

# **Chapter 1**

## **Introduction**

Automated Sickle Cell Anaemia Detector (A.S.C.A.D.) is a software application that diagnoses ‘Sickle Cell Anaemia’ (drepanocytosis) from microscopic images of the patient’s blood smear. It uses advanced image processing techniques to analyze the shape of ‘Red Blood Cells’ which are primary indicators of the disease. Our ultimate goal however, lies in automation of the pathological examinations, thereby expediting the process of differential diagnosis.

### **1.1 Description**

A red blood cell in normal physiological condition, is circular in front view and bi-concave in side view. Sickle cell anaemia is a hereditary blood disorder which primarily presents itself with high propensity for red blood cells to assume a crescentic or sickle-like shape.

When the patient provides microscopic image of his/her blood sample to A.S.C.A.D, it uses edge detection algorithms to scan for the presence of abnormally shaped red blood cells in it. The A.S.C.A.D then proceeds to compare the ratio of normal RBC count to sickle shaped RBC count. A decision considering a threshold is then made to arrive at the conclusion to whether the patient is anaemic or not. The application also provides a detailed report of its result for further diagnostic purposes if required. It also provides appropriate recommendations based on it.

## 1.2 Problem Formulation

Sickle cell anaemia affects about 300,000 children born each year, of which, tropical regions, particularly sub-Saharan Africa, India and the Middle-East witness much higher epidemiological occurrence [1]. According to studies, about 8–22% of Indians have this disease most of which remains untreated, especially in rural backgrounds [2].

In sickle celled disease, the abnormally shaped RBCs have a lower haemoglobin function. This results in anaemia and other complications such as sickle cell crisis, haemolytic crisis, cholelithiasis, increased risk of infection, pulmonary hypertension, hyposplenism, renal failure, stroke and consequentially an increased risk of death. Many of these complications can be mitigated and prevented to a large extent with vaccination, preventative antibiotics, blood transfusion, etc. Early detection of the disease thus becomes the pivotal antecedent in not only improving the patient's longevity but also the overall health standard of our nation.

## 1.3 Motivation

Despite onus being on detection of sickle celled disease, we are yet to see an automated screening test for it. Typically for a patient admitted in a medical environment, consideration of sickle cell anaemia as the likely cause depends on his symptoms, which usually varies in case of this disease. Furthermore, its confirmed diagnosis occurs only after an extensive and manual laboratorial examination of the patient's blood. As seen above, an outward manifestation of the disease to an extent that requires hospitalization, further prolonged by manual blood tests, are often needed before we even begin treatment of this already-fatal disease. In such cases, not only that the mitigation and prevention techniques are now out of question but also going forward, the impetus shifts from a preventive care to a suppressive treatment regimen. Thus an automated screening system and such a screening healthcare programme is imperative for us. This provided the motivation for developing A.S.C.A.D which allows simple, quick and inexpensive method of screening.

## 1.4 Proposed Solution

In U.K., new born babies are screened for sickle cell anaemia manually. Such a programme is more essential in India where the prevalence of the disease is much higher. But owing to the time and expenditure required in manual blood screening of every individual, such a practice is not followed in India.

A solution to this lies in the use of image processing. From a microscopic blood smear image, one can filter out all the components in the human blood other than the RBCs and determine a relative numeric value for each RBC that will help us in estimating its shape and decide whether it is normal or not. Thus, the software can be used to diagnose sickle cell anaemia from a microscopic image with high accuracy. A.S.C.A.D is such diagnostic software.

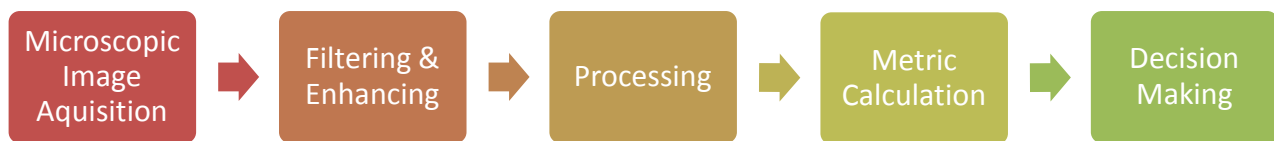


Fig 1.4.1 Proposed solution block diagram

A.S.C.A.D provides a solution that is not just fast and inexpensive, but also can be easily incorporated into routine blood-work performed at birth. It also expedites the diagnosis of sickle cell anaemia in adult patients, by circumventing the lengthy manual blood examinations. Additionally, it resolves the possibility of human-induced error that may occur during manual testing.

## **1.5 Scope of the project**

The scope of the whole project will be solely on software domain. A.S.C.A.D will directly accept microscopic image as a file input. An arrangement that allows a microscope embedded with an image capturing device to directly feed real-time images would be the ideal use of A.S.C.A.D. However, integration of this software onto a microscope will be beyond the scope of this project owing to the local unavailability of such hospital-grade microscopes and unavailability of real-time anaemic patient and his/her blood to test on. Images from the internet will be used as experimental data to test the software.

## Chapter 2

### Literature Survey

Sickle cell anaemia is a well understood medical condition. There is no widely available standard cure. However there are certain treatments to prevent the symptoms and thus avoid complications resulting from it. These treatments require early detection as mentioned above. However there isn't any automated test for sickle cell anaemia. This presumably stems from insufficiency of literature or studies involving automation of diagnostic process. Correspondingly, literature involving automated detection of sickle celled disease remains considerably scarce.

One of such recherche studies published on Science Direct is 'Detection of Abnormal Findings in Human RBC in Diagnosing Sickle Cell Anaemia Using Image Processing' authored by Pranati Rakshita and Kriti Bhowmik [3]. This study treads along parallel ideologies as our project. The proposed methodology in it involves preprocessing, edge detection and region selection.

