Automatic Red Blood Cell Counting using Watershed Segmentation

Mentor - **Manaswini**

Team Venom:
Aditya Bhadouria (2019102011) (ECE)
Ayush Kumar Lall (2020122001) (ECD)
Rishav Goenka (2019112007) (ECD)
Yash Motwani(2020122002) (ECD)

Contents

Introduction

A large number of medical images in digital format are generated by hospitals and medical institutions every day. These images are then used for diagnosis purposes. One of the significant challenges in the clinical laboratory is to produce a precise result for every test, especially in Red Blood Cell count.

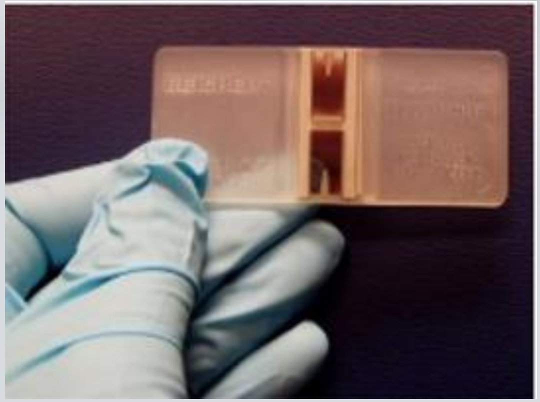
Red blood cell is one of the crucial elements that help us diagnose many patients' illnesses.

The analysis of these blood cells can give us an indication on how the body is reacting to infections, for example when infection occurs, the production of the WBCs increases, which ultimately leads to the abnormal high or low counts of these cells.

One more case we can take, the presence of bacterial infection is diagnosed from increasing WBC count.

Motivation

The conventional method used to count blood cells involves counting by Hemocytometer. This device is specially designed for the complete blood count. For counting purpose, person must view Hemocytometer through a microscope and count blood cells using hand tally count.



Drawbacks:

a) It is time consuming and laborious.

b) Counting overlapping blood cells is a major problem

c) It is difficult to get accurate results from visual inspection.

Goal

- In the field of biomedicine, because of the cell's complex nature, it remains a challenging task to segment cells from its background and count them automatically.
- So, we will make use of the properties of the blood cells, which includes texture, color, size and morphology of cell. Based on these parameters we can categorize the blood cells into 2 categories, which includes, RBCs and WBCs

Segment all the blood cells with the help of the most appropriate method described in the paper.

1. Count no. of these blood cells (red and white blood cells) present in the sample .
2. Create GUI which can help us visualize these blood cell counts. (One of the tool is MATLAB app)

Image Processing Approach

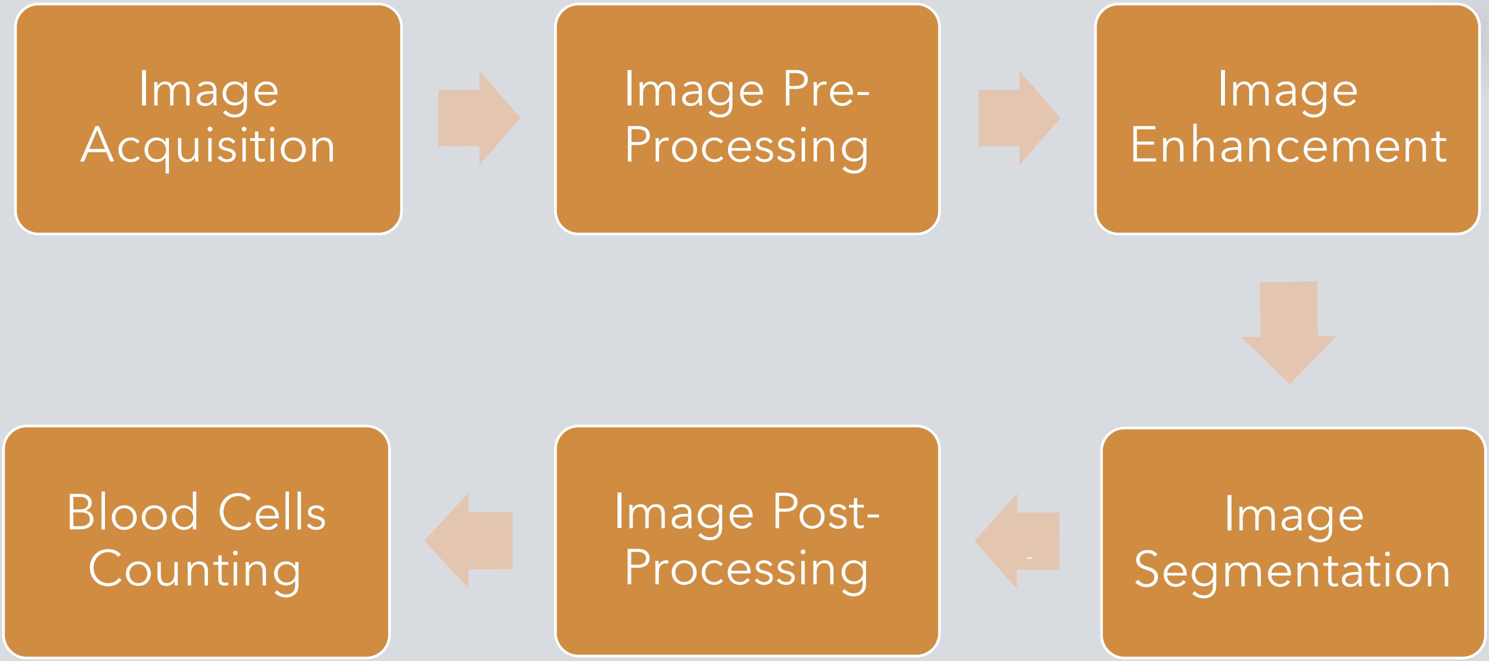
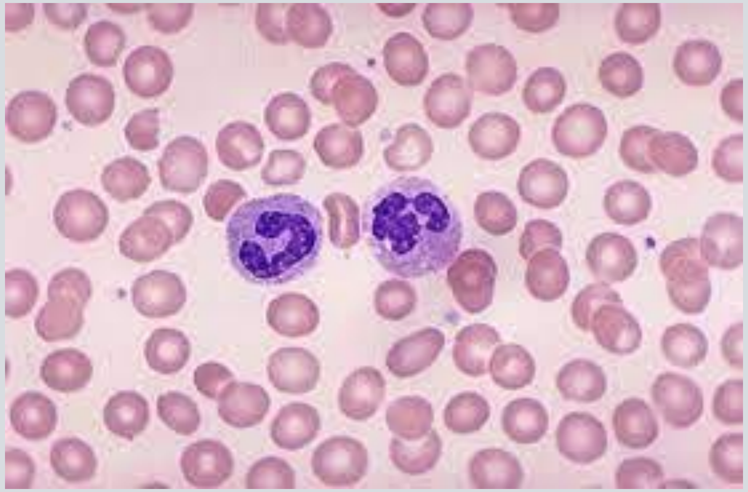


Image Acquisition

Image acquisition requires digital images of blood samples. These are microscopic image that are obtained from laboratories using digital microscopes. Images are also available online. The images we used were acquired online (through Google).



