# DEVELOPMENT OF DIGITAL PROCESSING TECHNIQUE TO CHARACTERIZE RED BLOOD CORPUSCLES

A PROJECT REPORT

*Submitted by*

SARAVANA VIJAYA KUMAR A

*In partial fulfilment for the award of the degree*

*Of*

**BACHELOR OF TECHNOLOGY**

**IN**

**BIOTECHNOLOGY**

**KUMARAGURU COLLEGE OF TECHNOLOGY, COIMBATORE**

**ANNA UNIVERSITY:: CHENNAI 600 025**

**APRIL 2010**

# ANNA UNIVERSITY: CHENNAI 600 025

## BONAFIDE CERTIFICATE

Certified that this project report "**DEVELOPMENT OF DIGITAL PROCESSING TECHNIQUE TO CHARACTERIZE RED BLOOD CORPUSCLES**", is the bonafide work of "**SARAVANA VIJAYA KUMAR A**" who carried out the project work under my supervision.

**SIGNATURE**

**Dr. S. SADASIVAM**

**DEAN**

Department of Biotechnology

Kumaraguru College of Technology

Coimbatore – 641 006

20/4/10.

**SIGNATURE**

**Dr. STEPHEN V. RAPHEAL**

**ASSISTANT PROFESSOR**

Department of Biotechnology

Kumaraguru College of Technology

Coimbatore - 641006

# CERTIFICATE OF EVALUATION

**COLLEGE : Kumaraguru College of Technology**

**BRANCH : Biotechnology**

**SEMESTER : Eighth Semester**

| NAME OF THE STUDENT | TITLE OF THE PROJECT | NAME OF THE SUPERVISOR WITH DESIGNATION |
|---|---|---|
| SARAVANA VIJAYA KUMAR (71206214035) | "DEVELOPMENT OF DIGITAL PROCESSING TECHNIQUE TO CHARACTERIZE RED BLOOD CORPUSCLES" | Dr. Stephen V. Rapheal<br>Assistant Professor<br>Dept. of Biotechnology |

The report of the project work submitted by the above student in partial fulfilment for the award of Bachelor of Technology degree in Biotechnology of Anna University was evaluated and confirmed.

(INTERNAL EXAMINER)

(EXTERNAL EXAMINER)

[illegible]

# Acknowledgements

# ACKNOWLEDGEMENTS

First and foremost, I thank the Almighty for giving me the courage and strength to carry out the project work successfully.

I wish to express our deep sense of gratitude and profound heartfelt thanks to **Dr. Stephen V. Rapheal**, Assistant Professor, for his valuable guidance, constant encouragement, constructive criticisms and enthusiastic suggestions in each and every step of this investigation.

I am highly thankful to **Dr. N. Saraswathy**, Assistant Professor and **Mr. Shanmuga Prakash,** Lecturer for their valuable input of ideas and suggestions during this investigation as Members of the Review Panel.

I wish to place on record our gratitude and heartfelt thanks to **Dr. S. Sadasivam**, Dean, for his constant motivation and support throughout the period of this investigation.

I wish to place on record our sincere thanks to **Dr. Ramalingam**, Assistant Professor and all our staff members for their constant support and valuable suggestions during the course of this study.

I wish to express our sincere gratitude to **Dr.S. Ramachandran**, Principal and the college management for their help and support during the period of this investigation.

I am highly thankful to all the non-teaching staff members of the Department of Biotechnology for their kind and patient help in all respects of this project work.

I wish to thank our friends for providing me with the moral support throughout our course of the project.

Last but not the least, I express my gratitude to my parents for their motivation, unflinching support and encouragement which has enabled me to climb the ladder of success.

SARAVANA VIJAYA KUMAR A

# Abstract

# ABSTRACT

Red blood cells (also referred to as erythrocytes) are the most common type of blood cell and the vertebrate organism's principal means of delivering oxygen ($O_2$) to the body tissues via the blood flow through the circulatory system. RBCs are stained by different staining procedures. Giemsa stain and leishman's stain are the most efficient staining techniques.

Digital signal processing is concerned with the representation of signals by a sequence of numbers or symbols and the processing of these signals. In digital image processing the analog images are converted pixels. Images of RBCs are obtained using microscopic digital camera and feed in the computer. These images are then processed using MATLAB v7. Image Processing Toolbox is used for this purpose.

Image processing toolbox is used to characterize the cells in the image. These codes are converted into an executable application using the MATLAB GUIDE, which is used to develop a Graphic User Interface. Once the application is created, it can be further modified to update the codes so that more features can be added to the original program.

# Table of Contents

# TABLE OF CONTENTS

# *Introduction*

# 1. INTRODUCTION

## 1.1 Red Blood Corpuscles

Red blood cells (also referred to as erythrocytes) are the most common type of blood cell and the vertebrate organism's principal means of delivering oxygen ($O_2$) to the body tissues via the blood flow through the circulatory system. They take up oxygen in the lungs or gills and release it while squeezing through the body's capillaries. These cells' cytoplasm is rich in hemoglobin, an iron-containing biomolecule that can bind oxygen and is responsible for the blood's red color. In humans, mature red blood cells are flexible biconcave disks that lack a cell nucleus and most organelles.

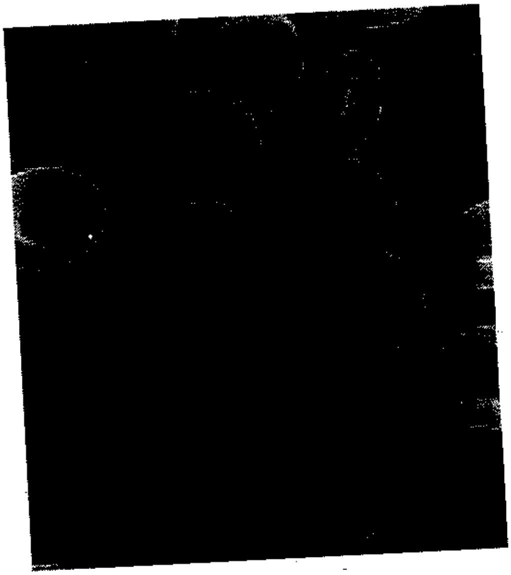


**Figure 1.1 Human Red Blood Cells**

A typical human erythrocyte has a disk diameter of 6–8 µm and a thickness of 2 µm, being much smaller than most other human cells. These cells have a volume of about 90 fL with a surface of about 136 µm$^2$, and can swell up to a sphere shape containing 150 fL, without membrane distension.

Adult humans have roughly 2–3 × $10^{13}$ (20-30 trillion) red blood cells at any given time, comprising approximately one quarter of the total human body cell number (women have about 4 to 5 million erythrocytes per microliter (cubic millimeter) of blood and men about 5 to 6 million.

## 1.2 Staining Red Blood Cells

Staining is an auxiliary technique used in microscopy to enhance contrast in the microscopic image. Stains and dyes are frequently used in biology and medicine to highlight structures in biological tissues for viewing, often with the aid of different microscopes. Stains may be used to define and examine bulk tissues (highlighting, for example, muscle fibers or connective tissue), cell populations (classifying different blood cells, for instance), or organelles within individual cells.

Red blood cells must be stained for the viewing under the microscope. Different techniques are available for staining red blood cells. The techniques are

### 1.2.1 Giemsa's Stain

Giemsa's stain is a member of the Romanowski group of stains, which are defined as being the black precipitate formed from the addition of aqueous solutions of methylene blue and eosin, dissolved in methanol. The variants of the Romanowski group differ in the degree of oxidation (polychroming) of the methylene blue stain prior to the precipitation.

The stain class was originally designed to incorporate cytoplasmic (pink) staining with nuclear (blue) staining and fixation as a single step for smears and thin films of tissue (spread preparations of omentum). Minor modifications of working stain concentration and staining time have been made over the years for fixed tissue sections.

### 1.2.2 Leishman's Stain

Leishman's stain, also Leishman stain, is used in microscopy for staining blood smears. It provides excellent stain quality. It is generally used to differentiate and identify leucocytes, malariaparasites, and trypanosomas. It is based on a mixture of methylene blue and eosin.

Leishman stain uses a methanol solution of staining dyes. 7-10 drops is applied to the slide with the specimen. After 5 minutes, 10-15 drops of a buffer solution (a Gurr buffer is used, pH 6.8) is added and mixed with the stain, then the specimen is left staying for 20–30 minutes, then washed off with the buffer solution.

### 1.2.3 Wright's stain

Wright's stain is a histologic stain that facilitates the differentiation of blood cell types. It is used primarily to stain peripheral blood smears and bone marrow aspirates which are examined under alight microscope. In cytogenetics it is used to stain chromosomes to facilitate diagnosis of syndromes and diseases.

It is named for James Homer Wright, who devised the stain, a modification of the Romanowsky stain, in 1902. Because it distinguishes easily between blood cells, it became widely used for performing differential white blood cell counts, which are routinely ordered when infections are expected.

### 1.2.4 Jenner's Stain

The Jenner stain Solution is a mixture of several thiazin dyes in a methanol solvent. Ionic and noionic forces are involved in the binding of these dyes. The staining solution has anionic and cationic properties. The negatively charged phosphoric acid groups of DNA attract the purple polychromatic cationic dyes to the nuclei. The blue basophilic granules are stained by the polychromatic cationic dyes. Cationic cellular components, such as erythrocytes and eosinophilic granules, are stained by the red and pink anionic dyes. The buffers used in the staining procedure liberate and activate dye ions allowing them to chemically bond with specific cellular components.

### 1.2.5 Field's Stain

Field stain is a histological method for staining of blood smears. It is used for staining thin blood films in order to discover malarial parasites. Field's stain is a version of a Romanowsky stain, used for rapid processing of the specimens. Field's stain uses methylene blue and Azure 1 dissolved in phosphate buffer solution, and Eosin Y in buffer solution.

## 1.3 Image Acquisition

Acquisition is usually done using a CCD camera mounted in the optical path of the microscope. The camera may be full colour or monochrome. Very often, very high resolution cameras are employed to gain as much direct information as possible. Often digital cameras used for this application provide pixel data.

For best results, appropriate sensor for a given application must be selected. A modest detector, with larger pixels, can produce higher quality images because of reduced noise. Cameras with 7-10 Megapixels are in fluorescence microscopy.

A lower resolution detector will often have a significantly higher acquisition rate, permitting the observation of faster events. Conversely, if the observed object is motionless, one may wish to acquire images at the highest possible spatial resolution without regard to the time required to acquire a single image.

## 1.4 Digital Signal Processing

Digital signal processing is concerned with the representation of signals by a sequence of numbers or symbols and the processing of these signals. Digital signal processing and analog signal processing are subfields of signal processing. DSP includes subfields like: audio and speech signal processing, sonar and radar signal processing, sensor array processing, spectral estimation, statistical signal processing, digital image processing, signal processing for communications, control of systems, biomedical signal processing, seismic data processing, etc.

The goal of DSP is usually to measure, filter and/or compress continuous real-world analog signals, the first step is usually to convert the signal from an analog to a digital form, by sampling it using an analog-to-digital converter (ADC), which turns the analog signal into a stream of numbers. However, often, the required output signal is another analog output signal, which requires a digital-to-analog converter (DAC). Even if this process is more complex than analog processing and has a discrete value range, the application of computational power to digital signal processing allows for many advantages over analog processing in many applications, such as error detection and correction in transmission as well as data compression.

### 1.4.1 Image Processing

Image processing is any form of signal processing for which the input is an image, such asphotographs or frames of video; the output of image processing can be either an image or a set of characteristics or parameters related to the image. Most image-processing techniques involve treating the image as a two-dimensional signal and applying standard signal-processing techniques to it. Image processing usually refers to digital image processing, but optical and analog image processing are also possible. This article is about general techniques that apply to all of them.

