation information while retaining the luminance. If you have Parallel Computing Toolbox™ installed, rgb2gray can perform this conversion on a GPU [5].

4.5. Calculating Threshold and Converting Gray Image to Binary (im2bw):

Converting the image from RGB into a binary image first requires computing the threshold of the gray image using the “graythresh” function [5].

Basic Syntax:

\[ \text{Level} = \text{graythresh}(\text{RGB}); \]

MATLAB computes a global threshold (Level) that can be used to convert an intensity image to a binary image with im2bw. Level is a normalized intensity value that lies in the
range \([0, 1]\). The graythresh function uses Otsu's method, which chooses the threshold to minimize the intraclass variance of the black and white pixels.

Then this threshold converts the image to black and white using “im2bw.” [5].

Basic Syntax:

\[
BW = \text{im2bw} (\text{Grayimage}, \text{Level});
\]

MATLAB converts the grayscale image \(\text{Grayimage}\) to a binary image. The output image \(\text{BW}\) replaces all pixels in the input image with luminance greater than Level with the value 1 (white) and replaces all other pixels with the value 0 (black). It specifies a Level in the range \([0,1]\). This range is relative to the signal levels possible for the image's class. Therefore, a Level value of 0.5 is midway between black and white, regardless of class. If you do not specify Level, im2bw uses the value 0.5 [5].

4.6. Filling Images Holes (Imfill):

\[
\text{Imfill fills image regions and holes [5].}
\]

Basic syntax:

\[
\text{Filled = imfill (BW, ‘holes’);}
\]

MATLAB fills holes in the input binary image \(\text{BW}\). In this syntax, a hole is a set of background pixels that cannot be reached by filling in the background from the edge of the image [5].

4.7. Clearing the Border (Imclearborder):

\[
\text{This function clears any objects that touch the image borders [5].}
\]
Basic Syntax:

\[
\text{Cleared} = \text{imclearborder}(\text{Filled});
\]

MATLAB suppresses structures that are lighter than their surroundings and that are connected to the image border. Use this function to clear the image border. Filled can be a grayscale or binary image. For grayscale images, imclearborder tends to reduce the overall intensity level in addition to suppressing border structures. The output image, Cleared, is grayscale or binary, depending on the input. The default connectivity is 8 for two dimensions, 26 for three dimensions, and concndef(ndims(Filled),'maximal') for higher dimensions [5].

4.8. Deleting Small Objects (Bwareaopen):

This function deletes any object that's area is less than a specific value [5].

Basic Syntax:

\[
\text{Deleted} = \text{bwareaopen}(\text{Cleared}, \text{Value});
\]

MATLAB removes all connected components (objects) that have fewer Value pixels from the binary image Cleared, producing another binary image, Deleted. The default connectivity is 8 for two dimensions, 26 for three dimensions, and concndef(ndims(Cleared), 'maximal') for higher dimensions. This operation is known as an area opening [5].

4.9. Labelling the Objects (Bwlabel):

This function labels all objects in the binary image [5].

Basic Syntax:
Labeled = bwater (BW);

MATLAB returns the label matrix Labeled that contains labels for the 8-connected objects found in BW. The label matrix, Labeled, is the same size as BW [5].

4.10. Calculating Region Properties (Regionprops):

Regionprops calculates the proprieties of image objects.

Some examples of proprieties:

1- Centroid: return the center of the object
2- Boundary box: return the smallest rectangle containing the object
3- Major axis length: return the length of the major axis of the ellipse
4- Minor axis length: return the length of the minor axis of the ellipse [5].

Basic Syntax:

Stats = regionprops (BW, properties)

MATLAB returns measurements for the set of properties specified by properties for each connected component (object) in the binary image, BW. Stats is struct array containing a struct for each object in the image. You can use regionprops on contiguous regions and discontiguous regions [5].
Chapter Five: The Analysis and Results

5.1. The Algorithm Steps:

5.1.1 Step One: Reading the Image.

Code: \texttt{b=imread ('SS16.jpg');}

Read the file named “SS16.jpg” into image matrix b to analyze the image and detect the sickle blood cells.

![Figure 5.1: The Results after Reading SS16.jpg Image File](image)

5.1.2. Step Two: Convert the Image into a Grayscale image.

The blood sample image is a colored image. It should be converted to gray scale for use in coming steps that handles 2-D images [11].