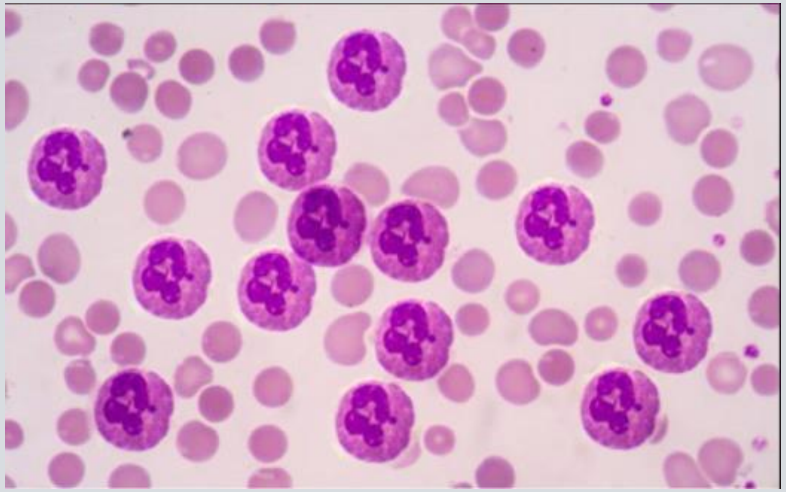


Image Pre-Processing

The acquired image is then pre-processed to improve the quality of the image. Acquired images have low contrast and due to clustered blood cells noise gets included. The pre-processing is done by first converting the original image into its grayscale image and then increasing the contrast by adaptive histogram equalization.

Adaptive Histogram Equalization

Adaptive histogram equalization operates on small regions in the image rather than the entire image. It calculates the contrast transform function for such regions individually. Each such region's contrast is enhanced, so that the histogram of the output region matches the histogram specified. The neighboring regions are then combined using bilinear interpolation to eliminate artificially induced boundaries.

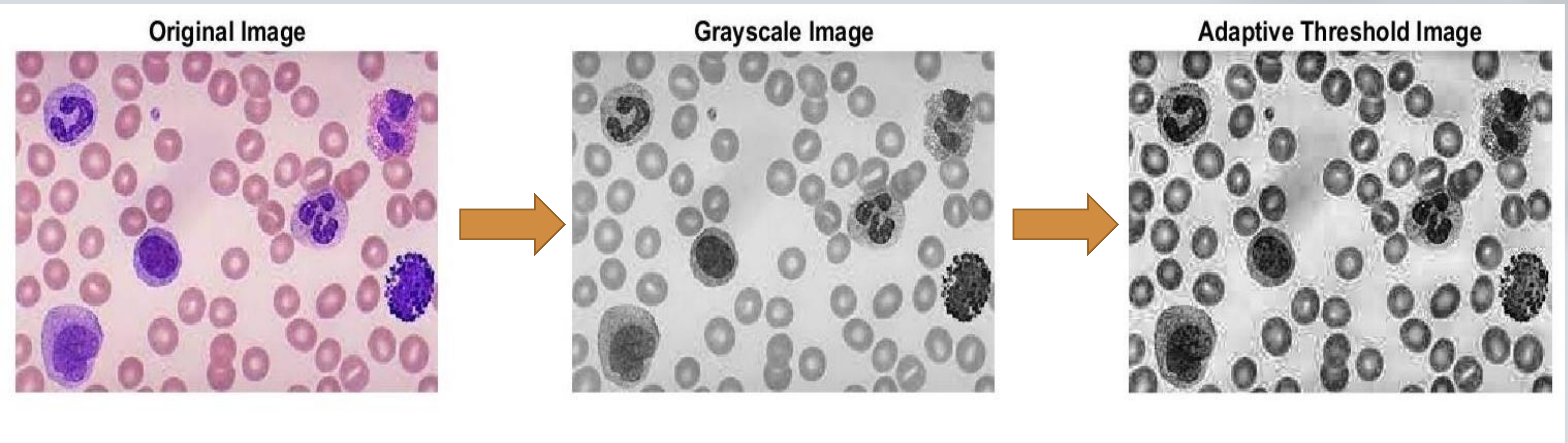
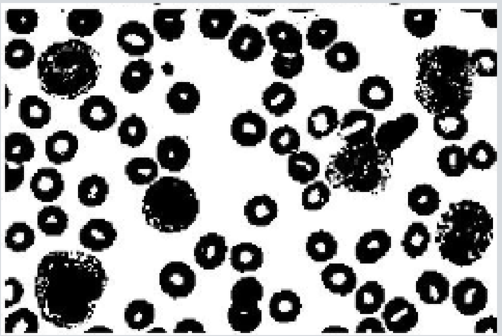

Original Image
Grayscale Image
Adaptive Threshold Image

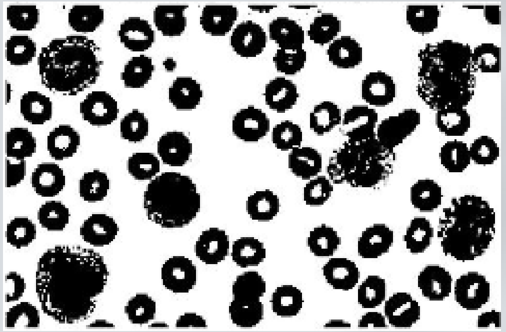
Image Enhancement

Enhancement improves the quality, contrast and brightness characteristics of an image and sharpen its details. Some of the methods include histogram equalization, image negation, image subtraction and filtering techniques. We have used adaptive histogram equalization discussed above for contrast enhancement, opening and erosion morphological operator and a combination of other morphological operations for filling holes in images.