## 1.5 Graphical User Interface

A graphical user interface (GUI) is a type of user interface item that allows people to interact with programs in more ways than typing such as computers; hand-held devices such as MP3 Players, Portable Media Players or Gaming devices; household appliances and office equipment with images rather than text commands. A GUI offers graphical icons, and visual indicators, as opposed to text-based interfaces, typed command labels or text navigation to fully represent the information and actions available to a user. The actions are usually performed through direct manipulation of the graphical elements.

Designing the visual composition and temporal behavior of GUI is an important part of software application programming. Its goal is to enhance the efficiency and ease of use for the underlying logical design of a stored program, a design discipline known as usability. Techniques of user-centered design are used to ensure that the visual language introduced in the design is well tailored to the tasks it must perform

# Objectives

## 2. OBJECTIVES

The project was carried out to

Stain the red blood cells from different Human samples

Acquire the digital images of the stained cells

Develop an algorithm to characterize the cells

Develop a graphical user interface for the algorithm

# Literature Review

# 3. LITERATURE REVIEW

## 3.1 Noise Reduction

Noise reduction is the process of removing noise from a signal. Noise reduction techniques are conceptually very similar regardless of the signal being processed, however a priori knowledge of the characteristics of an expected signal can mean the implementations of these techniques vary greatly depending on the type of signal.

All recording devices, both analogue or digital, have traits which make them susceptible to noise. Noise can be random or white noise with no coherence or coherent noise introduced by the devices mechanism or processing algorithms.

In electronic recording devices, a major form of noise is hiss caused by random electrons that, heavily influenced by heat, stray from their designated path. These stray electrons influence the voltage of the output signal and thus create detectable noise.

In the case of photographic film and magnetic tape, noise (both visible and audible) is introduced due to the grain structure of the medium. In photographic film, the size of the grains in the film determines the film's sensitivity, more sensitive film having larger sized grains. In magnetic tape, the larger the grains of the ma mages taken with both digital cameras and conventional film cameras will pick up noise from a variety of sources. Many further uses of these images require that the noise will be (partially) removed - foraesthetic purposes as in artistic work or marketing, or for practical purposes such as computer vision.

### 3.1.1Types

In salt and pepper noise (sparse light and dark disturbances), pixels in the image are very different in color or intensity from their surrounding pixels; the defining characteristic is that the value of a noisy pixel bears no relation to the color of surrounding pixels. Generally this type of noise will only affect a small number of image pixels. When viewed, the image contains dark and white dots, hence the term salt and pepper noise. Typical sources include flecks of dust inside the camera, or with digital cameras, faulty CCD elements.

In Gaussian noise, each pixel in the image will be changed from its original value by a (usually) small amount. A histogram, a plot of the amount of distortion of a pixel value against the frequency with which it occurs, shows a normal distribution of noise. While other distributions are possible, the Gaussian (normal) distribution is usually a good model, due to the central limit theorem that says that the sum of different noises tends to approach a Gaussian distribution.

In either case, the noises at different pixels can be either correlated or uncorrelated; in many cases, noise values at different pixels are modeled as being independent and identically distributed, and hence uncorrelated.

### 3.1.2 Removal

Tradeoffs

In selecting a noise reduction algorithm, one must weigh several factors:

- the available compute power and time available: a digital camera must apply noise reduction in a fraction of a second using a tiny onboard CPU, while a desktop computer has much more power and time
- whether sacrificing some real detail is acceptable if it allows more noise to be removed (how aggressively to decide whether variations in the image are noise or not)
- the characteristics of the noise and the detail in the image, to better make those decisions

### 3.1.3 Chroma and luminance noise separation

In real-world photographs, the highest spatial-frequency detail consists mostly of variations in brightness ("luminance detail") rather than variations in hue ("chroma detail"). Since any noise reduction algorithm should attempt to remove noise without sacrificing real detail from the scene photographed, one risks a greater loss of detail from luminance noise reduction than chroma noise reduction simply because most scenes have little high frequency chroma detail to begin with. In addition, most people find chroma noise in images more objectionable than luminance noise; the colored blobs are considered "digital-looking" and unnatural, compared to the grainy appearance of luminance noise that some compare to film grain. For these two reasons, most photographic noise reduction algorithms split the image detail into chroma and luminance components and apply more noise reduction to the former; the in-camera noise reduction algorithm used by Nikon's DSLR's, in particular, is known for this.[1]

Most dedicated noise-reduction computer software allows the user to control chroma and luminance noise reduction separately.

### 3.1.4 Linear smoothing filters

One method to remove noise is by convolving the original image with a mask that represents a low-pass filter or smoothing operation. For example, the Gaussian mask comprises elements determined by a Gaussian function. This convolution brings the value of each pixel into closer harmony with the values of its neighbors. In general, a smoothing filter sets each pixel to the average value, or a weighted average, of itself and its nearby neighbors; the Gaussian filter is just one possible set of weights.

Smoothing filters tend to blur an image, because pixel intensity values that are significantly higher or lower than the surrounding neighborhood would "smear" across the area. Because of this blurring, linear filters are seldom used in practice for noise reduction; they are, however, often used as the basis for nonlinear noise reduction filters.

### 3.1.5 Anisotropic diffusion

Another method for removing noise is to evolve the image under a smoothing partial differential equation similar to the heat equation which is called anisotropic diffusion. With a spatially constant diffusion coefficient, this is equivalent to the heat equation or linear Gaussian filtering, but with a diffusion coefficient designed to detect edges, the noise can be removed without blurring the edges of the image.

### 3.1.6 Nonlinear filters

A median filter is an example of a non-linear filter and, if properly designed, is very good at preserving image detail. To run a median filter:

1. consider each pixel in the image
2. sort the neighbouring pixels into order based upon their intensities
3. replace the original value of the pixel with the median value from the list

A median filter is a rank-selection (RS) filter, a particularly harsh member of the family of rank-conditioned rank-selection (RCRS) filters;[2] a much milder member of that family, for example one that selects the closest of the neighboring values when a pixel's value is external in its neighborhood, and leaves it unchanged otherwise, is sometimes preferred, especially in photographic applications.

Median and other RCRS filters are good at removing salt and pepper noise from an image, and also cause relatively little blurring of edges, and hence are often used in computer vision applications.

## 3.2 Image Processing Tools

To acquire the required results the images must be processed with the help of some tools. MATLAB is a computing software which contains the necessary tools to process the image. The following are some of the tools which can be used for processing the images.

### 3.2.1 IMOPEN

This function is used to morphologically opens the image

Syntax

IM2 = imopen(IM,SE)

IM2 = imopen(IM,NHOOD)

Description

IM2 = imopen(IM,SE) performs morphological opening on the grayscale or binary image IM with the structuring element SE. The argument SE must be a single structuring element object, as opposed to an array of objects. The morphological open operation is an erosion followed by a dilation, using the same structuring element for both operations.

IM2 = imopen(IM,NHOOD) performs opening with the structuring element strel(NHOOD), where NHOOD is an array of 0's and 1's that specifies the structuring element neighborhood.

Examples

Remove the smaller objects in an image.

1. Read the image into the MATLAB® workspace and display it.

```
I = imread('snowflakes.png');

imshow(I)
```

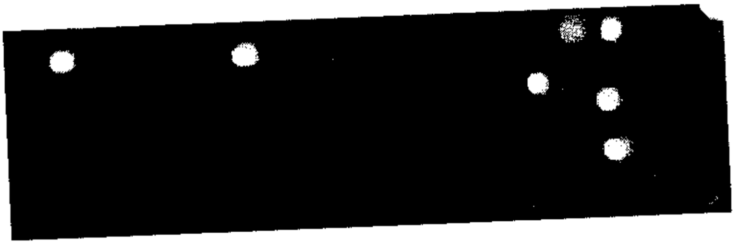
2.Create a disk-shaped structuring element with a radius of 5 pixels.

```
se = strel('disk',5);
```

3.Remove snowflakes having a radius less than 5 pixels by opening it with the disk-shaped structuring element created in step 2.

```
I_opened = imopen(I,se);

figure, imshow(I_opened,[])
```



## 3.2.2 IMFILL

Fill image regions and holes

Syntax

BW2 = imfill(BW)

[BW2,locations] = imfill(BW)

BW2 = imfill(BW,locations)

BW2 = imfill(BW,'holes')

I2 = imfill(I)

BW2 = imfill(BW,locations,conn)

Description

BW2 = imfill(BW) displays the binary image BW on the screen and lets you define the region to fill by selecting points interactively on using the mouse. To use this interactive syntax,BW must be a 2-D image. Press Backspace or Delete to remove the previously selected point. A shift-click, right-click, or double-click selects a final point and starts the fill operation. Pressing Return finishes the selection without adding a point.

[BW2,locations] = imfill(BW) returns the locations of points selected interactively in locations. locations is a vector of linear indices into the input image. To use this interactive syntax,BW must be a 2-D image.

BW2 = imfill(BW,locations) performs a flood-fill operation on background pixels of the binary image BW, starting from the points specified in locations. If locations is a P-by-1 vector, it contains the linear indices of the starting locations. If locations is a P-by-ndims(BW) matrix, each row contains the array indices of one of the starting locations.

BW2 = imfill(BW,'holes') fills holes in the binary image BW. A hole is a set of background pixels that cannot be reached by filling in the background from the edge of the image.

I2 = imfill(I) fills holes in the grayscale image I. In this syntax, a hole is defined as an area of dark pixels surrounded by lighter pixels.

BW2 = imfill(BW,locations,conn) fills the area defined by locations, where conn specifies the connectivity. conn can have any of the following scalar values.

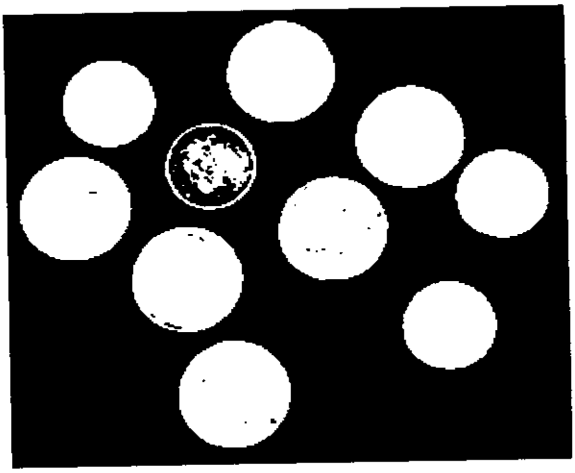
Example

Fill in the holes of a binary image.

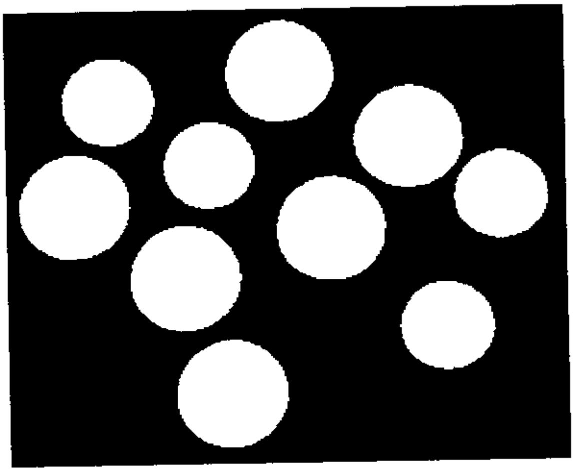
```
BW4 = im2bw(imread('coins.png'));

BW5 = imfill(BW4,'holes');

imshow(BW4), figure, imshow(BW5)
```



Original Image



Filled Image

### 3.2.3 IMSUBTRACT

Subtract one image from another or subtract constant from image

Syntax

Z = imsubtract(X,Y)

Description

Z = imsubtract(X,Y) subtracts each element in array Y from the corresponding element in array X and returns the difference in the corresponding element of the output array Z. X and Y are real, nonsparse numeric arrays of the same size and class, or Y is a double scalar. The array returned, Z, has the same size and class as X unless X is logical, in which case Z is double.