Preprocessing requires initially converting the blood smear image into a binary form. Then an adaptive filtering method is used to eliminate unwanted noise present in it. The filter of choice used in this paper was Weiner's filer. The next step is edge detection which demarcates boundaries for the red blood cells. The study approves the use of any of the following edge detectors: Sobel Operator, Robert's Operator, Canny Operator, LoG Operator, Zerocross Operator and Prewitt Operator [4]. The third step is region selection, wherein we measure properties of connection image components which satisfy certain predefined conditions. It is used to compute the shape measurements like the centroid, area, bounding box convexHull, convexArea, perimeter etc. An extensive use of properties area and perimeter is done. Area is a scalar value which represents the actual number of pixels in the region and perimeter is used to calculate the distance around the boundary of the region. In MATLAB, these measurements can be computed using the inbuilt function 'RegionProps'.

Using the measurements obtained, the study proposes a metric to determine the circularity of objects (which are the RBCs) in the image. This metric is defined as:  $(4 \cdot \pi \cdot \text{area}) / \text{perimeter}^2$  [1]. The metric ranges from values of 0 to 1; 1 for a perfectly 2D circle, 0.785 for a 2D square and so on goes decreasing as the shape resembles less of a circle. So a typical RBC would have a metric higher than 0.82 whereas a sickle shaped RBC would have a much lower metric of about 0.4 – 0.5. This we obtain a clear distinction between different shapes of RBCs present.

The final step involves classification of the input image as anaemic or not. In a study by NIIT Rourkela, they use clustering algorithms for segmentation of images [5]. One such algorithm is K-means clustering. It is an algorithm based on finding data clusters in a data set such that a cost function (or an objection function) of dissimilarity (or distance) measure is minimized. K-means uses an iterative algorithm that minimizes the sum of distances from each object to its cluster centroid, over all clusters. This algorithm moves objects between clusters until the sum cannot be decreased further. The result is a set of clusters that are as compact and well-separated as possible.

Another approach suggested here is the use of Fuzzy C-means clustering. It is one of the commonly used methods for image segmentation and its success is mainly due to the introduction of fuzziness for the belongingness of each image pixels. Compared with crisp or hard segmentation methods, FCM is able to retain more information from the original image, with the only disadvantage being increased sensitivity to noise and other imaging artifacts. Fuzzy c-means (FCM) is a data clustering technique in which a dataset is grouped into 'n' clusters with every data point in the dataset belonging to every cluster to a certain degree. For example, a certain data point that lies close to the center of a cluster will have a high degree of belonging or membership to that cluster and another data point that lies far away from the center of a cluster will have a low degree of belonging or membership to that cluster.

# **Chapter 3**

## **System Analysis**

### **3.1 Functional Requirements**

The functional requirements of the A.S.C.A.D. can be explained by considering the following features of the system:

#### **3.1.1 Record details of the patient**

The software must initially input necessary details about the patient such as his/her name, age, name of the medical practitioner etc. These details would later be displayed on the final report as well.

#### **3.1.2 Input Image**

The system should take as input a high quality microscopic image of the blood smear of the patient in order to perform various computations and analysis on the image and to determine whether the patient is diagnosed with sickle cell anaemia or not.

#### **3.1.3 Display results**

The system, after performing all the computations and analysis on the input image, must display the results of the patient with the diagnosis and along with the details of the patient taken at the start of the process.

### **3.2 Non-functional Requirements**

#### **3.2.1 Performance Requirements**

The system must be able to seamlessly diagnose the disorder in the patient's blood sample by analyzing the image of the sample. The system must be able to adjust itself appropriately to the different colors and shades of images taken using different cameras and must also be able to input images of different dimensions.

### 3.2.2 Safety and security requirements

The access to this system must lie only with authorized pathologists and technicians (in case of the system being used at the pathology lab) or only with the patient (if used personally). This would overcome the risk of having nuisance creators or any other person from making changes to the underlying code of the system. The device on which the software is ported must be kept in a stable environment that is free of any cause of physical damage that can be caused to it.

### 3.3.3 Software quality attributes

- **Reliability :**

The system must provide genuine results about the presence of the disorder in the patient, which must be accepted by all pathologists/medical bodies. The results displayed must include accurate information about the presence of sickle shaped cells.

- **Availability :**

This software must be available to the concerned officials/people at all times, whenever needed. The software does not include any login requirements and hence can be used immediately without any authorization if one has access to it. The use and access to this system must be confined by the organization/person by physical means.

- **Maintainability :**

The software would not require any maintenance explicitly. However the hardware over which this software would be installed would require frequent maintenance checks. The only change the software would probably undergo would be in case of any update to the current installed version being released by the proprietors of the software.

- **Portability :**

The software must be able to run on all standard devices across all possible operating systems. The end user must not face any difficulty or must not require any technical assistance in porting the software to another device.



### **3.3 System Requirements**

#### **3.3.1 Hardware Requirements**

For desktop PC, a mid-range specification, good enough to run MATLAB would suffice. Embedded systems will have much lower hardware requirements and would be dependent on features implemented.

A typical desktop PC would optimally run the program with the following specifications.

- ~2.0 Ghz Multi-core Processor or 2.6 Ghz Single core
- 1.5 GB or more RAM
- 2-3 GB of free space
- 128 MB of graphics memory

#### **3.3.2 Software Requirements**

MATLAB, if present, can directly run this software on any recent Windows, Linux or Macintosh systems. The following versions of operating systems are supported:

- Windows XP, Vista, 7, 8
- Mac OSX v10.7 or more
- Linux Ubuntu 13+, RedHat 6+, Debian 7.x

In case of embedded systems, this algorithm can be automatically translated to C using emlc/codegen, a command-line tool in Real-Time Workshop that generates C code from Embedded MATLAB code. A microcontroller with an appropriate compiler for embedded C could be used to run this.