Morphological Operation- Opening



Binarized Image

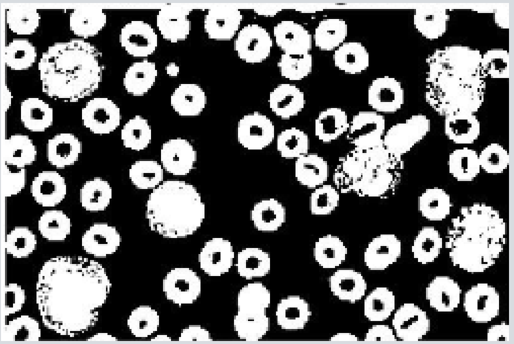


Opened Image

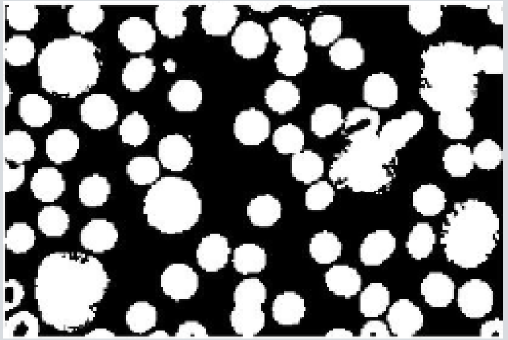
Opening removes small objects from the foreground (usually taken as the bright pixels) of an image, placing them in the background, while closing removes small holes in the foreground, changing small islands of background into foreground.

So, in our case also we tried to remove the small white spots in the image.

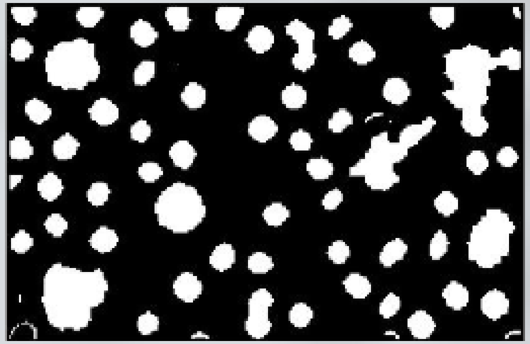
Morphological Operation- Erosion



Negation of the image



Holes filling



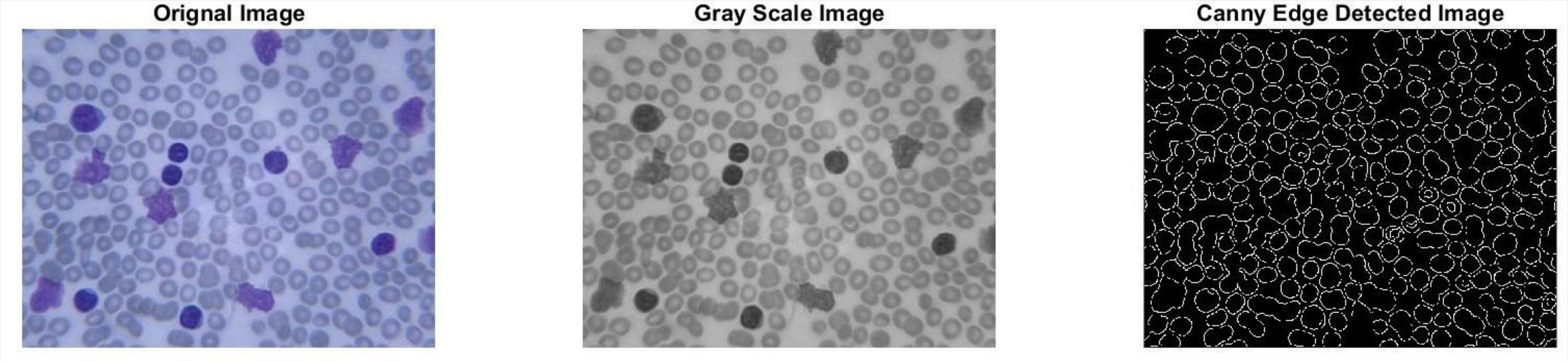
Erosion on the Image

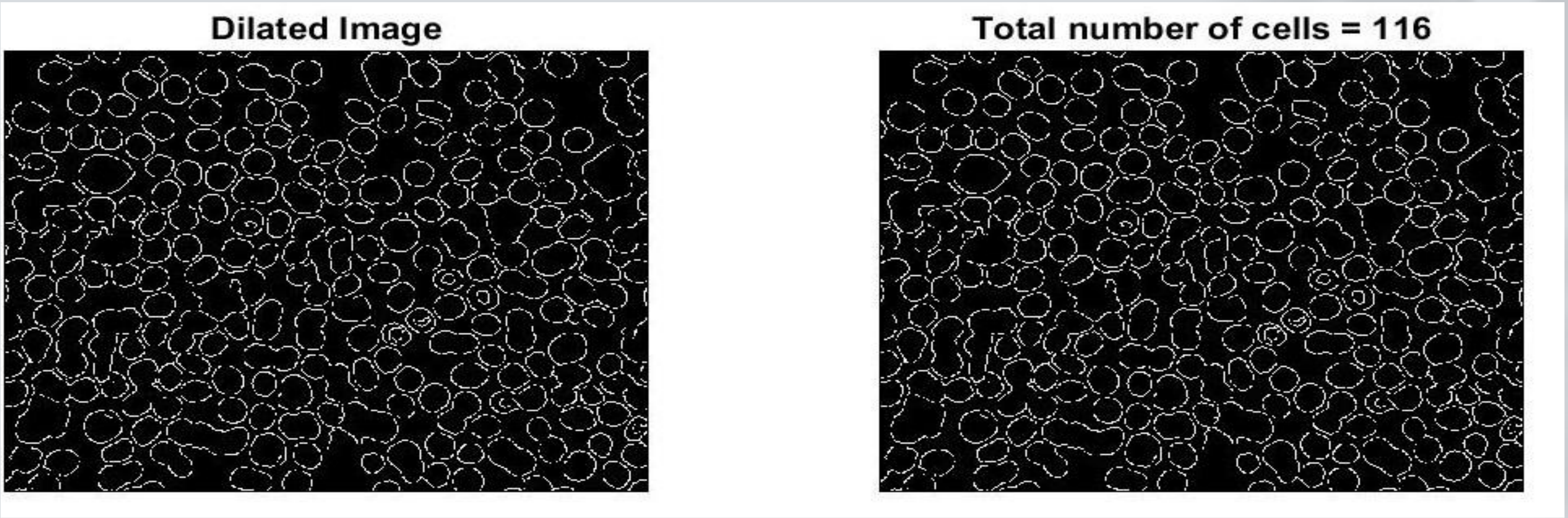
We will now negate the image, which will convert the white background to black color and black foreground to white color.

We filled the holes and applied erosion on it

Image Segmentation- Canny Detector

- It is an edge-based image segmentation. It detects the edge with low error rate, which means that the detection should accurately catch as many edges shown in the image as possible





Canny edge detector does not count the overlapped cells. Hence, it is not an optimal way for counting the red blood cells.
We will look into watershed algorithm now.

Image Segmentation- Watershed Algorithm

Image segmentation is used to separate the overlapped cells in the image. Watershed Segmentation can be used to segment contiguous regions of intersect into distinct objects.