Code: \texttt{c = rgb2gray (b);}
5.1.3. Step Three: Convert the Grayscale Image into Binary Image.

The image needs to be a binary image to analyze it, so it needs to be converted from grayscale to a binary image. The function graythresh calculates the threshold value of the gray image that is used to convert the gray image into binary image using im2bw.

Code: `level=graythresh(c);`  
`d=im2bw(c, level);`

Figure 5.2: The Results after Converting the Image into Grayscale

Figure 5.3: The Results after Converting the Gray Image into Binary Image
5.1.4. Step Four: Filling the Holes.

The objects that are in the previous step appear as a daunt shape instead of disk shape, so the object’s holes need to be filled using `imfill` to avoid the wrong analysis.

**Code:**
```matlab
d = imfill (~d, 'holes');
```

![Figure 5.4: The Results after Filling the Holes in the Binary Image](image)

5.1.5. Step Five: Clearing the Borders.

The objects that are connected to the boarder are incomplete. This may lead to a wrong analysis, so all these objects should be removed using `imclearborder`.

**Code:**
```matlab
d = imclearborder (d);
```
5.1.6. Step Six: Deleting Small Objects.

The result of the previous step shows there are many objects small in size. These objects can cause a wrong analysis, so these objects need to be removed.

Code: \texttt{d=bwareaopen (d, 150)};

Figure 5.5: The Results after Clearing the Objects Touching the Borders

Figure 5.6: The Results after Deleting all Objects that have an Area $< 150$
5.1.7. Step Seven: Labeling.

Each object needs to be extracted to calculate its properties and analyze it. This can be done by using `bwlabel`.

Code: 

```
[L, n] = bwlabel(d);
```

![Figure 5.7: The Results after labeling the Objects in the Image](image)

5.1.8. Step Eight: Calculating Region Properties.

The formula that determines whether the cell is sickle or not needs some properties. These properties can be calculated using `regionprops`.

Code:

```
s=regionprops(L,br,'Centroid','BoundingBox','MajorAxisLength','MinorAxisLength');
```
5.1.9. Step Nine: Determining the center and corner for the label to draw a rectangle around the sickle cell.

All the following properties are used to draw a rectangle around the sickle cells to show the results.

Code: `centroidx = C(1);`
    `centroidy = C(2);`
    `xcorner = BB(1);`
    `ycorner = BB(2);`
    `xwidth = BB(3);`
    `ywidth = BB(4);`

5.1.10. Step 10: Calculating the Crescent Shape.

The following code is used to calculate this formula:

Code: `mx = s(i).MajorAxisLength;`
    `mn = s(i).MinorAxisLength;`
    `sub = mx-mn;`
    `C = mx/sub;`

5.1.11. Step 11: Testing the Form Factor and Detecting the Sickle Cells.

The following condition tests whether the cell is sickle or not and then draws rectangle around the sickle cells.
Code:

```matlab
if(C<1.78)
    % draw rectangle around the sickle cells
    rect = [xcorner ycorner xwidth ywidth];
    rectangle ('Position', [xcorner ycorner xwidth ywidth], 'Edgecolor', 'r');
    plot (Centroidx, Centroidy, 'b*');
end
```

Figure (5.8) shows that the new algorithm is able to detect three sickle cells as expected.
Table 5.1: This Table Shows the Form Factor Values for Sample 1

<table>
<thead>
<tr>
<th></th>
<th>Form Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cell 1</td>
<td>1.5351</td>
</tr>
<tr>
<td>Cell 2</td>
<td>1.5960</td>
</tr>
<tr>
<td>Cell 3</td>
<td>1.7044</td>
</tr>
</tbody>
</table>

Table (5.1) shows the values of the form factor of the detected sickle cells. All values are less than 1.78.
5.2. The Results after Running the New Algorithm:

These figures show the results after running the algorithm on 20 samples.

Figure 5.9: The Results after Detecting Sickle Cells in Sample 2

Figure (5.9) shows that the new algorithm is able to detect one sickle cell as expected. The form factor of this cell is 1.4272 which is less than 1.78.