### 3.4 Use case diagram and description

#### 3.4.1 Use case diagram

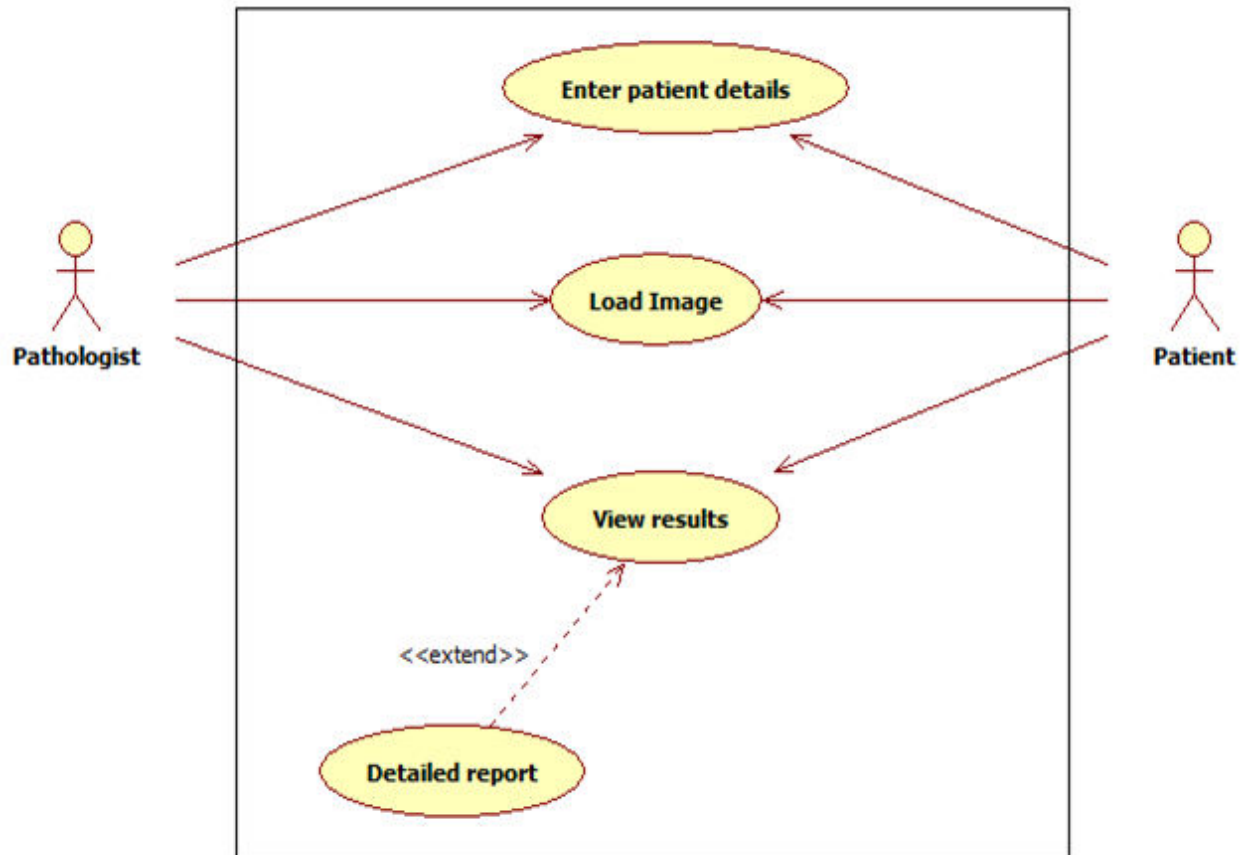


Fig. 3.4.1.1 Use Case Diagram

#### 3.4.2 Use case description

Actor	Use case	Description
Pathologist / Patient	Enter patient details	The actor initially has to feed in the details about the patient such as patient name, contact details, age, name of the medical practitioner etc.
Pathologist / Patient	Load image	High quality microscopic image of the blood smear of the patient is fed into the system for further computation and analysis to diagnose the presence of sickle shaped cells in the smear.

Pathologist / Patient	View results	On completion of the analysis carried out by the software on the input image, the result is displayed to the actors.
	Detailed report	On completion of the analysis carried out by the software on the input image, a detailed report of the analysis will be displayed to the actors