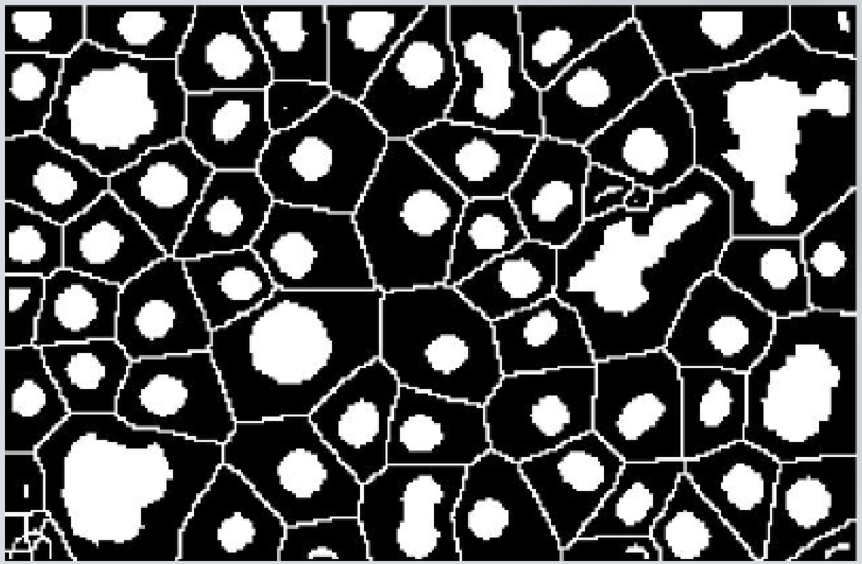
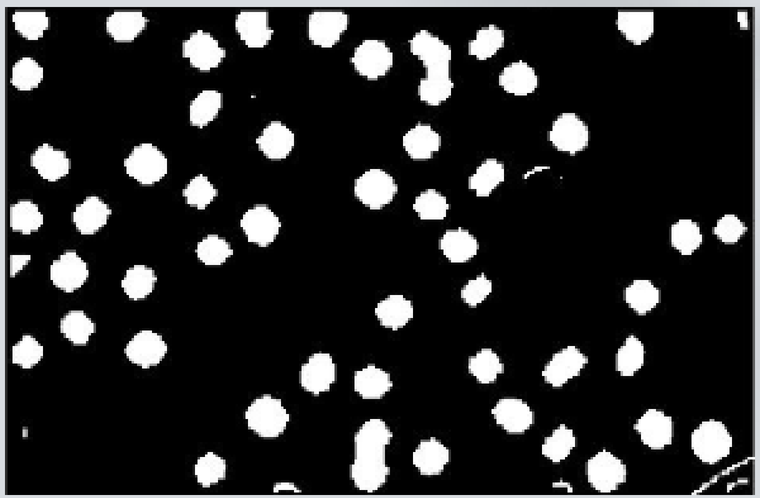


Image Post-processing

WBC Segmented



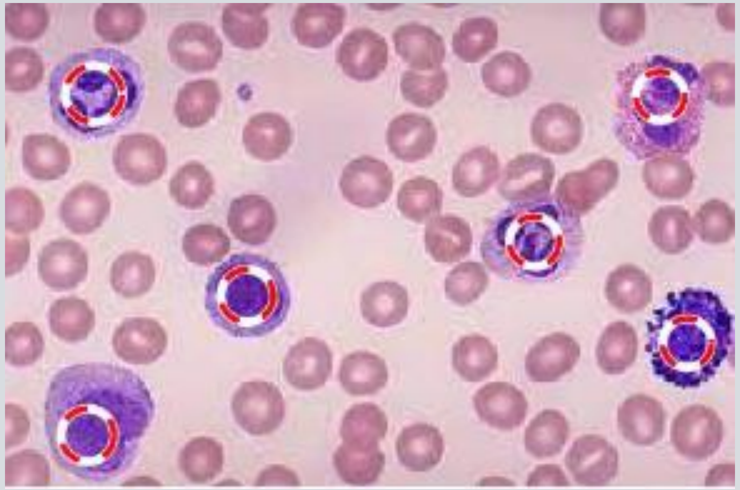
RBC Segmented

The white blood cells and red blood cells are separated by checking the image for different segments of varied radius.

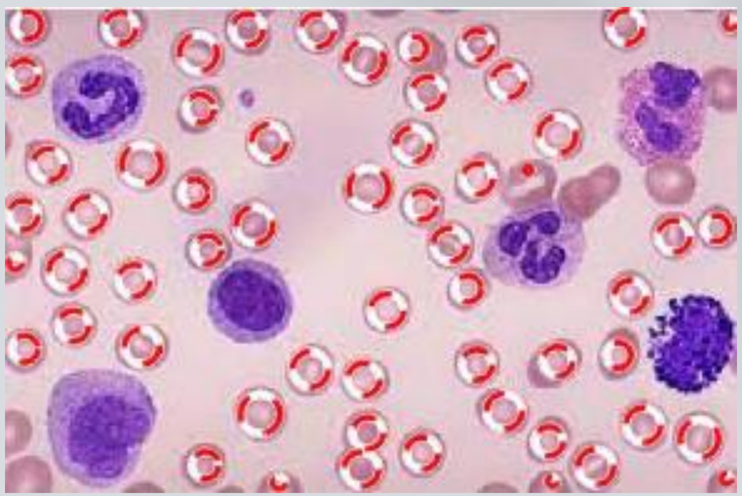
The white blood cells happen to be the ones with bigger radius whereas the red blood cells are the ones with smaller sizes.

The counting of different types of blood cells is then performed on this post-processed image to get the separate count for WBCs and RBCs.