Figure 5.10: The Results after Detecting Sickle Cells in Sample 3
Figure (5.10) shows that the new algorithm is able to detect six sickle cells as expected.

Table 5.2: This Table Shows the Form Factor Values for Sample 3

<table>
<thead>
<tr>
<th></th>
<th>Cell 1</th>
<th>Cell 2</th>
<th>Cell 3</th>
<th>Cell 4</th>
<th>Cell 5</th>
<th>Cell 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Form Factor</td>
<td>1.3339</td>
<td>1.6255</td>
<td>1.388</td>
<td>1.5884</td>
<td>1.4235</td>
<td>1.6145</td>
</tr>
</tbody>
</table>

Table (5.2) shows the values of the form factor of the detected sickle cells. All values are less than 1.78.

Figure (5.11) shows that the new algorithm is able to detect two sickle cells as expected.

Table 5.3: This Table Shows the Form Factor Values for Sample 4

<table>
<thead>
<tr>
<th></th>
<th>Cell 1</th>
<th>Cell 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Form Factor</td>
<td>1.5910</td>
<td>1.7038</td>
</tr>
</tbody>
</table>
Table (5.3) shows the values of the form factor of the detected sickle cells. All values are less than 1.78

Figure (5.12) shows that the new algorithm is able to detect six sickle cells as expected.

Figure 5.12: The Results after Detecting Sickle Cells in Sample 5

Table 5.4: This Table Shows the Form Factor Values for Sample 5

<table>
<thead>
<tr>
<th></th>
<th>Cell 1</th>
<th>Cell 2</th>
<th>Cell 3</th>
<th>Cell 4</th>
<th>Cell 5</th>
<th>Cell 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Form Factor</td>
<td>1.4419</td>
<td>1.6889</td>
<td>1.2749</td>
<td>1.3351</td>
<td>1.7800</td>
<td>1.5179</td>
</tr>
</tbody>
</table>

Table (5.4) shows the values of the form factor of the detected sickle cells. All values are less than 1.78
Figure (5.13) shows that the new algorithm is able to detect seven sickle cells as expected.

Table 5.5: This Table Shows the Form Factor Values for Sample 6

<table>
<thead>
<tr>
<th>Form Factor</th>
<th>Cell 1</th>
<th>Cell 2</th>
<th>Cell 3</th>
<th>Cell 4</th>
<th>Cell 5</th>
<th>Cell 6</th>
<th>Cell 7</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1.3748</td>
<td>1.7731</td>
<td>1.2837</td>
<td>1.3246</td>
<td>1.3339</td>
<td>1.2966</td>
<td>1.6663</td>
</tr>
</tbody>
</table>

Table (5.5) shows the values of the form factor of the detected sickle cells. All values are less than 1.78
Figure 5.14 shows that the new algorithm is able to detect two sickle cells as expected.

Table 5.6: This Table Shows the Form Factor Values for Sample 7

<table>
<thead>
<tr>
<th>Form Factor</th>
<th>Cell 1</th>
<th>Cell 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.6166</td>
<td>1.2641</td>
<td></td>
</tr>
</tbody>
</table>

Table (5.6) shows the values of the form factor of the detected sickle cells. All values are less than 1.78.
Figure (5.15) shows that the new algorithm is able to detect six sickle cells as expected.

Table 5.7: This Table Shows the Form Factor Values for Sample 8

<table>
<thead>
<tr>
<th></th>
<th>Cell 1</th>
<th>Cell 2</th>
<th>Cell 3</th>
<th>Cell 4</th>
<th>Cell 5</th>
<th>Cell 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Form Factor</td>
<td>1.6909</td>
<td>1.3596</td>
<td>1.4890</td>
<td>1.5263</td>
<td>1.6730</td>
<td>1.74</td>
</tr>
</tbody>
</table>

Table (5.7) shows the values of the form factor of the detected sickle cells. All values are less than 1.78
Figure 5.16: The Results after Detecting Sickle Cells in Sample 9

Figure (5.16) shows that the new algorithm is able to detect four sickle cells as expected.