Table 3.4.2.1 Use Case Description

# Chapter 4

## Analysis Modeling

### 4.1 Activity Diagram

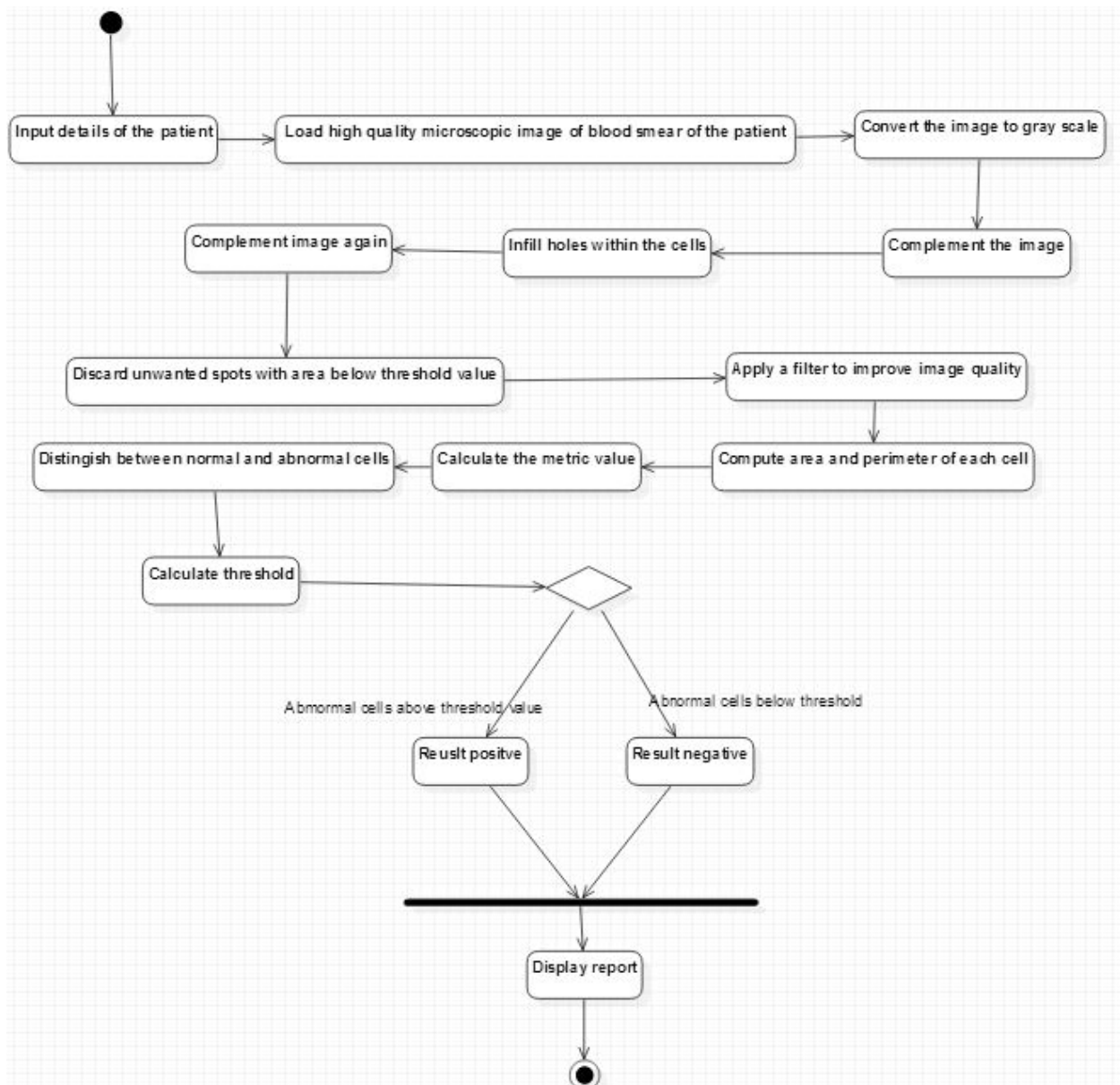


Fig. 4.1.1 Activity Diagram

## 4.2 Functional Modeling

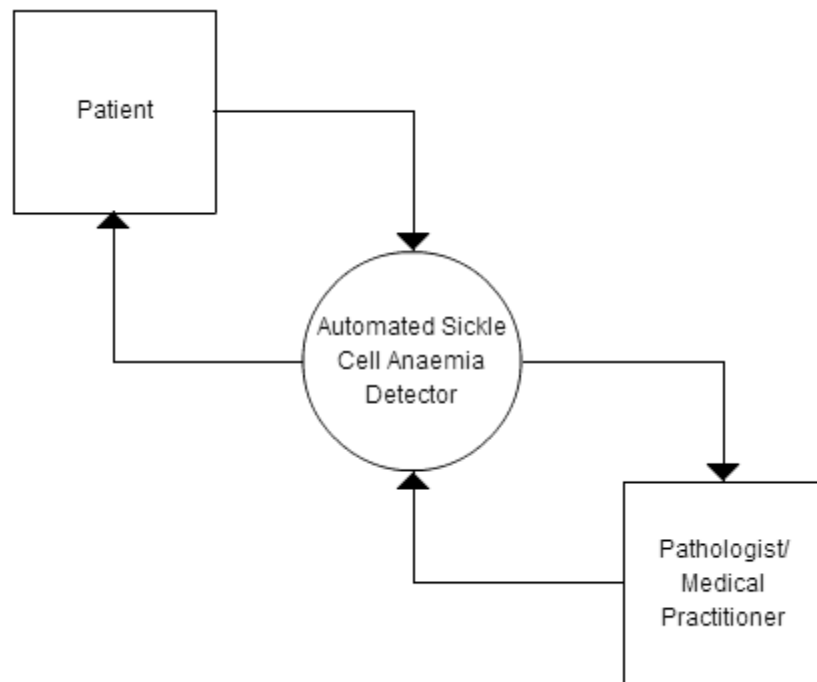


Fig. 4.2.1 Data Flow Diagram (Level 0)

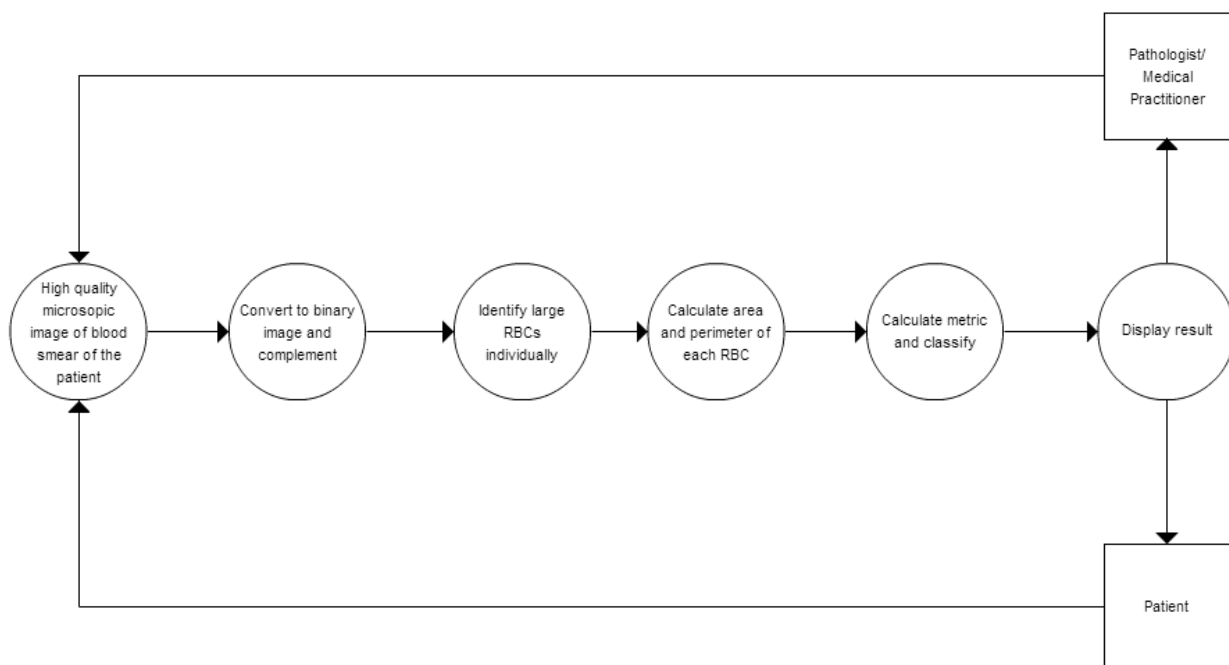


Fig. 4.2.2 Data Flow Diagram (Level 1)