If X is an integer array, elements of the output that exceed the range of the integer type are truncated, and fractional values are rounded.

### 3.2.4 IM2BW

Convert image to binary image, based on threshold

Syntax

BW = im2bw(I, level)

BW = im2bw(X, map, level)

BW = im2bw(RGB, level)

Description

BW = im2bw(I, level) converts the grayscale image I to a binary image. The output image 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). You specify level in the range [0,1], regardless of the class of the input image. The function graythresh can be used to compute the level argument automatically. If you do not specify level, im2bw uses the value 0.5.

BW = im2bw(X, map, level) converts the indexed image X with colormap map to a binary image.

BW = im2bw(RGB, level) converts the truecolor image RGB to a binary image.

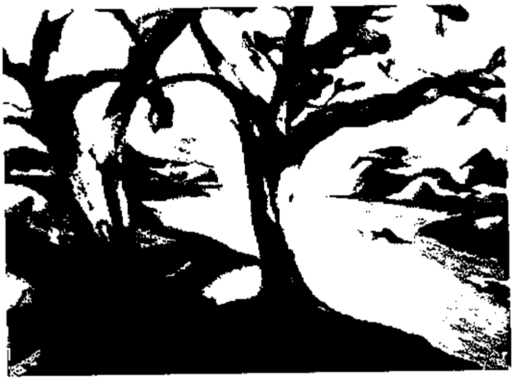
If the input image is not a grayscale image, im2bw converts the input image to grayscale, and then converts this grayscale image to binary by thresholding.

Example

```
load trees

BW = im2bw(X,map,0.4);

imshow(X,map), figure, imshow(BW)
```



### 3.2.5 BWLABEL

Label connected components in binary image

Syntax

L = bwlabel(BW,n)

[L,num] = bwlabel(BW,n)

Description

L = bwlabel(BW,n) returns a matrix L, of the same size as BW, containing labels for the connected objects in BW. n can have a value of either 4 or 8, where 4 specifies 4-connected objects and 8 specifies 8-connected objects; if the argument is omitted, it defaults to 8.

The elements of L are integer values greater than or equal to 0. The pixels labeled 0 are the background. The pixels labeled 1 make up one object, the pixels labeled 2 make up a second object, and so on.

[L,num] = bwlabel(BW,n) returns in num the number of connected objects found in BW.

### 3.2.6 REGIONPROPS

Measure properties of image regions (blob analysis)

Syntax

STATS = regionprops(L, properties)

STATS = regionprops(L, I,properties)

Description

STATS = regionprops(L, properties) measures a set of properties for each labeled region L. L can be a label matrix or a multidimensional array. WhenL is a label matrix, positive integer elements of L correspond to different regions. For example, the set of elements of L equal to 1 corresponds to region 1; the set of elements of L equal to 2 corresponds to region 2; and so on. The return value STATS is a structure array of length max(L(:)). The fields of the structure array denote different measurements for each region, as specified by properties. See Properties for a list of valid property strings.

STATS = regionprops(L, I,properties) measures a set of properties for each labeled region in the 2-D or N-D grayscale image I. L is a label matrix that identifies the regions in I and must have the same size as I.

Properties

properties can be a comma-separated list of strings, a cell array containing strings, the single string 'all', or the string 'basic'. If properties is the string 'all', regionprops computes all the shape measurements, listed in Shape Measurements. If called with a grayscale image, regionprops also returns the pixel value measurements, listed in Pixel Value Measurements. If properties is not specified or if it is the string 'basic', regionprops computes only the 'Area', 'Centroid', and 'BoundingBox' measurements. The following properties can be calculated on N-D label matrices: 'Area', 'BoundingBox','Centroid', 'FilledArea', 'FilledImage', 'Image','PixelIdxList', 'PixelList', and 'SubarrayIdx'.

## 3.3 GUIDE Tools Summary

The GUIDE tools are available from the Layout Editor shown in the figure below. The tools are called out in the figure and described briefly below. Subsequent sections show you how to use them.

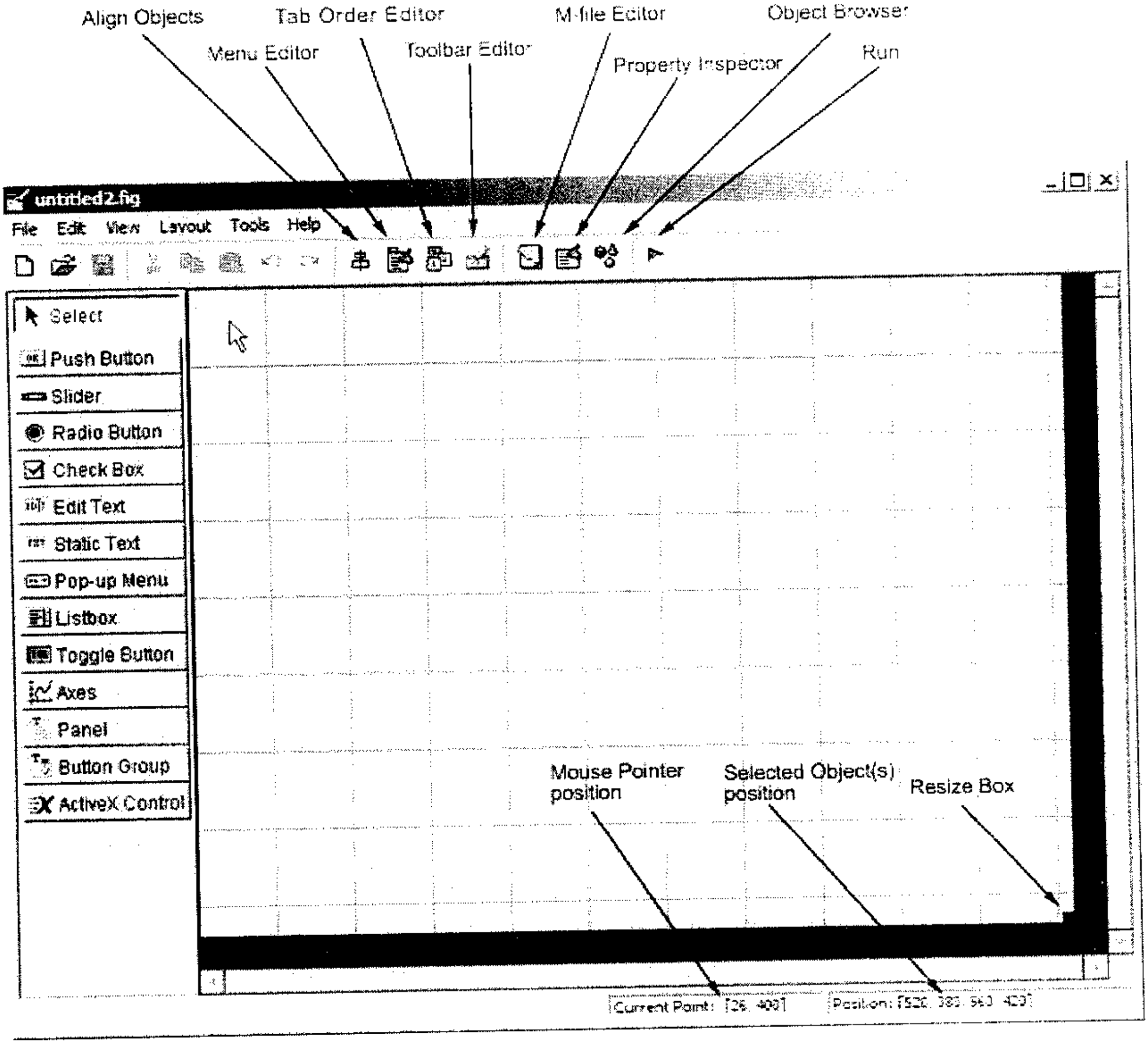


**Fig 3.3 GUIGE Tools Summary**

## TABLE 3.3 Functions of GUIDE Tools

| TOOL | FUNCTION |
|---|---|
| Layout Editor | Select components from the component palette, at the left side of the Layout Editor, and arrange them in the layout area. See Adding Components to the GUI for more information. |
| Figure Resize Tab | Set the size at which the GUI is initially displayed when you run it. See Setting the GUI Size for more information. |
| Menu Editor | Create menus and context, i.e., pop-up, menus. See Creating Menus for more information. |
| Align Objects | Align and distribute groups of components. Grids and rulers also enable you to align components on a grid with an optional snap-to-grid capability. See Aligning Components for more information. |
| Tab Order Editor | Set the tab and stacking order of the components in your layout. See Setting Tab Order for more information. |
| Toolbar Editor | Create Toolbars containing predefined and custom push buttons and toggle buttons. See Creating Toolbars for more information. |
| Icon Editor | Create and modify icons for tools in a toolbar. See Creating Toolbars for more information. |
| Property Inspector | Set the properties of the components in your layout. It provides a list of all the properties you can set and displays their current values. |
| Object Browser | Display a hierarchical list of the objects in the GUI. See Viewing the Object Hierarchy for more information. |
| Run | Save and run the current GUI. See Saving and Running a GUIDE GUI for more information. |
| M-File Editor | Display, in your default editor, the M-file associated with the GUI. See GUI Files: An Overview for more information. |
| Position Readouts | Continuously display the mouse cursor position and the positions of selected objects |

# Materials and Methods

# 4. MATERIALS AND METHODS

## 4.1 MATERIALS

| NO | MATERIALS |
|---|---|
| 1. | Human Blood Samples |
| 2. | Glass slides |
| 3. | Staining Kit: Leishman's Stain and Giemsa Stain |
| 4. | Microscopic Digital Camera |
| 5. | MATLAB : Image Processing Toolbox$^{TM}$ |
| 6. | MATLAB : GUIDE$^{TM}$ |

### 4.1.1 Human Blood Samples

A blood film or peripheral blood smear is a microscope slide made from a drop of blood, that allows the cells to be examined microscopically. Blood films are usually done to investigate hematological problems (disorders of the blood) and, occasionally, to look for parasites within the blood such as malaria and filaria.

### 4.1.2 Glass Slides

- Glass slides are washed properly with water and then washed with ethanol. Then the glass slides are dried with filter paper. It should be noted that there are no dust in the slides.

### 4.1.3 Staining Kits

#### 4.1.3.1 Leishman's Stain

- Leishman's stain
- pH 6.8 phosphate buffer (Gurr's tablets, Biomedical Specialties cat. # 33199)
- Working stain: Leishman's stain diluted 1:4 with buffer
- Xylene

#### 4.1.3.2 Giemsa Stain

- Giemsa stain, working solution
  - Giemsa stock solution ------ 40 drops
  - Distilled water -------------- 40 ml

  The diluted stain keeps well, but is best made up fresh each time.

- Acetic acid 0.5%

### 4.1.4 Microscopic Digital Camera

Olympus Cx2 series biological microscope is used to view the stained cells. The microscope attachable digital camera is used to capture the image of the cells.