Table 5.8: This Table Shows the Form Factor Values for Sample 9

<table>
<thead>
<tr>
<th>Form Factor</th>
<th>Cell 1</th>
<th>Cell 2</th>
<th>Cell 3</th>
<th>Cell 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.3719</td>
<td>1.4775</td>
<td>1.3100</td>
<td>1.3172</td>
<td></td>
</tr>
</tbody>
</table>

Table (5.8) shows the values of the form factor of the detected sickle cells. All values are less than 1.78
Figure (5.16) shows that the new algorithm is able to detect four sickle cells as expected.

Table 5.9: This Table Shows the Form Factor Values for Sample 10

<table>
<thead>
<tr>
<th>Cell</th>
<th>Form Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cell 1</td>
<td>1.4383</td>
</tr>
<tr>
<td>Cell 2</td>
<td>1.3853</td>
</tr>
<tr>
<td>Cell 3</td>
<td>1.3514</td>
</tr>
<tr>
<td>Cell 4</td>
<td>1.5578</td>
</tr>
</tbody>
</table>

Table (5.9) shows the values of the form factor of the detected sickle cells. All values are less than 1.78
Figure (5.16) shows that the new algorithm is able to detect six sickle cells as expected.

Table 5.10: This Table Shows the Form Factor Values for Sample 11

<table>
<thead>
<tr>
<th>Cell</th>
<th>Form Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cell 1</td>
<td>1.2643</td>
</tr>
<tr>
<td>Cell 2</td>
<td>1.2650</td>
</tr>
<tr>
<td>Cell 3</td>
<td>1.3258</td>
</tr>
<tr>
<td>Cell 4</td>
<td>1.3111</td>
</tr>
<tr>
<td>Cell 5</td>
<td>1.3361</td>
</tr>
<tr>
<td>Cell 6</td>
<td>1.3946</td>
</tr>
</tbody>
</table>

Table (5.10) shows the values of the form factor of the detected sickle cells. All values are less than 1.78.
Figure (5.19) shows that the new algorithm detects only four sickle cells while it is expected to detect 5 sickle cells. The algorithm is unable to detect the sickle cell that touches other cells.

Table (5.11) shows the values of the form factor of the detected sickle cells. All values are less than 1.78
Figure (5.20) shows that the new algorithm is able to detect one sickle cell as expected. Its form factor is 1.3474.

Figure 5.20: The Results after Detecting Sickle Cells in Sample 13

Figure 5.21: The Results after Detecting Sickle Cells in Sample 14
Figure (5.21) shows that the new algorithm is able to detect four sickle cells as expected.

Table 5.12: This Table Shows the Form Factor Values for Sample 14

<table>
<thead>
<tr>
<th></th>
<th>Form Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cell 1</td>
<td>1.4569</td>
</tr>
<tr>
<td>Cell 2</td>
<td>1.4711</td>
</tr>
<tr>
<td>Cell 3</td>
<td>1.3162</td>
</tr>
<tr>
<td>Cell 4</td>
<td>1.2718</td>
</tr>
</tbody>
</table>

Table (5.12) shows the values of the form factor of the detected sickle cells. All values are less than 1.78.

Figure 5.22: The Results after Detecting Sickle Cells in Sample 15

Figure (5.22) shows that the new algorithm is able to detect four sickle cells as expected.
Table 5.13: This Table Shows the Form Factor Values for Sample 15

<table>
<thead>
<tr>
<th></th>
<th>Form Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cell 1</td>
<td>1.4146</td>
</tr>
<tr>
<td>Cell 2</td>
<td>1.5806</td>
</tr>
<tr>
<td>Cell 3</td>
<td>1.5818</td>
</tr>
<tr>
<td>Cell 4</td>
<td>1.4001</td>
</tr>
</tbody>
</table>

Table (5.13) shows the values of the form factor of the detected sickle cells. All values are less than 1.78

Figure (5.23) shows that the new algorithm is able to detect one sickle cell as expected. The form factor value for this cell is 1.28 which is less than 1.78
Figure 5.24: The Results after Detecting Sickle Cells in Sample 17

Form Factor of the cell = 1.4790

Figure (5.24) shows that the new algorithm is able to detect one sickle cell as expected. The form factor value for this cell is 1.4790 which is less than 1.78

Figure 5.25: The Results after Detecting Sickle Cells in Sample 18
Figure (5.25) shows that the new algorithm is able to detect one sickle cell as expected. The form factor value for this cell is 1.2358 which is less than 1.78.