### 4.3 Timeline chart

ID		Task Name	Duration	Start	Finish	Predecessors
1		<b>Documentation</b>	<b>172 days</b>	<b>Thu 8/7/14</b>	<b>Fri 4/3/15</b>	
2		<b>Understanding the paper</b>	<b>12 days</b>	<b>Thu 8/7/14</b>	<b>Fri 8/22/14</b>	
3		Studied the base paper	2 days	Thu 8/7/14	Fri 8/8/14	
4		Studied the algorithm	4 days	Mon 8/11/14	Thu 8/14/14	3
5		Referred other papers	3 days	Mon 8/18/14	Wed 8/20/14	4
6		Problem formulation	1 day	Thu 8/21/14	Thu 8/21/14	5
7		Scope and need of project	1 day	Fri 8/22/14	Fri 8/22/14	6
8		<b>Hardware Analysis</b>	<b>3 days</b>	<b>Thu 8/28/14</b>	<b>Mon 9/1/14</b>	
9		Requirement gathering	2 days	Thu 8/28/14	Fri 8/29/14	7
10		Analyse the requirements	1 day	Mon 9/1/14	Mon 9/1/14	9
11		<b>Literature Review</b>	<b>4 days</b>	<b>Tue 9/2/14</b>	<b>Fri 9/5/14</b>	
12		Survey on the topic	4 days	Tue 9/2/14	Fri 9/5/14	10
13		<b>System Analysis</b>	<b>4 days</b>	<b>Mon 9/8/14</b>	<b>Thu 9/11/14</b>	
14		Functional Requirements	1 day	Mon 9/8/14	Mon 9/8/14	12
15		Non-Functional Requirements	1 day	Tue 9/9/14	Tue 9/9/14	14
16		Specific Requirements	1 day	Wed 9/10/14	Wed 9/10/14	15
17		Use Case diagram	1 day	Thu 9/11/14	Thu 9/11/14	16
18		<b>Design</b>	<b>10 days</b>	<b>Fri 9/12/14</b>	<b>Thu 9/25/14</b>	
19		Activity diagram	1 day	Fri 9/12/14	Fri 9/12/14	17
20		Functional Modelling (DFD)	2 days	Mon 9/15/14	Tue 9/16/14	19
21		Architecture	3 days	Wed 9/17/14	Fri 9/19/14	20
22		Circuit design	2 days	Mon 9/22/14	Tue 9/23/14	21
23		User interface	2 days	Wed 9/24/14	Thu 9/25/14	22
24		<b>Synopsis</b>	<b>6 days</b>	<b>Fri 9/26/14</b>	<b>Fri 10/3/14</b>	
25		White book synopsis	6 days	Fri 9/26/14	Fri 10/3/14	23
26		<b>Implementation Plannig</b>	<b>25 days</b>	<b>Thu 1/15/15</b>	<b>Wed 2/18/15</b>	
27		Planning the modules	12 days	Thu 1/15/15	Fri 1/30/15	25
28		Gathering the test data	6 days	Mon 2/2/15	Mon 2/9/15	27
29		Checking the feasibility of the modules	7 days	Tue 2/10/15	Wed 2/18/15	28
30		<b>Implementation</b>	<b>24 days</b>	<b>Thu 2/19/15</b>	<b>Tue 3/24/15</b>	
31		Coding	24 days	Thu 2/19/15	Tue 3/24/15	29
32		<b>Testing</b>	<b>8 days</b>	<b>Wed 3/25/15</b>	<b>Fri 4/3/15</b>	
33		Testing the project	8 days	Wed 3/25/15	Fri 4/3/15	31