Blood Cells Counting



WBC Count=6



RBC Count=58

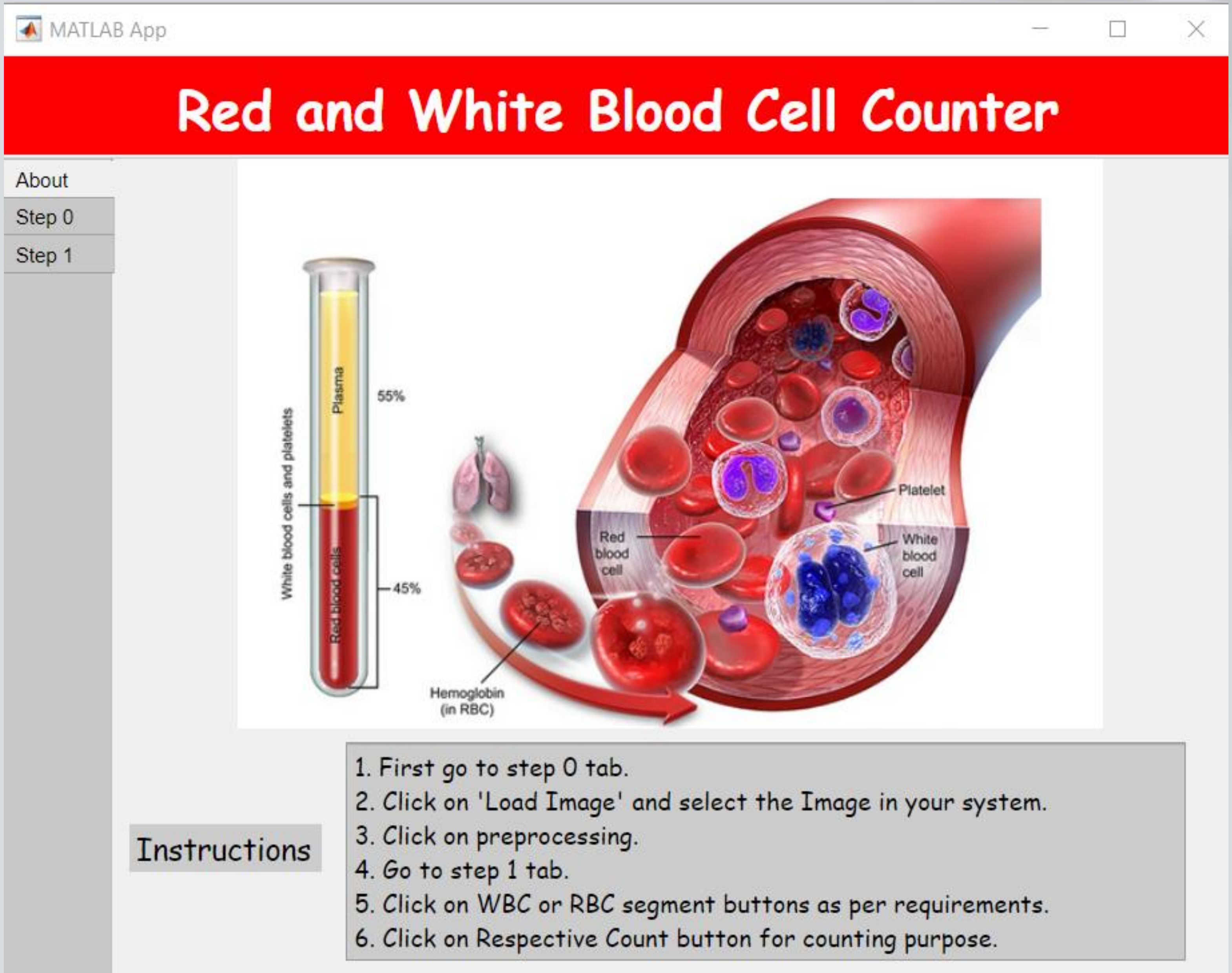
The post processed image containing only the WBCs or RBCs are then used for counting.

The circles formed from the centers of the cell calculated by the algorithm is then imprinted on the original image to get a better understanding of the image along with the WBCs or RBCs highlighted.

GUI

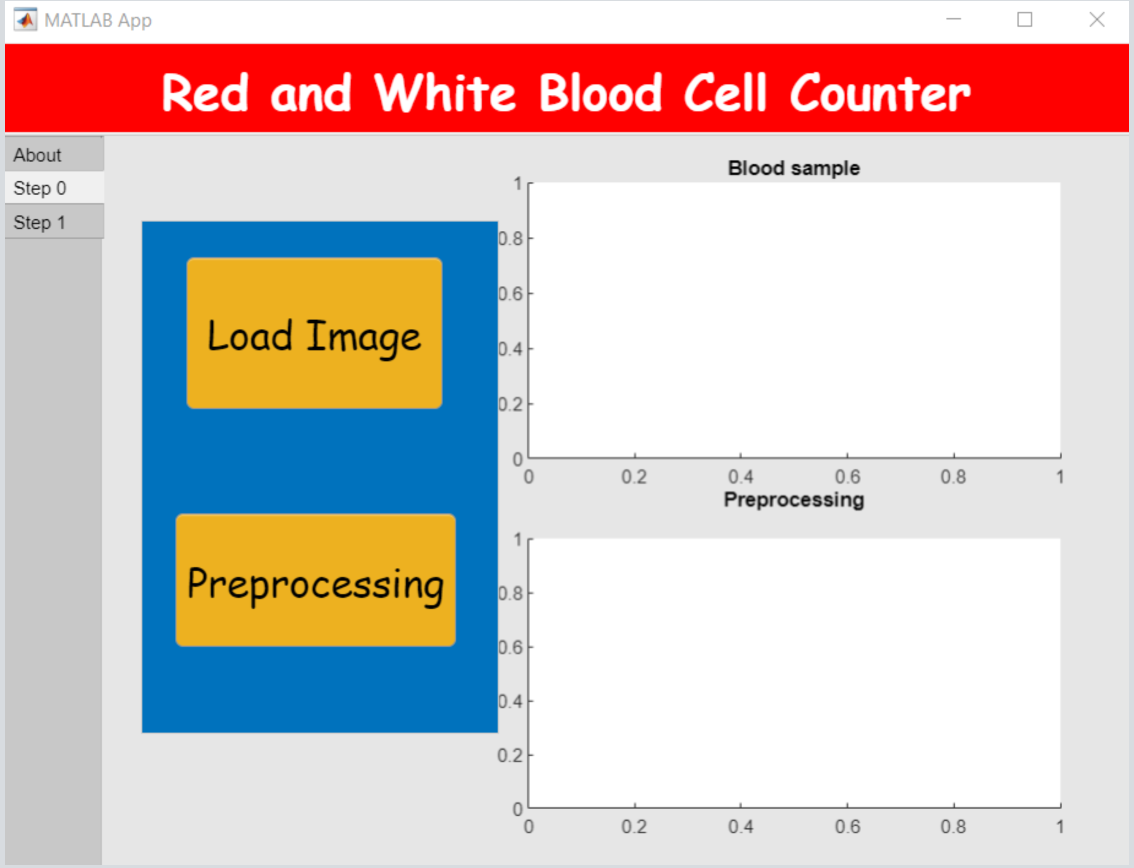
- For our project we have made a GUI with the help of MATLAB.
- The GUI is made such that, it has 2 tabs, and an about section. The two tabs are named as 'Step 0' and 'Step 1' .
- In 'Step 0' tab, we have 2 buttons in which one is for loading input image from the user and other is for performing preprocessing methods.
- After we execute all the parts in 'Step 0' tab, we will go to 'Step 1' tab. Where we will have the WBC and RBC segmentation buttons and their corresponding count buttons.
- In coming slides, we will look at the design and output .