![Figure 5.25: Detection of a Sickle Cell](image)

Figure (5.26) shows that the new algorithm is able to detect six sickle cells as expected.

![Figure 5.26: The Results after Detecting Sickle Cells in Sample 19](image)

Table 5.14: This Table Shows the Form Factor Values for Sample 19

<table>
<thead>
<tr>
<th>Cell</th>
<th>Form Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cell 1</td>
<td>1.6024</td>
</tr>
<tr>
<td>Cell 2</td>
<td>1.3016</td>
</tr>
<tr>
<td>Cell 3</td>
<td>1.4970</td>
</tr>
<tr>
<td>Cell 4</td>
<td>1.2497</td>
</tr>
<tr>
<td>Cell 5</td>
<td>1.4892</td>
</tr>
<tr>
<td>Cell 6</td>
<td>1.7420</td>
</tr>
</tbody>
</table>
Table (5.14) shows the values of the form factor of the detected sickle cells. All values are less than 1.78

Table 5.14:

<table>
<thead>
<tr>
<th>Cell</th>
<th>Form Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cell 1</td>
<td>1.2688</td>
</tr>
<tr>
<td>Cell 2</td>
<td>1.6196</td>
</tr>
</tbody>
</table>

Figure 5.27: The Results after Detecting Sickle Cells in Sample 20

Figure (5.27) shows that the new algorithm is able to detect two sickle cells as expected.

Table 5.15: This Table Shows the Form Factor Values for Sample 20

Table (5.15) shows the values of the form factor of the detected sickle cells. All values are less than 1.78
5.3. Summary

This table shows the form factor values for the correct results that are detected by the new algorithm. It contains the values of 20 samples. It proves that the maximum value of form factor is 1.78.

<table>
<thead>
<tr>
<th>Sample</th>
<th>Cell 1</th>
<th>Cell 2</th>
<th>Cell 3</th>
<th>Cell 4</th>
<th>Cell 5</th>
<th>Cell 6</th>
<th>Cell 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sample 1</td>
<td>1.5351</td>
<td>1.5960</td>
<td>1.7044</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sample 2</td>
<td>1.4272</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sample 3</td>
<td>1.3339</td>
<td>1.6255</td>
<td>1.388</td>
<td>1.5884</td>
<td>1.4235</td>
<td>1.6145</td>
<td></td>
</tr>
<tr>
<td>Sample 4</td>
<td>1.5910</td>
<td>1.7038</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sample 5</td>
<td>1.4419</td>
<td>1.6889</td>
<td>1.2749</td>
<td>1.3351</td>
<td>1.700</td>
<td>1.5179</td>
<td></td>
</tr>
<tr>
<td>Sample 6</td>
<td>1.3748</td>
<td>1.7731</td>
<td>1.2837</td>
<td>1.3246</td>
<td>1.3339</td>
<td>1.2966</td>
<td>1.6663</td>
</tr>
<tr>
<td>Sample 7</td>
<td>1.6166</td>
<td>1.2641</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sample 8</td>
<td>1.6909</td>
<td>1.3596</td>
<td>1.4890</td>
<td>1.5263</td>
<td>1.6730</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sample 9</td>
<td>1.3719</td>
<td>1.4775</td>
<td>1.3100</td>
<td>1.3172</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sample 10</td>
<td>1.4383</td>
<td>1.3853</td>
<td>1.3514</td>
<td>1.5578</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sample 11</td>
<td>1.2643</td>
<td>1.2650</td>
<td>1.3258</td>
<td>1.3111</td>
<td>1.3361</td>
<td>1.3946</td>
<td></td>
</tr>
<tr>
<td>Sample 12</td>
<td>1.5181</td>
<td>1.4203</td>
<td>1.3097</td>
<td>1.2607</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sample 13</td>
<td>1.3474</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sample 14</td>
<td>1.4569</td>
<td>1.4711</td>
<td>1.3162</td>
<td>1.2718</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sample 15</td>
<td>1.4146</td>
<td>1.5806</td>
<td>1.5818</td>
<td>1.4001</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sample 16</td>
<td>1.2888</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sample 17</td>
<td>1.4790</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sample 18</td>
<td>1.2358</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sample 19</td>
<td>1.6024</td>
<td>1.3016</td>
<td>1.4970</td>
<td>1.2497</td>
<td>1.4892</td>
<td>1.7420</td>
<td></td>
</tr>
<tr>
<td>Sample 20</td>
<td>1.2688</td>
<td>1.6196</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 5.16: This Table Summarizes the Results of 20 Samples
Chapter Six: Comparing and Resulting