Figure 4.1 TimeLine Chart

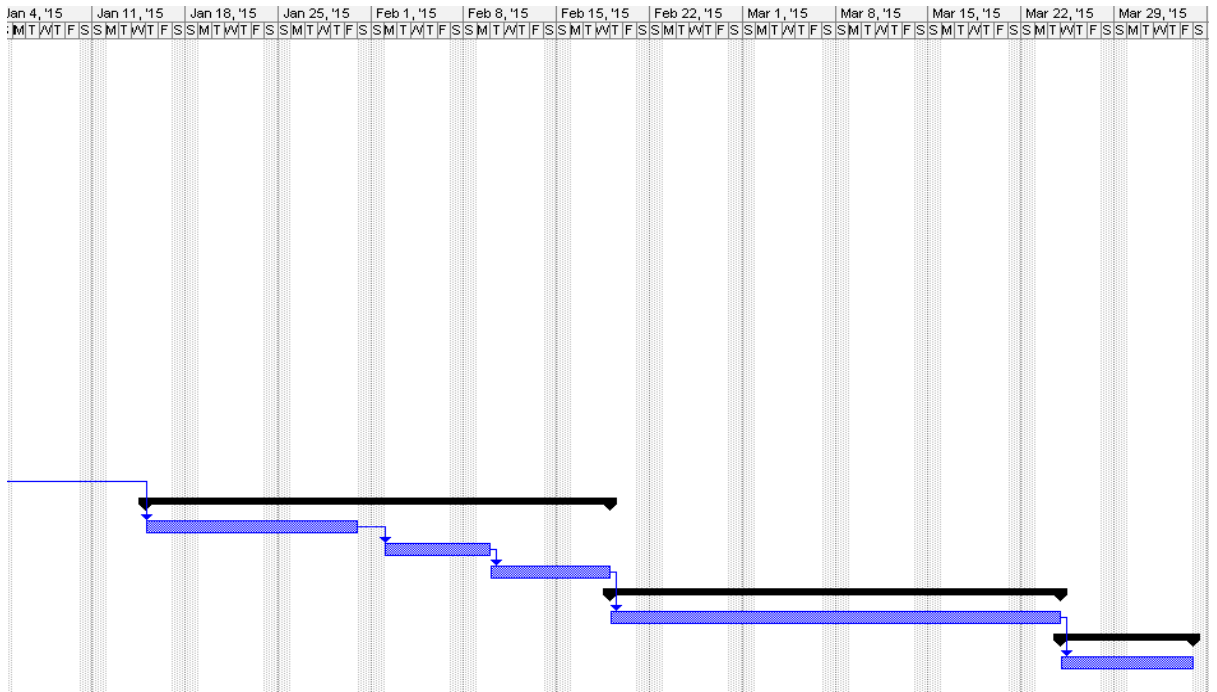
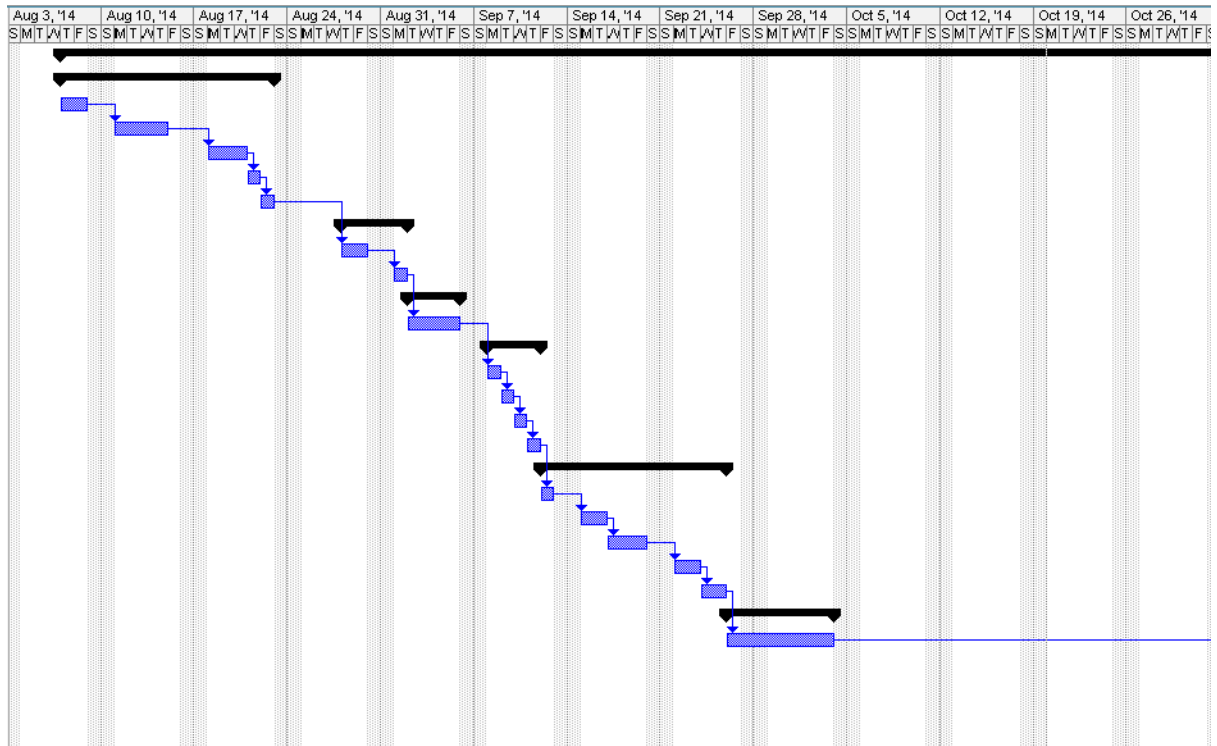


Figure 4.2 Gantt chart

# Chapter 5

## Design

### 5.1 Architectural design

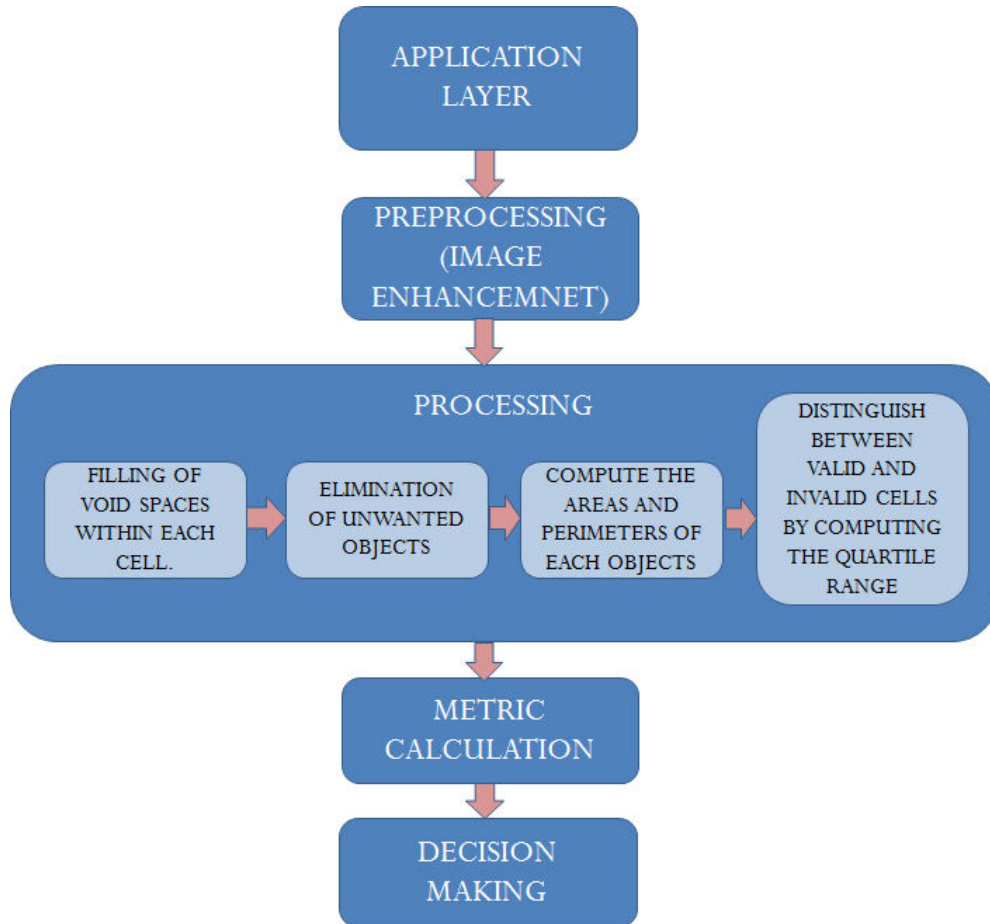


Fig. 5.1 Architectural design

The project consists of various functional stages shown in the form of modules in the diagram above. These modules function in quorum to achieve the desirable output. The explanations of each of the modules are as follows:



### Application layer:

This layer acts as the interface between the user and the internal logic of the software. The GUI is implemented on this layer. It allows the user to enter in the basic details regarding the patient and also input the microscopic blood image. After calculations and internal classification, we obtain a detailed report.