### 4.1.5 MATLAB Image Processing Toolbox ™

The Image Processing Toolbox™ software is a collection of functions that extend the capability of the MATLAB® numeric computing environment. The toolbox supports a wide range of image processing operations, including

- Spatial image transformations
- Morphological operations
- Neighborhood and block operations
- Linear filtering and filter design
- Transforms

- Image analysis and enhancement
- Image registration
- Deblurring
- Region of interest operations

Many of the toolbox functions are MATLAB M-files, a series of MATLAB statements that implement specialized image processing algorithms.

### 4.1.6 MATLAB GUIDE ™

GUIDE, the MATLAB® graphical user interface development environment, provides a set of tools for creating graphical user interfaces (GUIs). These tools simplify the process of laying out and programming GUIs.

Using the GUIDE Layout Editor, you can populate a GUI by clicking and dragging GUI components—such as axes, panels, buttons, text fields, sliders, and so on—into the layout area. You can also create menus and context menus for the GUI. From the Layout Editor, you can size the GUI, modify component look and feel, align components, set tab order, view a hierarchical list of the component objects, and set GUI options.

## 4.2 METHODS

### 4.2.1 Blood Smearing

Blood films are made by placing a drop of blood on one end of a slide, and using a spreader slide to disperse the blood over the slide's length. The aim is to get a region where the cells are spaced far enough apart to be counted and differentiated.

The slide is left to air dry, after which the blood is fixed to the slide by immersing it briefly in methanol. The fixative is essential for good staining and presentation of cellular detail. After fixation, the slide is stained to distinguish the cells from each other.

### 4.2.2 Staining

#### 4.2.2.1 Leishman's Stain

The following is the procedure to stain the red blood cells using Leishman's stain

- Stain slides 3 to 5 minutes in working stain.
- Rinse well in buffer. If stain is too intense, wash longer in buffer. If it is too weak, restain the slides.
- Blot dry with bibulous paper. Air dry completely.
- Rinse in xylene.
- Mount in neutral mounting medium.

#### 4.2.2.2 Giemsa Stain

The following is the procedure to stain the red blood cells using Giemsa stain

- Bring sections to distilled water
- Stain with diluted Giemsa's stain made up fresh (see technical point 1)
- Rinse in distilled water

- Differentiate with 0.5% aqueous acetic acid (see technical point 2)
- Dehydrate rapidly
- Clear and mount

### 4.2.3 Image Acquisition

The following is the procedure to obtain the image of the cells

- The slides are mount on the microscope after staining
- The cells are viewed at 100x magnification
- Using the attachable microscopic digital camera capture the image of the cells

### 4.2.4 Image Processing

The images acquired through the digital microscopic camera are processed using the computing language MATLAB v7. Image Processing Toolbox™ of MATLAB is used to process the images. The images must be processed with involves a lot of steps. The following are different stages in processing the images

#### 4.2.4.1 Noise Reduction

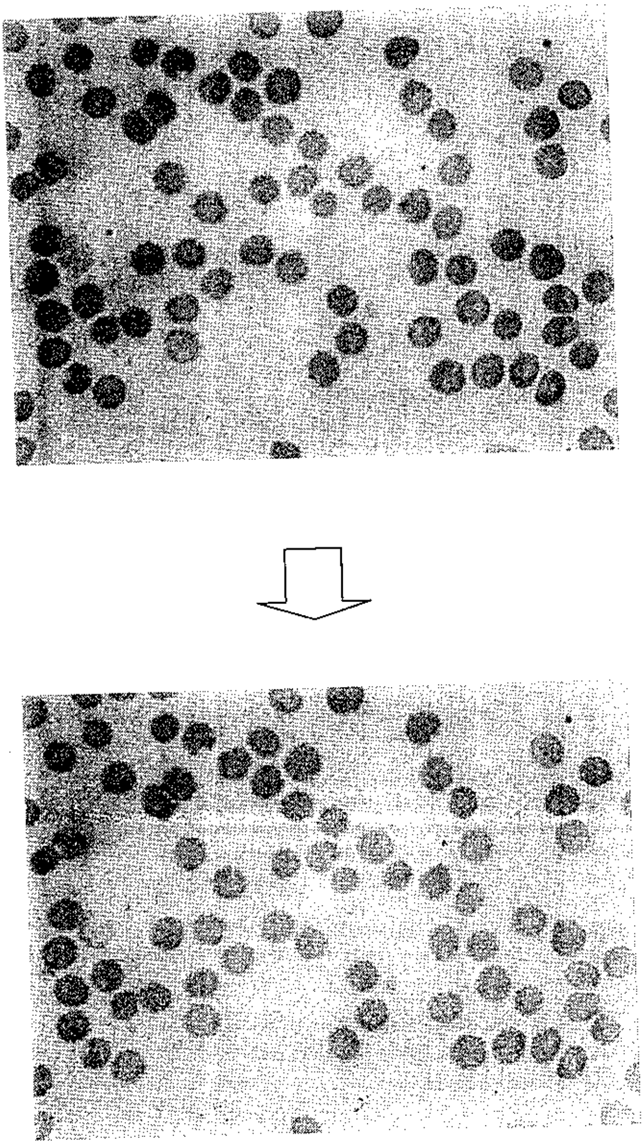
The noise in the images can be removed by performing a number of operations. These operations are some of the tools in the image processing toolbox of MATLAB. The following are operations which must be done to reduce the noise.

### 4.2.4.1.1 Conversion to gray scale

The image must be converted to a gray scale to perform morphological operations on the image. By converting the image to gray scale the image retains its luminescence.

Syntax

I = rgb2gray(RGB)

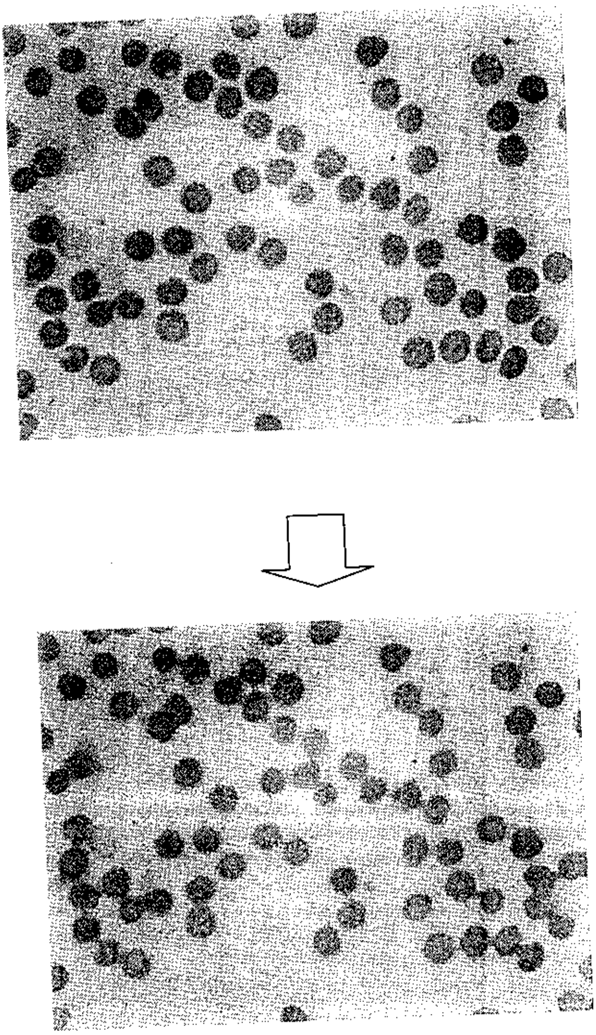


### 4.2.4.1.2 Morphologically opening the image

The next operation is morphologically opening the image using the imopen syntax. This operation opens up cells and other object which are very small in size.

Syntax

```
I2 = imopen(I,SE)
```

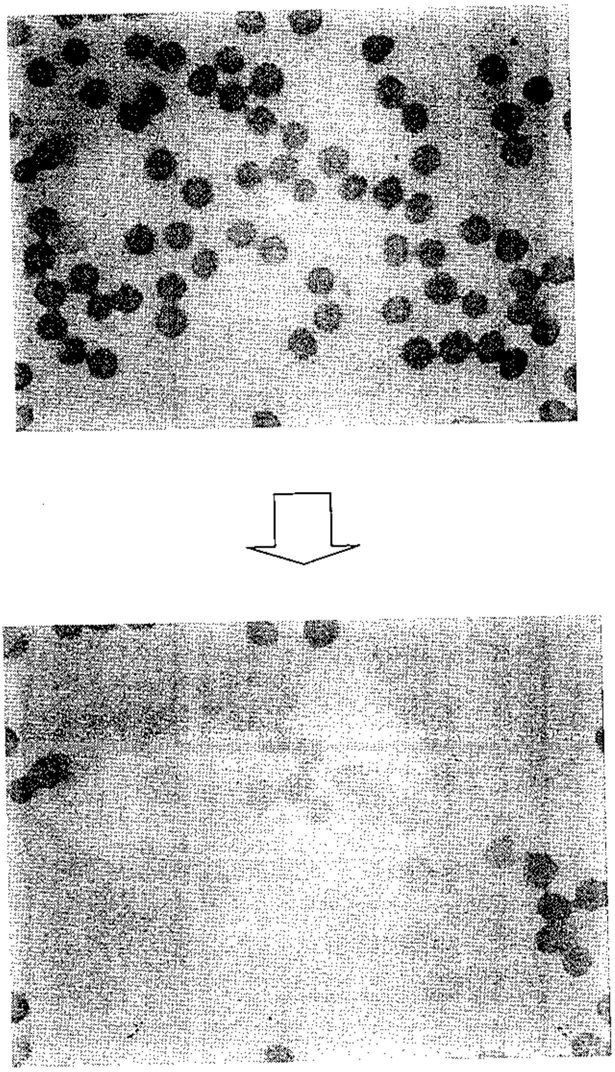


### 4.2.4.2 Image correction

Once the noise is reduced the next step is to remove the cells which are not fully covered in the picture i.e. the images which are cut at the edges of the images, as they don't count for fully captured cells.

Syntax

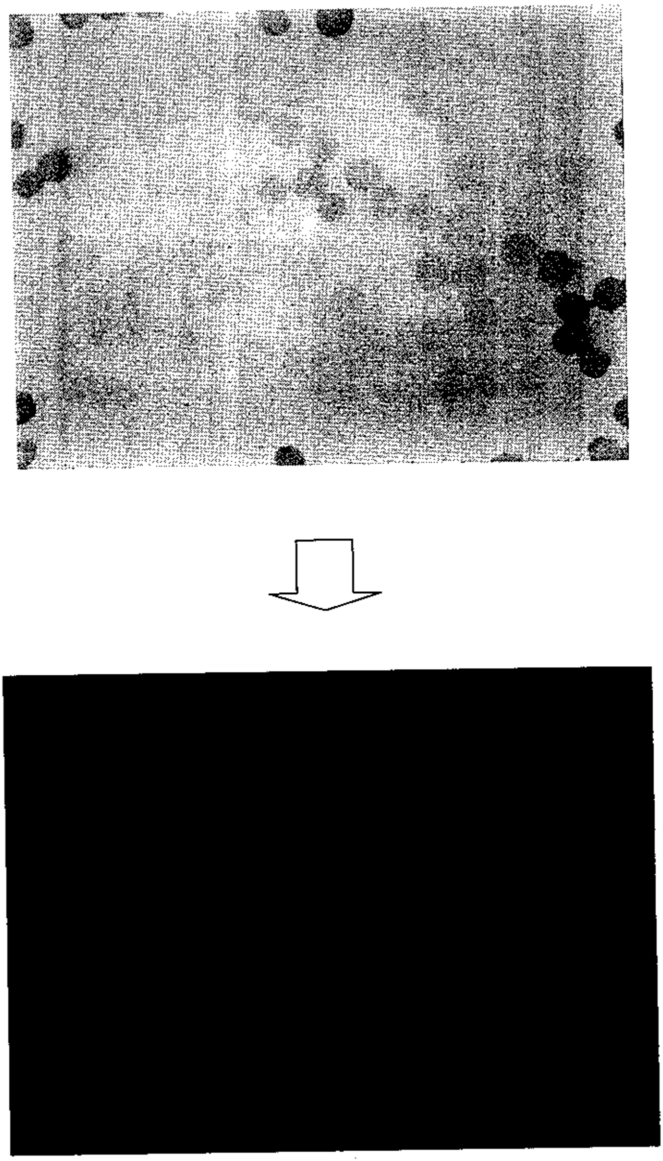
```
I3 = imfill(I2,'holes')
```



In Fig  we can see that the cells in the edges are highlighted. Now to remove these cells the image must be subtracted with the previous image.