The following figures summarizes the differences in the results between the two algorithms comparing to the expected results. It shows the accuracy in each algorithm on all 20 samples. The accuracy can be calculated using this formula:

\[ \text{Accuracy} = \frac{\text{Detected Sickle Cells}}{\text{Actual}} \times 100 \]  

![Expected Result, Circularity Based Algorithm, Crescent Shape Based Algorithm](image)

Figure 6.1: The Differences between the Algorithms for Sample 1

<table>
<thead>
<tr>
<th></th>
<th>Expected</th>
<th>Circularity</th>
<th>Crescent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positive</td>
<td>3</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Negative</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Accuracy</td>
<td></td>
<td>33%</td>
<td>100%</td>
</tr>
</tbody>
</table>

Table 6.1: The Difference in Accuracy on Sample 1

Figure (6.1) and table (6.1) show that the crescent based algorithm has more accuracy. It achieves 100% accuracy while the circularity-based algorithm achieves only 33% accuracy.
Figure 6.2: The Differences between the Algorithms for Sample 2

Table 6.2: The Difference in Accuracy on Sample 2

<table>
<thead>
<tr>
<th></th>
<th>Expected</th>
<th>Circularity</th>
<th>Crescent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positive</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Negative</td>
<td></td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>Accuracy</td>
<td>0%</td>
<td>100%</td>
<td></td>
</tr>
</tbody>
</table>

Figure 6.3: The Differences between the Algorithms for Sample 3
Table 6.3: The Difference in Accuracy on Sample 3

<table>
<thead>
<tr>
<th></th>
<th>Expected</th>
<th>Circularity</th>
<th>Crescent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positive</td>
<td>6</td>
<td>0</td>
<td>6</td>
</tr>
<tr>
<td>Negative</td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Accuracy</td>
<td>0%</td>
<td>100%</td>
<td></td>
</tr>
</tbody>
</table>

Figure 6.4: The Differences between the Algorithms for Sample 4

Table 6.4: The Difference in Accuracy on Sample 4

<table>
<thead>
<tr>
<th></th>
<th>Expected</th>
<th>Circularity</th>
<th>Crescent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positive</td>
<td>2</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>Negative</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Accuracy</td>
<td>0%</td>
<td>100%</td>
<td></td>
</tr>
</tbody>
</table>

Figure (6.2), table (6.2), figure (6.3), table (6.3), figure (6.4) and table (6.4) show that the crescent based algorithm has more accuracy. It achieves 100% accuracy while the circularity-based algorithm achieves 0% accuracy.
Figure 6.5: The Differences between the Algorithms for Sample 5

Table 6.5: The Difference in Accuracy on Sample 5

<table>
<thead>
<tr>
<th></th>
<th>Expected</th>
<th>Circularity</th>
<th>Crescent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positive</td>
<td>6</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>Negative</td>
<td></td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Accuracy</td>
<td>16%</td>
<td>100%</td>
<td></td>
</tr>
</tbody>
</table>

Figure (6.5) and table (6.5) show that the crescent based algorithm has more accuracy. It achieves 100% accuracy while the circularity-based algorithm achieves only 16% accuracy.
Figure (6.6) and table (6.6) show that the crescent based algorithm has more accuracy. It achieves 100% accuracy while the circularity-based algorithm achieves only 42% accuracy.
Figure 6.7 and table (6.7) show that the crescent based algorithm has more accuracy. It achieves 100% accuracy while the circularity-based algorithm achieves 50% accuracy.
Figure (6.8) and table (6.8) show that the crescent based algorithm has more accuracy. It achieves 100% accuracy while the circularity-based algorithm achieves only 40% accuracy.
Figure (6.9) and table (6.9) show that the crescent based algorithm has more accuracy. It achieves 100% accuracy while the circularity-based algorithm achieves 75% accuracy.