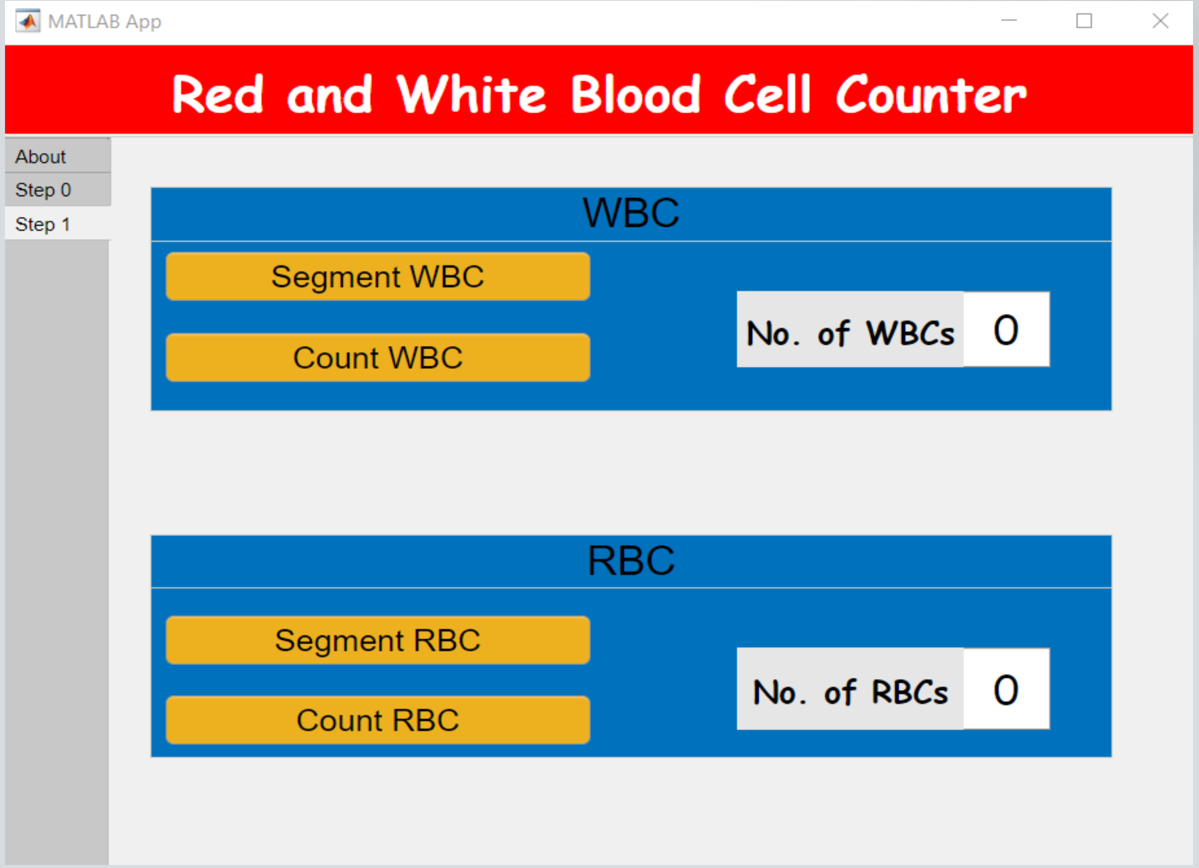
GUI Design



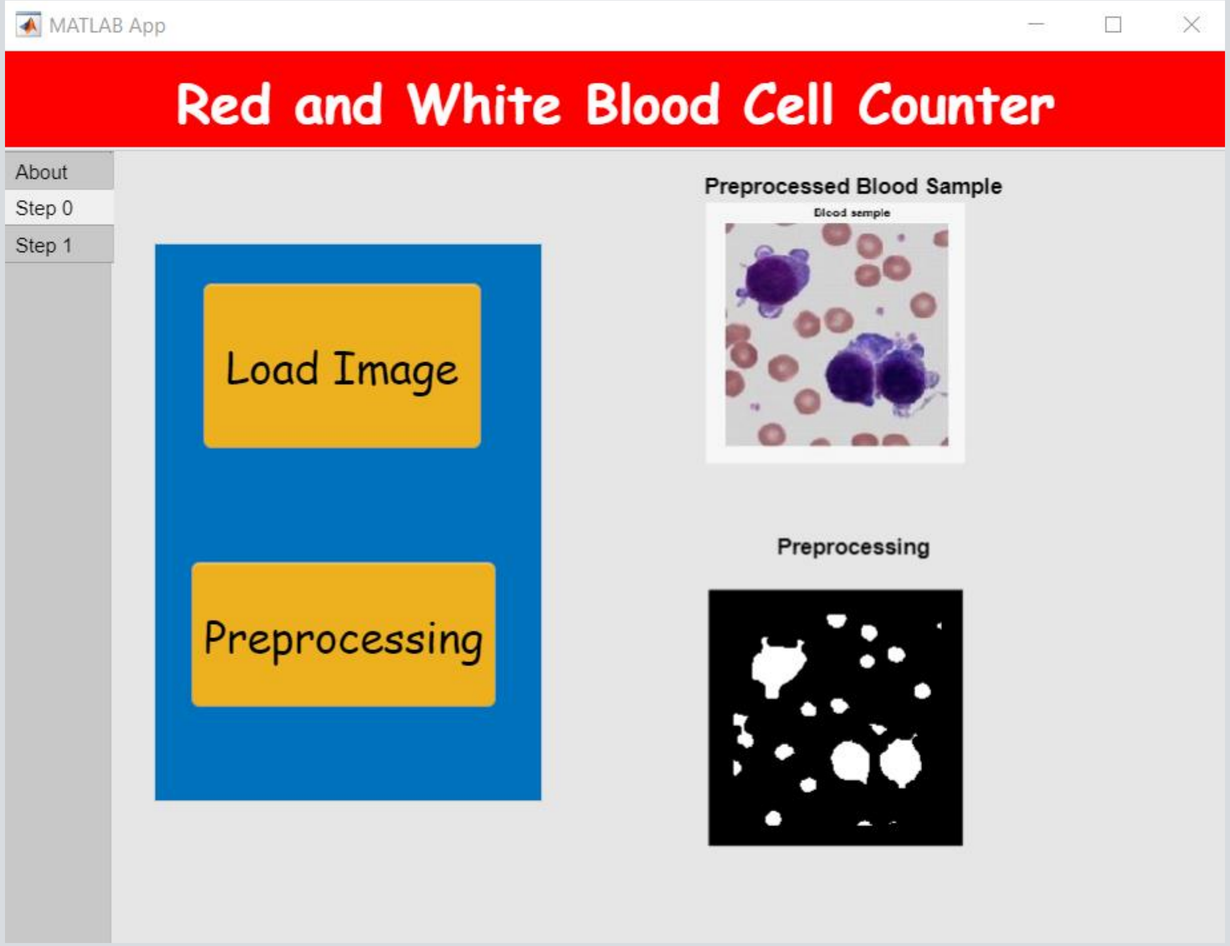
MATLAB App
Red and White Blood Cell Counter
About
Step 0
Step 1
Plasma
55%
White blood cells and platelets
Red blood cells
45%
Platelet
Red blood cell
White blood cell
Hemoglobin (in RBC)
Instructions
1. First go to step 0 tab.
2. Click on 'Load Image' and select the Image in your system.
3. Click on preprocessing.
4. Go to step 1 tab.
5. Click on WBC or RBC segment buttons as per requirements.
6. Click on Respective Count button for counting purpose.