Syntax

```
    I4=imsubtract(I3,I)
```

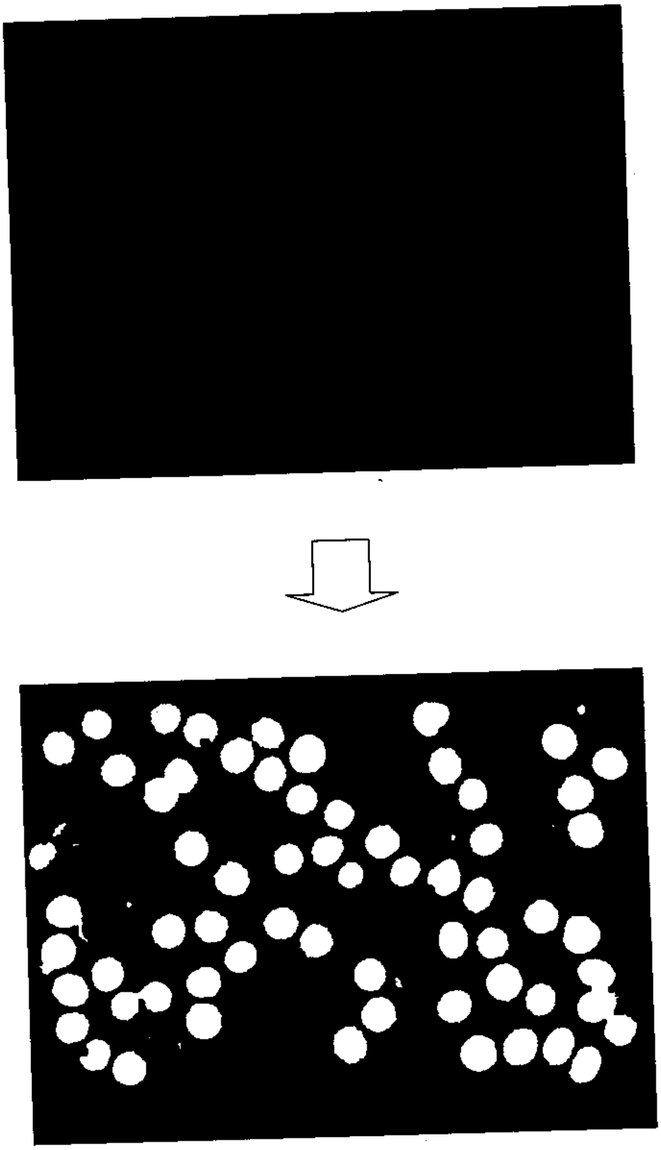


### 4.2.4.3 Conversion to Binary Image

The image is now in gray-scale format. It must be converted to binary image. Once the image is converted to binary image, the image is in digital format now. The image can be read in pixels. Thus this conversion completes the transformation of image from analog signal to digital signal.

Syntax

```
        BW = im2bw(I4, level)
```



### 4.2.4.4 Labeling the cells

Once the image is converted to binary image it has transformed from analog signal to digital signal. Now the cells in the image can be denoted in pixels. These pixels can be labeled using the following function.

Syntax

```
[L,num] = bwlabel(BW,n)
```

Labeling the image with numbers thus gives us the number of cells present in the image. But still there is a possibility of counting the noise along with the cells. Therefore further operations must be carried out to eliminate the noise from the number of cells.

### 4.2.4.5 Finding the Area

After labeling the area of each of the image can be calculated in terms of pixels. This is done to find the average area of the cells and differentiate them according to their area. The following is the function used to find the area of the cells.

Syntax

```
STATS = regionprops(L, 'Area')
```

Thus the area of all the cells will be calculated and stored in STATS. STATS is a structure variable which stores the area of all the cells in an array. The following is the method to access the values.

```
area=STATS(Area)
```

### 4.2.4 Code for Image Processing

```
global image
x1 = image;
i=rgb2gray(x1);
i2=imopen(i,strel('disk',0));
i3=imfill(i2,'holes');
i4=imsubtract(i3,i);
level=graythresh(i4);
bw=im2bw(i4,level);
fill=imfill(bw,'holes');
[lab,num]=bwlabel(fill,4);
stats=regionprops(lab,'Area');
area=[stats.Area];
minarea=mean(area)-0.25*std(area);
maxarea=mean(area)+0.25*std(area)+1000;
noofcells=find(area>=minarea&area<=maxarea);
meanarea=mean(area);
```

### 4.2.5 GUI Development

The code is written for processing the image of the red blood cells. Now the code must be presented in way such that it can be used as an application. So it can be converted into an application by developing a graphic user interface. Then the application can be reused by converting it to an executable file. The following are the steps to be followed to develop a graphic user interface.

1. MATLAB GUIDE is used to develop the graphic user interface. The GUIDE is opened from the shortcut present in the tool box. A window opens from which a blank GUI is opened.

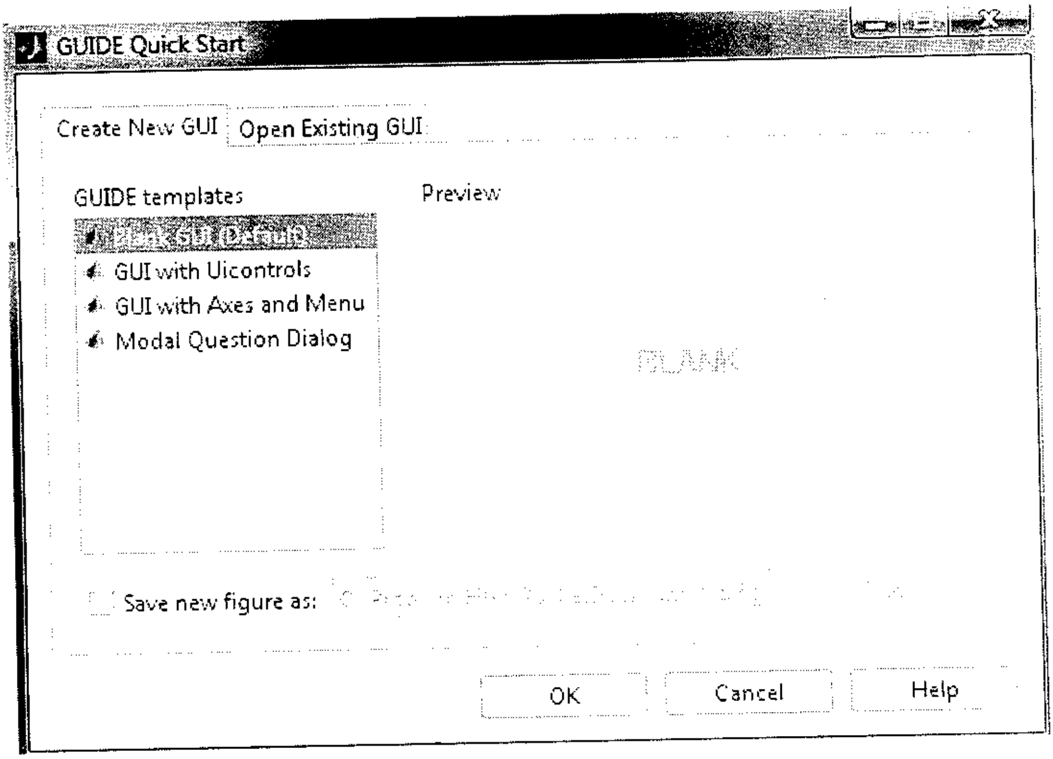


Fig 4.2.5.1 GUIDE Quick Start

2) Now a window names 'untitled1.fig' is opened. This window contains tools which can be used to develop a graphic user interface. An area is displayed where the objects can be placed. The size of the area can be adjusted accordingly.

We can even write code in the editor window to add objects to the interface. But adding code is a time consuming process and this process saves time and user friendly.

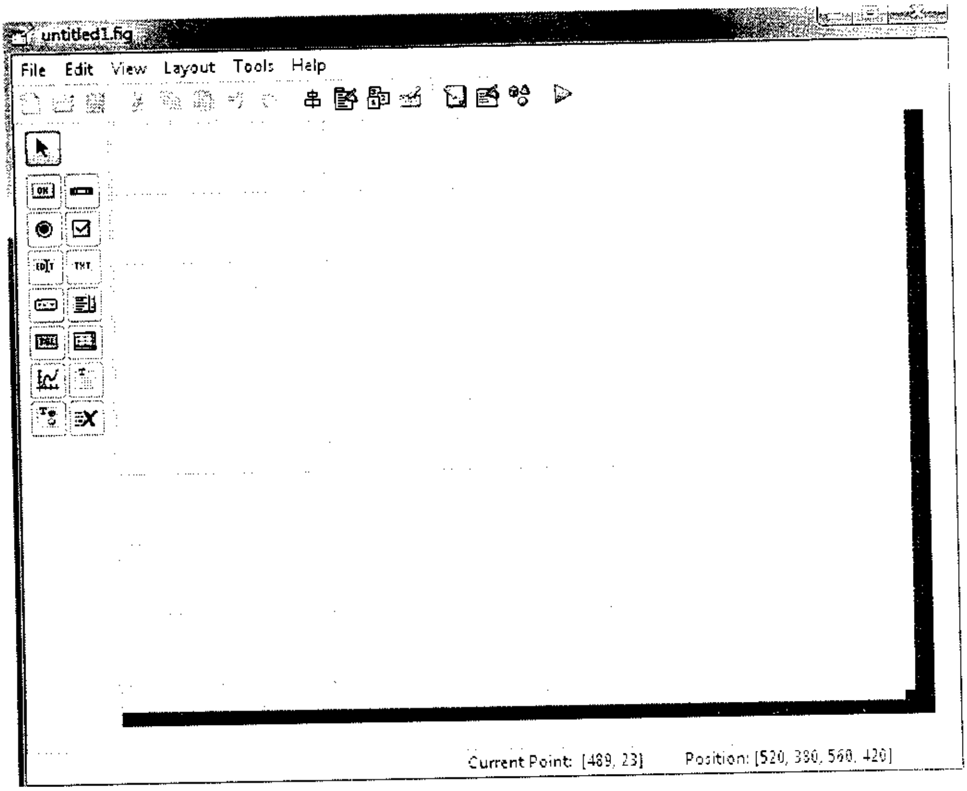
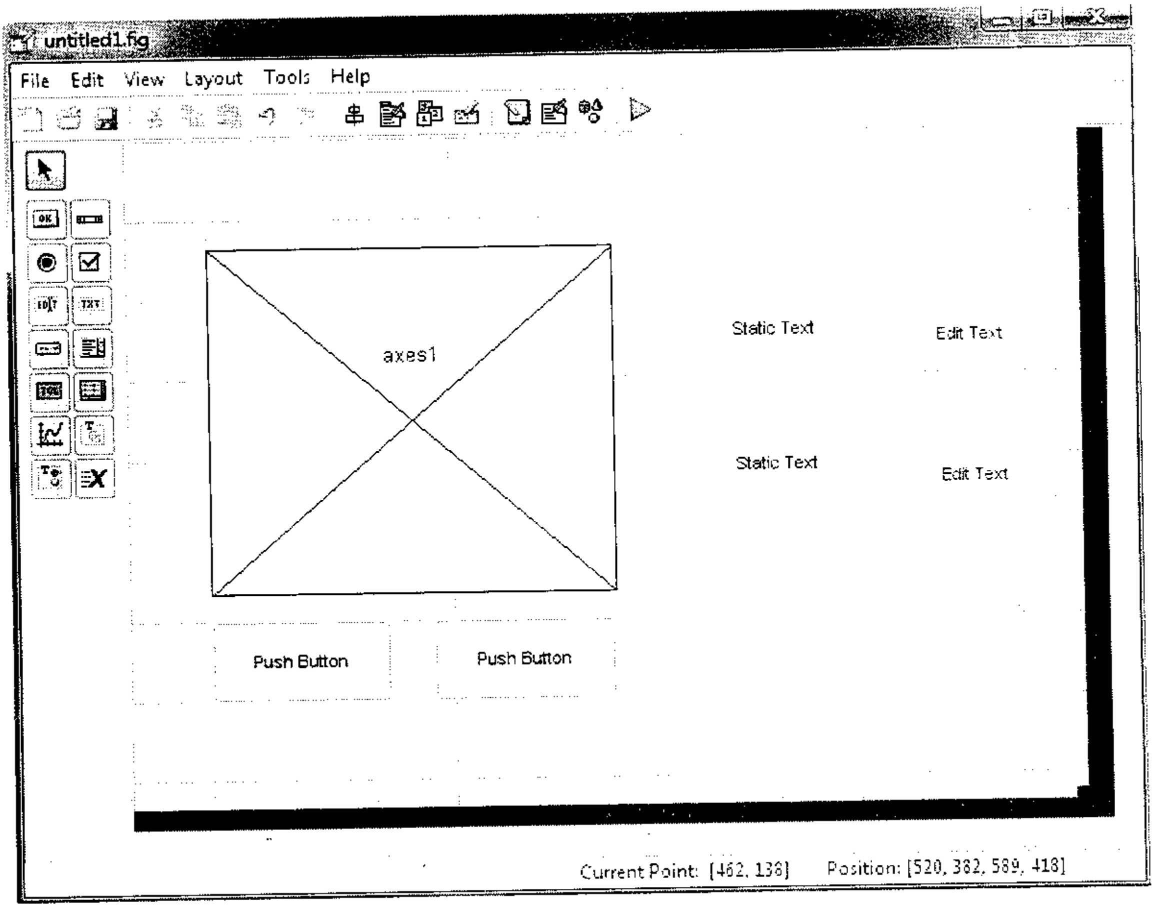