<table>
<thead>
<tr>
<th></th>
<th>Expected</th>
<th>Circularity</th>
<th>Crescent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positive</td>
<td>5</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>Negative</td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Accuracy</td>
<td>20%</td>
<td>100%</td>
<td></td>
</tr>
</tbody>
</table>

Figure (6.10) and table (6.10) show that the crescent based algorithm has more accuracy. It achieves 100% accuracy while the circularity-based algorithm achieves only 20% accuracy.
Figure 6.11 and table (6.11) show that the crescent based algorithm has more accuracy. It achieves 100% accuracy while the circularity-based algorithm achieves 66% accuracy.
Figure (6.12) and table (6.12) show that the two algorithms have the same accuracy, both achieve 80% accuracy. This case needs to improve in future work.

![Expected Result](image1) ![Circularity Based Algorithm](image2) ![Crescent Shape Based Algorithm](image3)

**Figure 6.13: The Differences between the Algorithms for Sample 13**

<table>
<thead>
<tr>
<th></th>
<th>Expected</th>
<th>Circularity</th>
<th>Crescent</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Positive</strong></td>
<td>5</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td><strong>Negative</strong></td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td><strong>Accuracy</strong></td>
<td>80%</td>
<td>80%</td>
<td></td>
</tr>
</tbody>
</table>

**Table 6.12: The Difference in Accuracy on Sample 12**

Figure (6.13) and table (6.13) show that the crescent based algorithm has more accuracy. It achieves 100% accuracy while the circularity-based algorithm achieves 0% accuracy.

<table>
<thead>
<tr>
<th></th>
<th>Expected</th>
<th>Circularity</th>
<th>Crescent</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Positive</strong></td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td><strong>Negative</strong></td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td><strong>Accuracy</strong></td>
<td>0%</td>
<td>100%</td>
<td></td>
</tr>
</tbody>
</table>

**Table 6.13: The Difference in Accuracy on Sample 13**
Expected Circularty Crescent
Positive 4 1 4
Negative 1 0
Accuracy 25% 100%

Figure (6.14) and table (6.14) show that the crescent based algorithm has more accuracy. It achieves 100% accuracy while the circularity-based algorithm achieves only 25% accuracy.
Table 6.15: The Difference in Accuracy on Sample 15

<table>
<thead>
<tr>
<th></th>
<th>Expected</th>
<th>Cirularity</th>
<th>Crescent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positive</td>
<td>4</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>Negative</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Accuracy</td>
<td>25%</td>
<td>100%</td>
<td></td>
</tr>
</tbody>
</table>

Figure (6.15) and table (6.15) show that the crescent based algorithm has more accuracy. It achieves 100% accuracy while the circularity-based algorithm achieves only 25% accuracy.

Table 6.16: The Difference in Accuracy on Sample 16

<table>
<thead>
<tr>
<th></th>
<th>Expected</th>
<th>Cirularity</th>
<th>Crescent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positive</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Negative</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Accuracy</td>
<td>100%</td>
<td>100%</td>
<td></td>
</tr>
</tbody>
</table>

Figure 6.16: The Differences between the Algorithms for Sample 16
Figure (6.16) and table (6.16) show that the crescent based algorithm and the circularity-based algorithm both achieve 100% accuracy.

<table>
<thead>
<tr>
<th></th>
<th>Expected</th>
<th>Circularity</th>
<th>Crescent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positive</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Negative</td>
<td>3</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Accuracy</td>
<td>100%</td>
<td>100%</td>
<td></td>
</tr>
</tbody>
</table>

Figure (6.17) and table (6.17) show that the crescent based algorithm and the circularity-based algorithm achieve 100% accuracy.
Figure (6.18) and table (6.18) show that the crescent based algorithm and the circularity-based algorithm both achieve 100% accuracy.