### Preprocessing:

This module is responsible for performing histogram stretching operation, conversion of the image into binary form and finally applying filtering the image to discard the unwanted objects.

### Processing:

- Filling void space:  
After conversion to grayscale, there are several cells with void spaces at the center. This has to be filled using imfill operator in MATLAB. This would help in accurate computation of the cell area.
- Elimination of unwanted objects:  
The microscopic image contains cells and object other than RBCs. These objects are filtered out by using bwareaopen function in MATLAB, which removes objects having an area below a specified threshold. It also eliminates noise and other minute unwanted components.
- Computing the areas and perimeters:  
The objects in the image (which are the connected components) are identified using the bwconncomp function. The area and perimeter of each of these objects are obtained by using regionprop function. Thus we obtain the area and perimeter of each RBC. This is then used assign a metric based on the shape of the cell.
- Distinguishing between valid and invalid cells:  
Based on the area obtained in the above step, we determine the cells that will be eligible for the metric calculation (which we term as a valid cell). Now the objects that can be classified as an outlier based on statistical definition are not considered for further processing (which we term as an invalid cell). The metric is applied on the resulting valid cells and we thus obtain the count of normal and abnormal cells. This metric is calculated by the formula  $4 \cdot \pi \cdot \text{area} / \text{perimeter}^2$ , and its value ranges between 0 and 1. Higher the value, more circular is the image. The value thus indicates the circularity of the object and can be used to obtain the count of normal and abnormal cells.

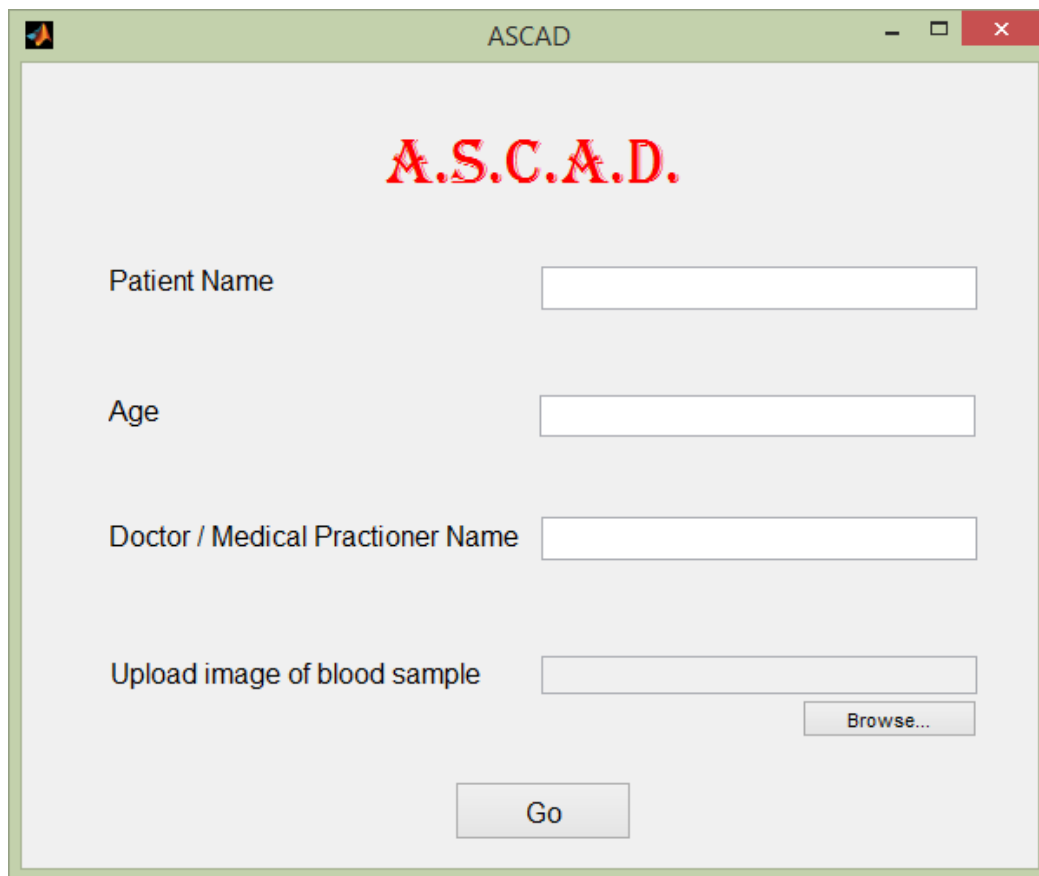
### Metric calculation:

This metric allows us to estimate the shape of the object. This metric is calculated by the formula  $4 \cdot \pi \cdot \text{area} / \text{perimeter}^2$ , and its value ranges between 0 and 1. Higher the value, more circular is the image. So a typical RBC would have a metric higher than 0.82 whereas an abnormally RBC would have a much lower metric of about 0.4 – 0.5. The value thus indicates the circularity of the object and can be used to obtain the count of normal and abnormal cells.

### Decision making:

Based on the metric above, a decision is made on whether the particular cell is normal or not. A metric of 0.75 is used as the threshold lower-bound for classifying a cell as normal. After obtaining the count of all the normal and abnormal cells, if the ratio of abnormal to normal is higher than 1:10, we arrive at the conclusion that the given sample is anaemic.