Fig 4.2.5.2 Blank GUI

3) The following image displays the GUI window containing

- 1 axes
- 2 push button
- 2 static text
- 2 edit text

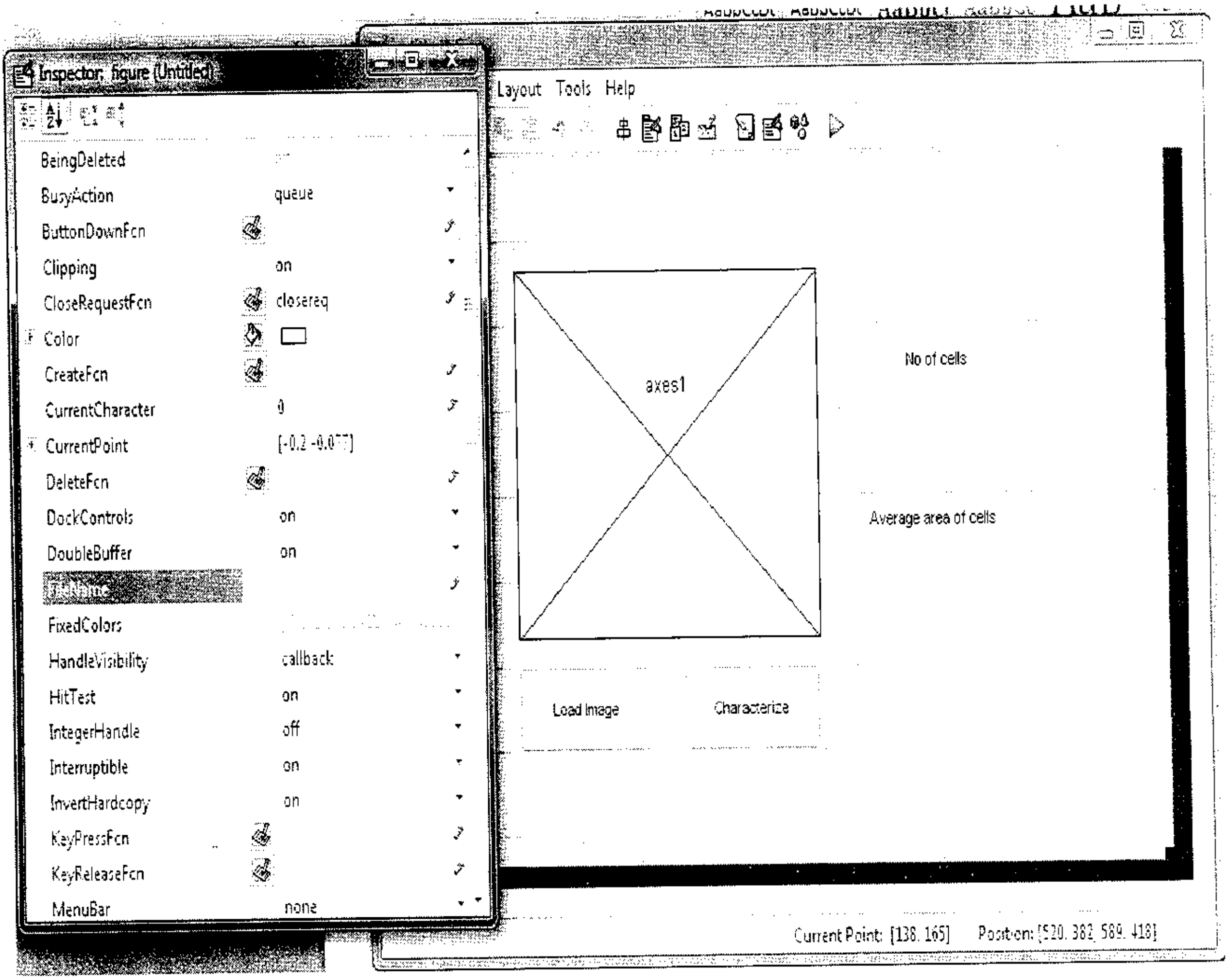


Current Point: [462, 138] Position: [520, 382, 589, 418]

Fig 4.2.5.3 Inserting Objects

4) Now the objects must be renamed according to their functions. The push buttons are used to call a function, while axes is used to display the image. One static text displays 'No of cells', while the other displays 'Average area of cells'. The edit text will be empty. The push button will display 'Load Image' and 'Characterize' respectively.

These properties are changed by opening the Inspector window and editing the properties of each of the object.



Inspector: figure (Untitled)

| | |
|---|---|
| BeingDeleted | |
| BusyAction | queue |
| ButtonDownFcn | |
| Clipping | on |
| CloseRequestFcn | closereq |
| Color | ▭ |
| CreateFcn | |
| CurrentCharacter | 0 |
| CurrentPoint | [-0.2 -0.077] |
| DeleteFcn | |
| DockControls | on |
| DoubleBuffer | on |
| FileName | |
| FixedColors | |
| HandleVisibility | callback |
| HitTest | on |
| IntegerHandle | off |
| Interruptible | on |
| InvertHardcopy | on |
| KeyPressFcn | |
| KeyReleaseFcn | |
| MenuBar | none |

Current Point: [138, 165] Position: [520, 382, 589, 418]

Fig 4.2.5.4 Changing Properties

5) Codes must be written to the push buttons so that the codes are executed when the push button is clicked.

### 4.2.5.1 Code for Load image Push button

```
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global image
[file path] = uigetfile('*.bmp;*.jpg','Select an Image:');
image = imread([path file]);
axis(handles.axes1);
imshow(image);
```

### 4.2.5.2 Code for Characterize Push button

```
function pushbutton2_Callback(hObject, eventdata, handles)
global image
x1 = image;
i=rgb2gray(x1);
i2=imopen(i,strel('disk',0));
i3=imfill(i2,'holes');
i4=imsubtract(i3,i);
level=graythresh(i4);
bw=im2bw(i4,level);
fill=imfill(bw,'holes');
[lab,num]=bwlabel(fill,4);
stats=regionprops(lab,'Area');
area=[stats.Area];
minarea=mean(area)-0.25*std(area);
maxarea=mean(area)+0.25*std(area)+1000;
noofcells=find(area>=minarea&area<=maxarea);
meanarea=mean(area);
set(handles.edit1,'string',length(noofcells));
set(handles.edit2,'string',[num2str(meanarea) ' pixels']);
```

6) After the code is written it must be run to find any errors in the program. If any error is displayed the code must be verified and debugged for errors.

7) Once the codes are verified and the program is free of errors it must be saved. After saving, the code must be converted to an executable file. The following is the syntax used to convert the program into an executable file.

Syntax

```
        mcc –m filename.m
```

This command will create an filename.exe file which is an application of the above program.

# Results and Discussion

# 5. RESULTS AND DISCUSSION

## 5.1 Staining

The blood samples were taken from different sources and the blood smears were prepared. Fine blood smears were prepared by taking the blood drops in one slide and using another slide the blood is smeared by keeping the slide at 45° and dispersing the blood finely on the slide.

After the blood smear is prepared, the staining is carried out. One set of blood smear is stained with Giemsa stain and the other stain is Leishman's stain. The protocol is followed to attained the required results.

Once the slides are stained, its marked and mounted on the optical biological microscope at 100x magnification. Oil must be applied to attain better results. The cells are viewed and it has been observed that the cells are stained prominently by Giemsa stain, while the cells are not clearly defined in Lieshman's stain.

When viewed under the microscope we can observe that in some areas the cells are overlapped. This is due to the improper smearing of the blood over the slide. These areas must be avoided. In other areas the cells will be clearly dispersed. In those areas the cells will be clearly defined. So these cells must be captured by the digital camera.