Table 6.18: The Difference in Accuracy on Sample 18

<table>
<thead>
<tr>
<th></th>
<th>Expected</th>
<th>Circularity</th>
<th>Crescent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positive</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Negative</td>
<td></td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Accuracy</td>
<td>100%</td>
<td>100%</td>
<td></td>
</tr>
</tbody>
</table>

Figure 6.18: The Differences between the Algorithms for Sample 18

Figure 6.19: The Differences between the Algorithms for Sample 19
Table 6.19: The Difference in Accuracy on Sample 19

<table>
<thead>
<tr>
<th></th>
<th>Expected</th>
<th>Circularity</th>
<th>Crescent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positive</td>
<td>6</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>Negative</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Accuracy</td>
<td>50%</td>
<td>100%</td>
<td></td>
</tr>
</tbody>
</table>

Figure (6.19) and table (6.19) show that the crescent based algorithm has more accuracy. It achieves 100% accuracy while the circularity-based algorithm achieves 50% accuracy.

Table 6.20: The Difference in Accuracy on Sample 20

<table>
<thead>
<tr>
<th></th>
<th>Expected</th>
<th>Circularity</th>
<th>Crescent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positive</td>
<td>2</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Negative</td>
<td></td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Accuracy</td>
<td>50%</td>
<td>100%</td>
<td></td>
</tr>
</tbody>
</table>

Figure 6.20: The Differences between the Algorithms for Sample 20

Table 6.20: The Difference in Accuracy on Sample 20
Figure (6.20) and table (6.20) show that the crescent based algorithm has more accuracy. It achieves 100% accuracy while the circularity-based algorithm achieves 50% accuracy.

6.1 Summary
The following table summarizes the accuracy results of 20 samples in the crescent shape based algorithm and circularity based algorithm.

<table>
<thead>
<tr>
<th>Sample</th>
<th>Circularity</th>
<th>Crescent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sample 1</td>
<td>33</td>
<td>100</td>
</tr>
<tr>
<td>Sample 2</td>
<td>0</td>
<td>100</td>
</tr>
<tr>
<td>Sample 3</td>
<td>0</td>
<td>100</td>
</tr>
<tr>
<td>Sample 4</td>
<td>0</td>
<td>100</td>
</tr>
<tr>
<td>Sample 5</td>
<td>16</td>
<td>100</td>
</tr>
<tr>
<td>Sample 6</td>
<td>42</td>
<td>100</td>
</tr>
<tr>
<td>Sample 7</td>
<td>50</td>
<td>100</td>
</tr>
<tr>
<td>Sample 8</td>
<td>40</td>
<td>100</td>
</tr>
<tr>
<td>Sample 9</td>
<td>75</td>
<td>100</td>
</tr>
<tr>
<td>Sample 10</td>
<td>20</td>
<td>100</td>
</tr>
<tr>
<td>Sample 11</td>
<td>66</td>
<td>100</td>
</tr>
<tr>
<td>Sample 12</td>
<td>80</td>
<td>80</td>
</tr>
<tr>
<td>Sample 13</td>
<td>0</td>
<td>100</td>
</tr>
<tr>
<td>Sample 14</td>
<td>25</td>
<td>100</td>
</tr>
<tr>
<td>Sample 15</td>
<td>25</td>
<td>100</td>
</tr>
<tr>
<td>Sample 16</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>Sample 17</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>Sample 18</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>Sample 19</td>
<td>50</td>
<td>100</td>
</tr>
<tr>
<td>Sample 20</td>
<td>50</td>
<td>100</td>
</tr>
</tbody>
</table>

Table (6.21) shows that the crescent based algorithm has more accuracy. It achieves 95% accuracy. However, it is unable to achieve 100% accuracy on sample 12.
Chapter Seven: Conclusion and Future Work

7.1. Conclusion

Sahu, Kumar Biswas, and Uma’s literature shows that traditional methods are very tedious, have less accuracy and depend on the patience of the technician in pathology [8].

Using image processing techniques is useful and better than existing techniques of medical diagnosis [7].

This paper introduced an algorithm to detect sickle blood cells in the blood samples using image segmentation techniques that were based on calculating form factor and then marking the sickle cells with a red rectangle.

The form factor was calculated using the properties of the cell, including max axis and min axis.

The compared results in the previous chapter prove that the new algorithm is more accurate than the existing algorithm. This algorithm achieves the 95% accuracy.

7.2. Future Work:

This paper showed that the new algorithm has high accuracy. However, it still needs to be improved to achieve 100% accuracy. In sample 12, the algorithm achieved only 80% accuracy. It detected only four out of five sickle cells. It was unable to detect the sickle cell that touched the other cells.
REFERENCES