GUI Design


MATLAB App
Red and White Blood Cell Counter
About
Step 0
Step 1
Load Image
Preprocessing
Blood sample
Preprocessing


MATLAB App
Red and White Blood Cell Counter
About
Step 0
Step 1
WBC
Segment WBC
Count WBC
No. of WBCs
0
RBC
Segment RBC
Count RBC
No. of RBCs
0

GUI Output


MATLAB App
Red and White Blood Cell Counter
About
Step 0
Step 1
Load Image
Preprocessing
Preprocessed Blood Sample
Blood sample
Preprocessing

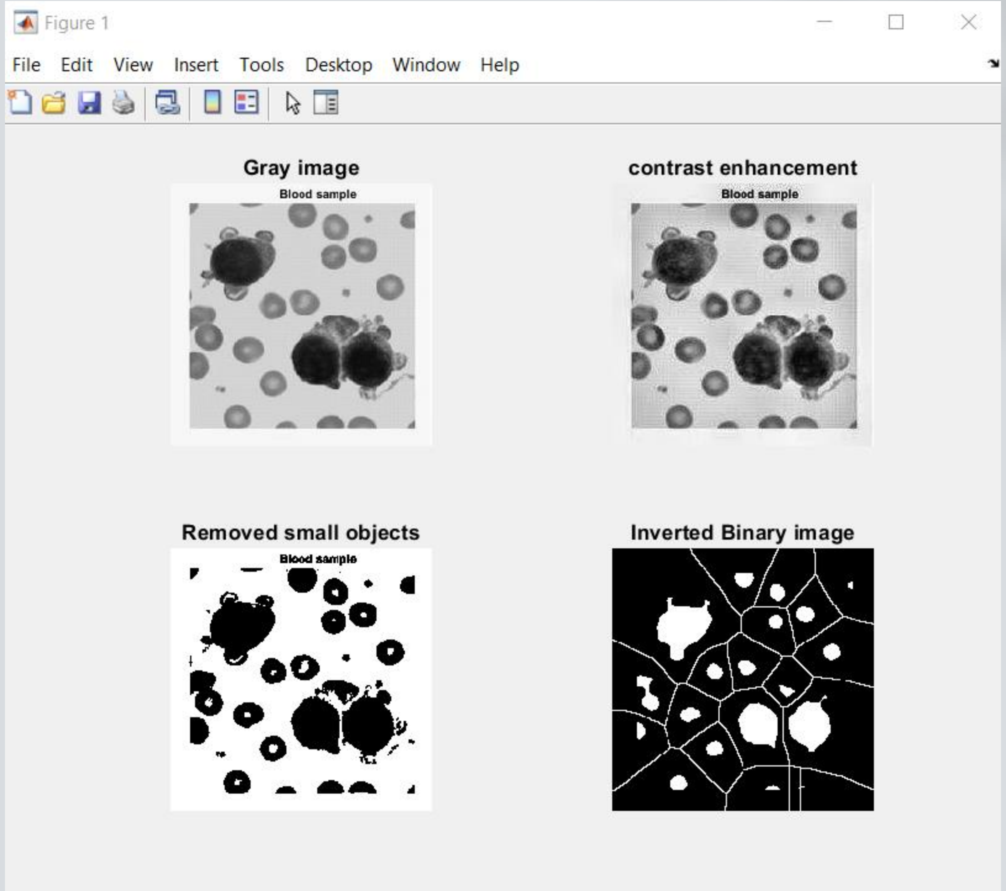
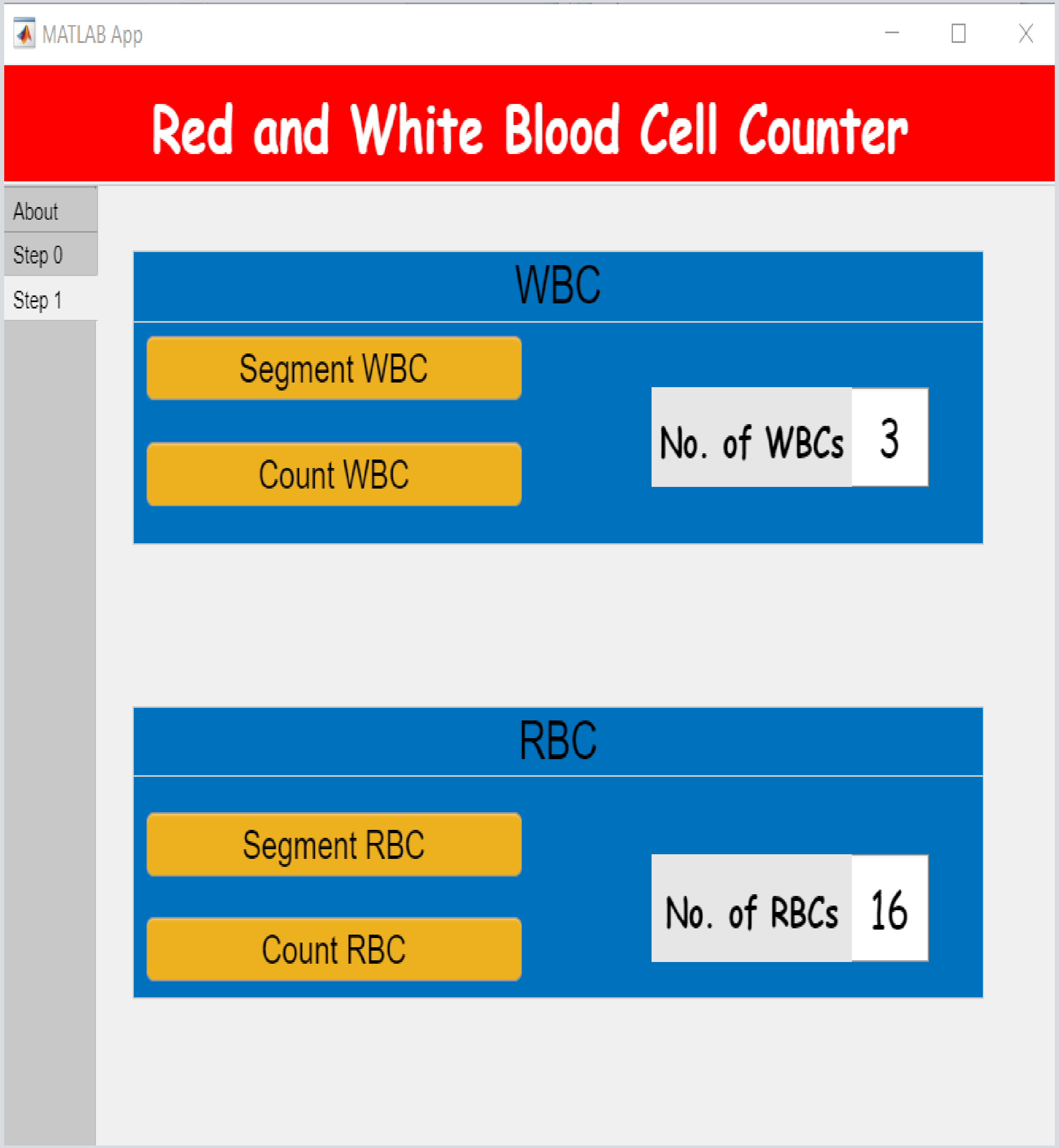