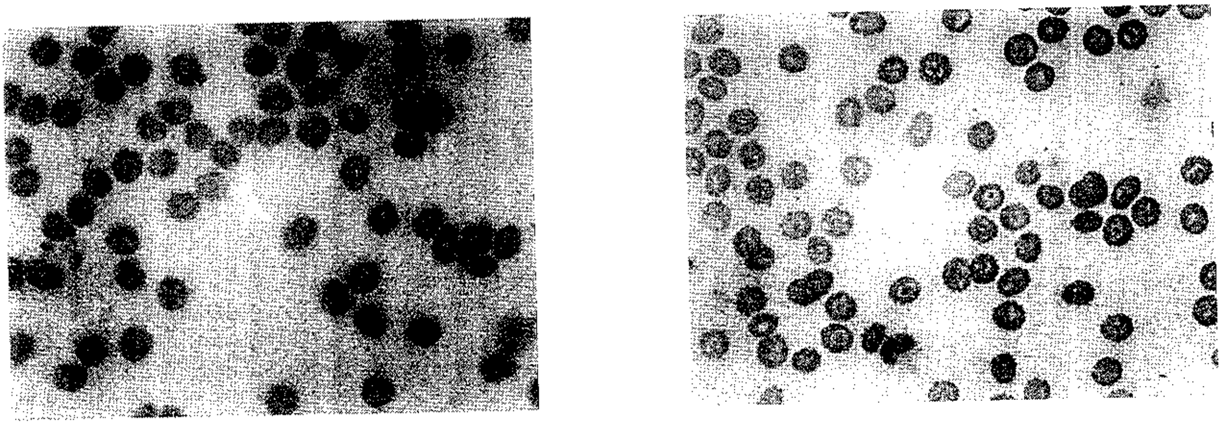


Fig 5.1.1 Giemsa Stain

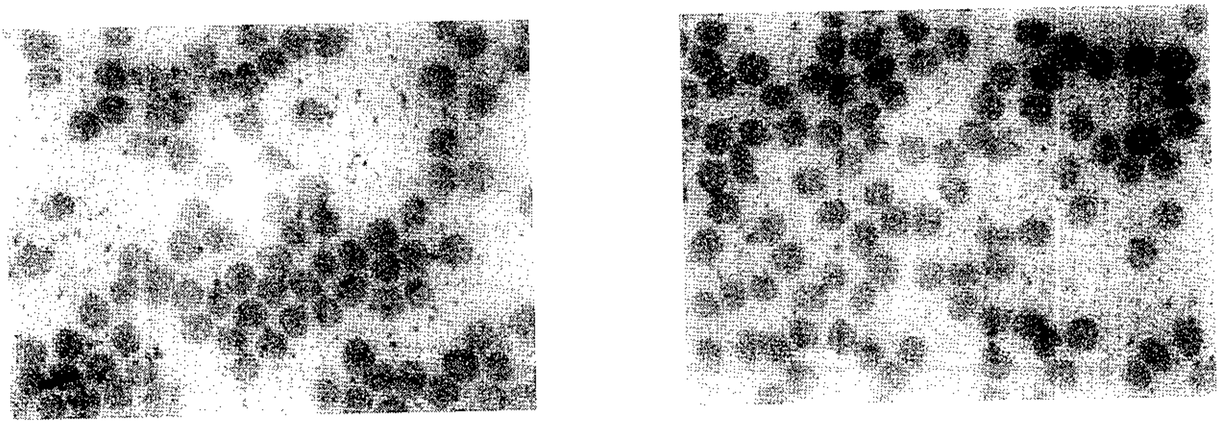
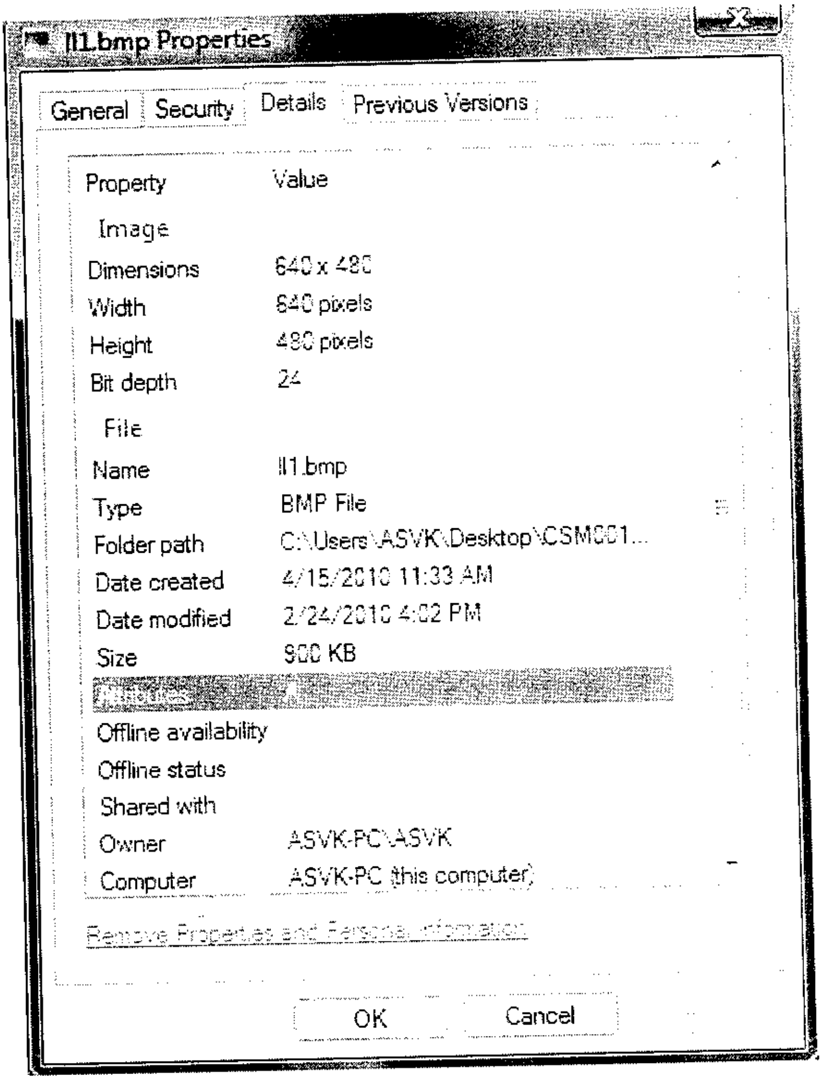


Fig 5.1.2 Leishman's stain

## 5.2 Image Acquisition

After the clear focusing of the cells, the image is captured using the microscopic digital camera. The images are stored in the computer under appropriate names. The images are stored as a bitmap file. All the images are organized and stored so that it will be easy to load the image for processing.



| Property | Value |
|---|---|
| Image | |
| Dimensions | 640 x 480 |
| Width | 640 pixels |
| Height | 480 pixels |
| Bit depth | 24 |
| File | |
| Name | ll1.bmp |
| Type | BMP File |
| Folder path | C:\Users\ASVK\Desktop\CSM001... |
| Date created | 4/15/2010 11:33 AM |
| Date modified | 2/24/2010 4:02 PM |
| Size | 900 KB |
| Attributes | |
| Offline availability | |
| Offline status | |
| Shared with | |
| Owner | ASVK-PC\ASVK |
| Computer | ASVK-PC (this computer) |

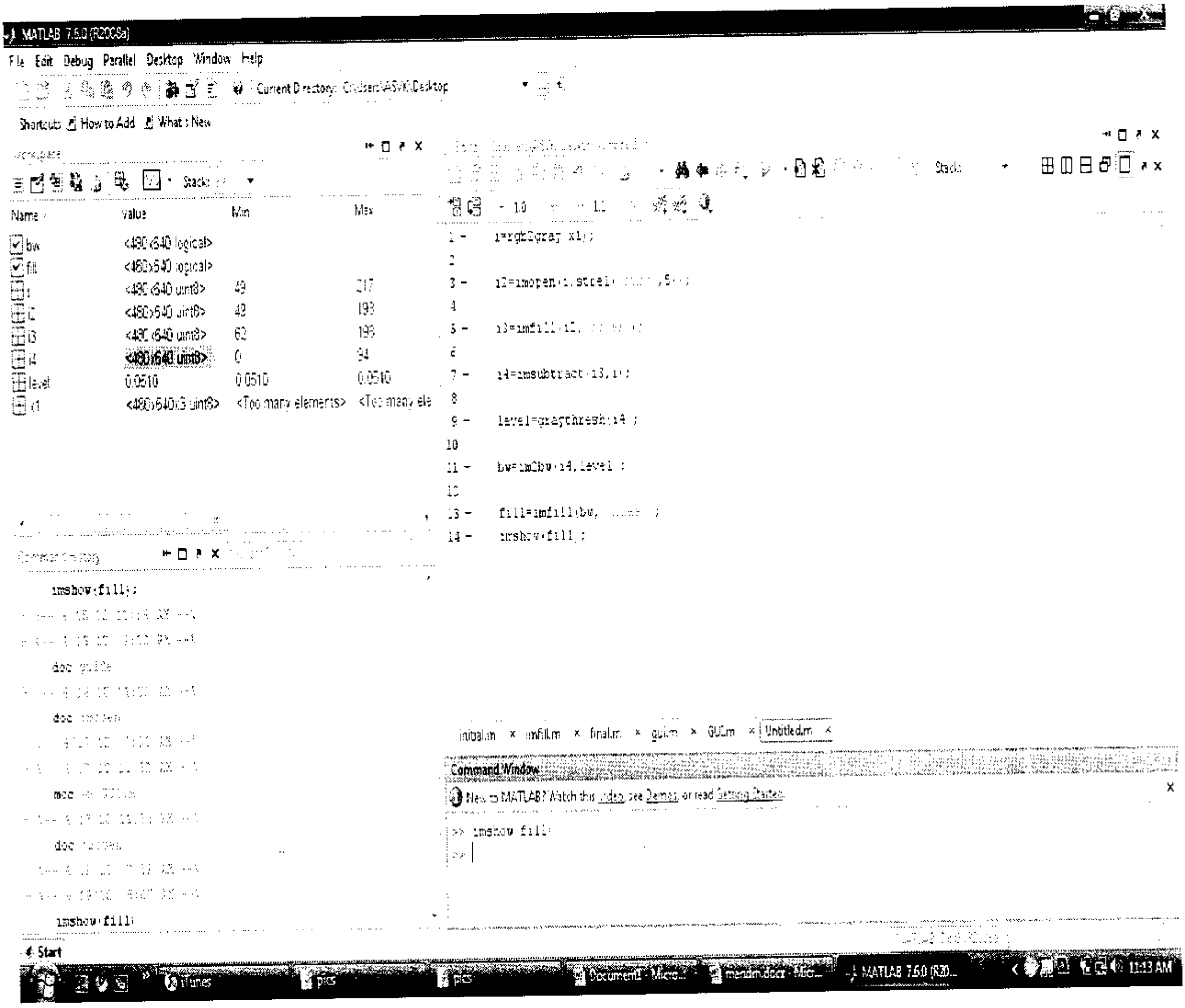
Remove Properties and Personal Information

**Fig 5.2 Properties of image**

## 5.3 Image Processing

The images are then processed in MATLAB v7 using the Image processing toolbox. The codes are written in the Editor window of the MATLAB screen. The codes can be written and edited in this window. All the files and variables created during coding are found in the workspace window along with their properties.

The command window allows us to enter command and execute it one by one. The command history window displays all recent commands entered in the command window along with the date and time.



MATLAB 7.5.0 (R2008a)
File Edit Debug Parallel Desktop Window Help
Current Directory: C:\Users\ASVK\Desktop
Shortcuts How to Add What's New
Stack:
initial.m × imfill.m × final.m × gui.m × GUI.m × Untitled.m ×
Command Window
New to MATLAB? Watch this Video, see Demos, or read Getting Started.
Start
iTunes
pics
pics
Document1 - Micro...
mendmi.docx - Mic...
MATLAB 7.5.0 (R20...
11:13 AM

| Name | Value | Min | Max |
|---|---|---|---|
| ☑ bw | <480x640 logical> | | |
| ☑ fill | <480x640 logical> | | |
| i | <480x640 uint8> | 49 | 217 |
| i2 | <480x640 uint8> | 49 | 193 |
| i3 | <480x640 uint8> | 62 | 193 |
| i4 | <480x640 uint8> | 0 | 94 |
| level | 0.0510 | 0.0510 | 0.0510 |
| x1 | <480x640x3 uint8> | <Too many elements> | <Too many ele |

```
i=rgb2gray(x1);

i2=imopen(i,strel([illegible],5));

i3=imfill(i2,[illegible]);

i4=imsubtract(i3,i);

level=graythresh(i4);

bw=im2bw(i4,level);

fill=imfill(bw,[illegible]);
imshow(fill);
```

```
imshow(fill);
doc guide
imshow(fill)
```

```
>> imshow(fill)
>> 
```

**Fig 5.3.1 MATLAB window**

The codes written are executed and the final image is acquired. The final image is a binary image, which is a digital image. These images are read in the form of pixels.