Figure 1
File Edit View Insert Tools Desktop Window Help
Gray image
Blood sample
contrast enhancement
Blood sample
Removed small objects
Blood sample
Inverted Binary image


MATLAB App
Red and White Blood Cell Counter
About
Step 0
Step 1
WBC
Segment WBC
Count WBC
No. of WBCs
3
RBC
Segment RBC
Count RBC
No. of RBCs
16

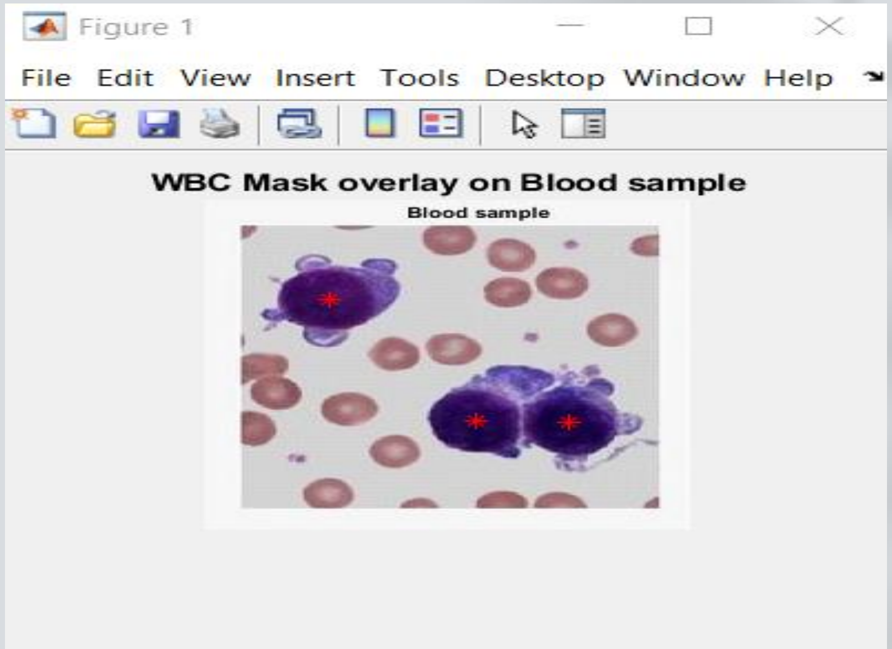


Figure 1
File Edit View Insert Tools Desktop Window Help
WBC Mask overlay on Blood sample
Blood sample

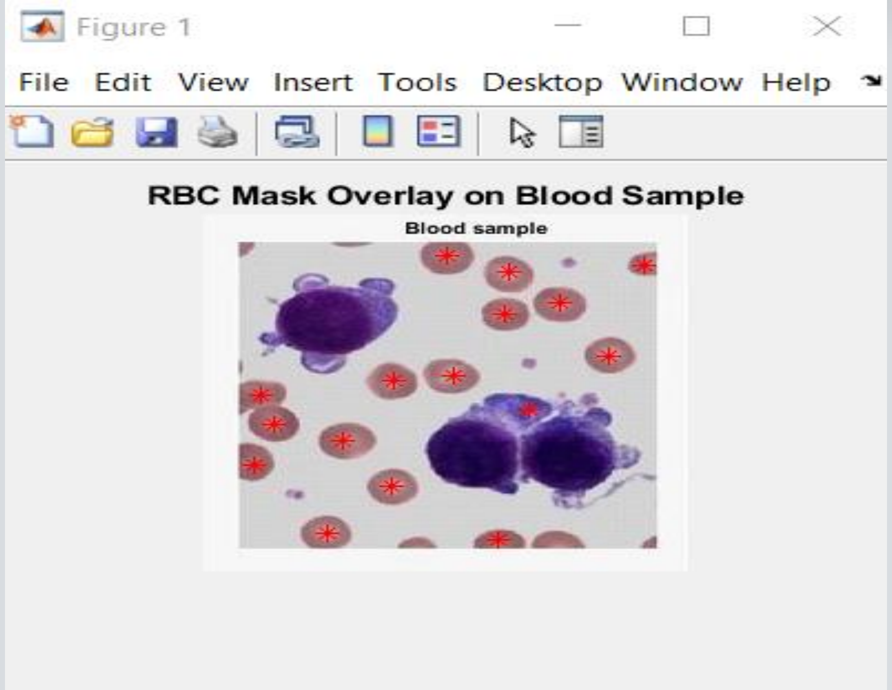


Figure 1
File Edit View Insert Tools Desktop Window Help
RBC Mask Overlay on Blood Sample
Blood sample

Conclusion

Image processing-based methods for cell counting is fast, cost effective and produces accurate results. The accuracy of a system depends upon the quality of input image. We also show that watershed segmentation algorithm is better than canny edge detection method for our requirement. We also implemented a graphical user interface for our use case to easily demonstrate the process.

Contribution

Members	
Aditya	Implemented the canny edge detector for counting the number of cells.
Ayush	Made a GUI app for RBC and WBC counting and implemented the code of it.
Rishav Yash	Implemented the "main.m" code which includes image enhancement, morphological operations, watershed segmentation and the counting of RBC and WBC.