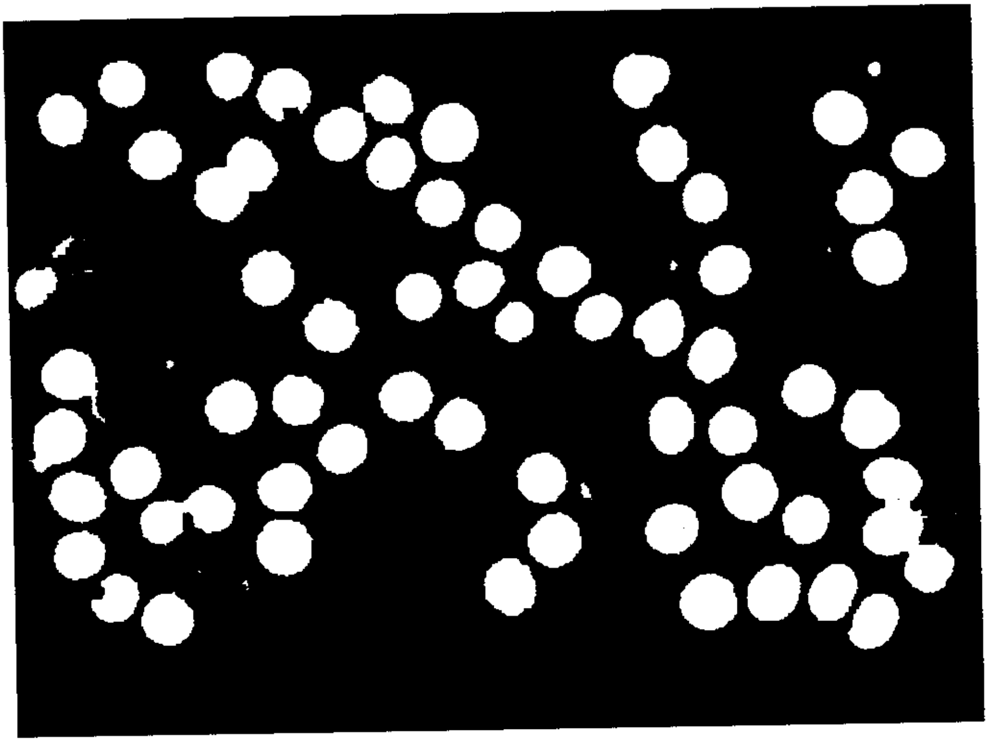
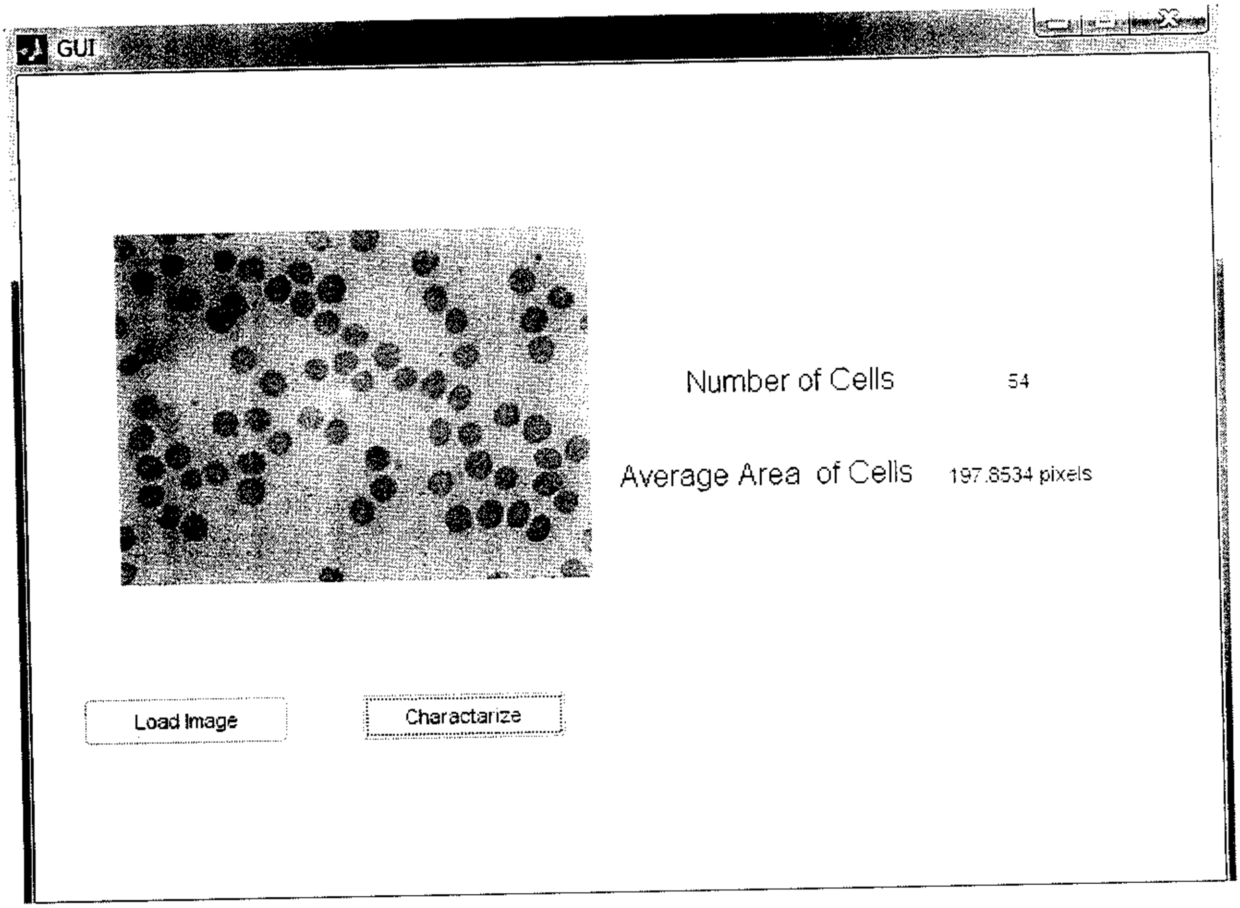


**Fig 5.3.2 Processed Image**

The image contains the cells and some noise which was not removed. Some cells are not clearly defined due to the overlapping of the cells. The background has a pixel value 0 and the cells have a pixel value 1.

## 5.4 Graphic User Interface

The graphic user interface was developed for the program using the MATLAB GUIDE. The use interface contains an area to display the image. Two text box in which the calculated results will be displayed. Two push buttons, one is used to load the image and the other is used to characterize the red blood cells.



Number of Cells 54

Average Area of Cells 197.8534 pixels

**Fig 5.4 Graphic User Interface**

Further objects can be added to the graphic user interface to enhance the application. Image recognition can be done to find the type of cells present in the image. Code can be written to find the area of individual cells by clicking on the cells.

# *Conclusion*

# 6. CONCLUSION

The present technique was carried out to characterize the digital image of the red blood corpuscles.

- Giemsa and Leishman staining was carried out on the blood samples smeared on glass slides.
- The images of the red blood cells were then acquired using the microscopic attachable digital camera.
- Then image processing was carried out on the image to find the number of cells in the image and the average area of the cells.
- MATLAB GUIDE was used to develop a graphic user interface for the program developed to characterize the red blood cells

# References

1. Tunali, S., Cabuk, A. And Akar, T. (2006) “Removal of lead and copper ions from aqueous solutions by bacterial strain isolated from soil” *Chem. Eng. J.,* 115, pp.203–211.

2. Weston, A. and Murthy, M. (1996). Synthesis of Fullerenes: “An Effort to Optimize Process Parameters” Carbon, 34, pp. 1267-1274.

3. Guo X.Y., Zhang A.Z and Shan X.Q. (2008). “Adsorption of metal ions on lignin”, *J. Hazard. Mater.* 151, pp. 134-142.

4. Gulliver G. (1875). "On the size and shape of red corpuscles of the blood of vertebrates, with drawings of them to a uniform scale, and extended and revised tables of measurements". *Proceedings of the Royal Society of London* **1875**: 474–495.
5. Snyder, Gregory K.; Sheafor Brandon A. (1999). "Red Blood Cells: Centerpiece in the Evolution of the Vertebrate Circulatory System". *Integrative and Comparative Biology* **39**: 189.
6. Diesen D.L, Hess D.T, Stamler J.S (2008). "Hypoxic vasodilation by red blood cells: evidence for an s-nitrosothiol-based signal". *Circulation Research* **103** (5): 545–53.
7. Kleinbongard P, Schutz R, Rassaf T, et al (2006). "Red blood cells express a functional endothelial nitric oxide synthase". *Blood* **107**: 2943–51.
8. Gustashaw K.M. (1991) Chromosome Stains. In “The ACT Cytogenetics Laboratory Manual”, Second Edition, edited by M. J. Barch. The Association of Cytogenetic Technologists, Raven Press, Ltd., New York,.
9. Shapiro H.M., Mandy F. (2007). "Cytometry in malaria: moving beyond GiemsaCytometry". “The journal of the International Society for Analytical Cytology”**71** (9): 643–5.

10. James D. Broesch, Dag Stranneby and William Walker. *Digital Signal Processing: Instant access*. Butterworth-Heinemann. p. 3.

11. Granger BL, Green SA, Gabel CA, Howe CL, Mellman I, Helenius A: Characterization and cloning of lgp110, a lysosomal membrane glycoprotein from mouse and rat cells. J Biol Chem 265:12036–12043, 1990.

12. Haralick RM: Statistical and structural approaches to texture. Proc IEEE 67:786–804, 1979.

13. Hornick K, Stinchcombe M, White H: Multilayer feedforward networks are universal approximators. Neural Netw 2:359–366, 1989.

14. Hu M-K: Visual pattern recognition by moment invariants. IRE Trans Inf Theory IT 8:179–187, 1962.

15. Jaggi B, Poon SS, MacAulay C, Palcic B: Imaging system for morphometric assessment of absorption or fluorescence in stained cells. Cytometry 9:566–572, 1988.

16. Khotanzad A, Hong YH: Invariant image recognition by Zernike moments. IEEE Trans Pattern Anal Machine Intell 12:489–497, 1990.

17. Khotanzad A, Hong YH: Rotation invariant image recognition using features selected via a systematic method. Pattern Recognit 23:1089–1101, 1990.

18. Khotanzad A, Lu J-H: Classification of invariant image representations using a neural network. IEEE Trans Acoust Speech Signal Process 38:1028–1038, 1990.

19. Klecka WR: Discriminant analysis. In: Quantitative Applications in the Social Sciences, Vol. 19, Sullivan JL (ed). Sage University Paper, Beverly Hills and London, 1980.

20. Lee BR, Haseman DB, Reynolds CP: A digital image microscopy system for rare-event detection using fluorescent probes. Cytometry 10:256–262, 1989.

21. Linstedt AD, Hauri HP: Giantin, a novel conserved Golgi membrane protein containing a cytoplasmic domain of at least 350 kDa. Mol Biol Cell 4:679–693, 1993.

22. Lockett SJ, Jacobson K, Herman B: Quantitative precision of an automated, fluorescence-based image cytometer. Anal Quant Cytol Histol 14:187–202, 1992.