## 5.2 User Interface:



The screenshot shows a window titled "ASCAD" with a standard Windows-style title bar (minimize, maximize, close buttons). The main content area has a light gray background. At the top center, the text "A.S.C.A.D." is displayed in a large, bold, red serif font. Below this, there are four input fields arranged vertically, each with a label to its left: "Patient Name", "Age", "Doctor / Medical Practioner Name", and "Upload image of blood sample". The "Upload image of blood sample" field is a file selection box, and a "Browse..." button is located to its right. At the bottom center of the form area, there is a "Go" button.

Fig. 5.2.1. User interface

# Chapter 6

## Implementation

### 6.1 Algorithm

Step 1: Stained and magnified human blood smear image is fed to the program as input.

Step 2: Input image is converted to grayscale and is enhanced by performing histogram stretching.

Step 3: The void spaces within the cells of the enhanced image are filled using the `imfill()` command. Use of `imfill()` would require the image to be complemented before and after the operation since the function only fills white into blacks spaces bounded by white boundaries and not vice-versa.

Step 4: The resulting image is then converted into a binary image.

Step 4: The image is complemented and small unwanted spots are discarded from the image using 'bwareaopen', which removes all the objects in the diagram containing fewer than the number of pixels mentioned in the threshold level (here, threshold level= 50). The image is then complemented back to its original form for further processing.

Step 5: The cells/objects intersecting the borders of the image would produce inaccurate results. Therefore, such cells are discarded using `imclearborder()`.

Step 6: Individual cells are detected by using `bwconncomp()` command in Matlab. `bwconncomp()` returns a structure of connected components in the image passed to it. This structure is then passed to `regionprops()` to obtain the area and perimeter of every cell.

Step 7: The number of valid cells are computed by considering only the cells falling within the quartile range.

Step 8: The metric value for every valid cell is computed and stored. Metric value is computed using the formula  $4 \cdot \pi \cdot \text{area} / (\text{perimeter})^2$ .

Step 9: The valid cells are further classified as normal and abnormal cells depending on the metric value corresponding to the cells. A threshold metric value is decided (usually above 0.75), above which a cell is classified as normal.

Step 10: A threshold is decided depending on which a decision is made whether the individual is diagnosed with the disease or not. If the number of abnormal cells is equal to or more than the threshold, a positive SCA result is displayed.

## 6.2 Working of the project

### ASCAD.fig

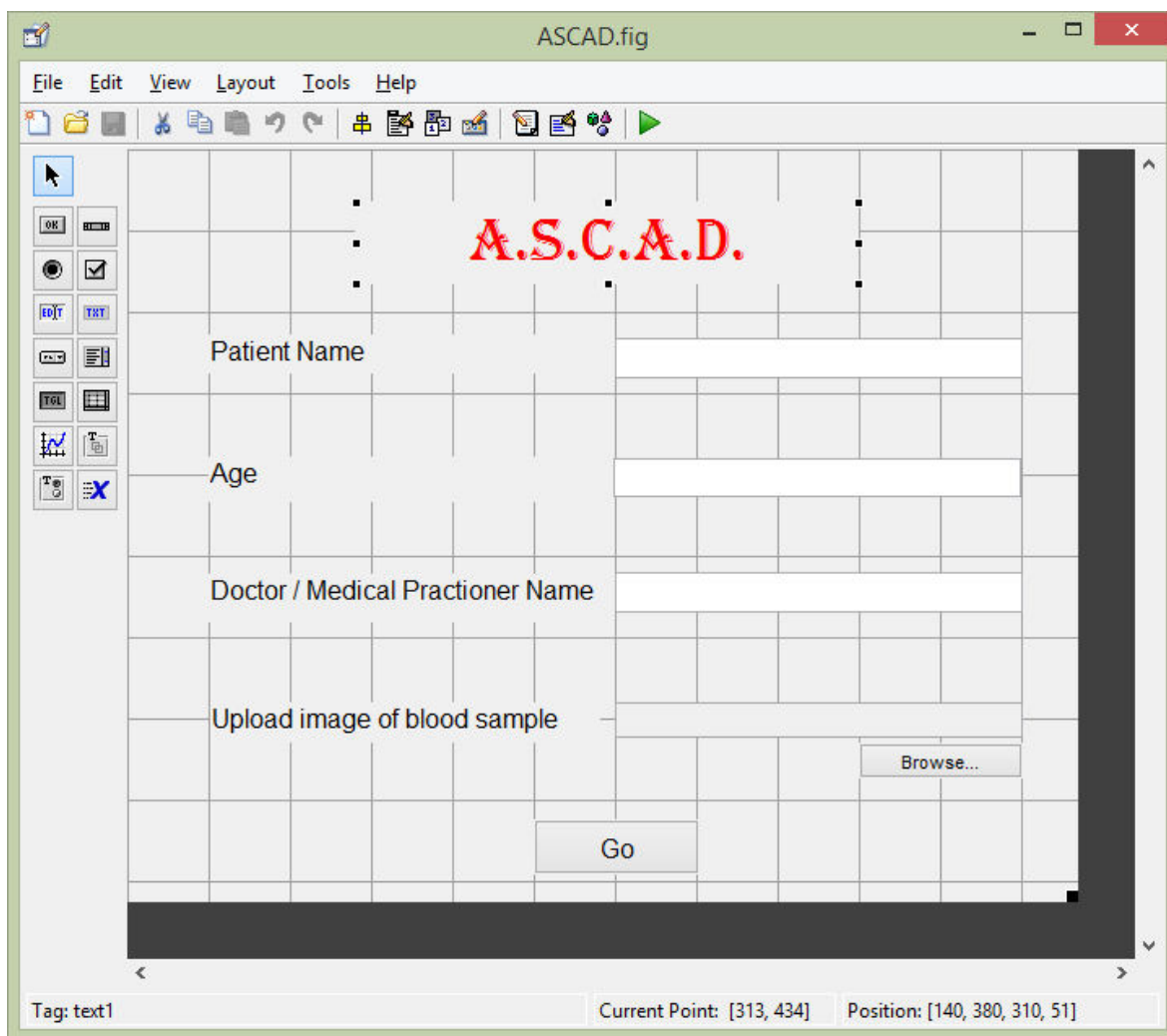


Fig. 6.2.1 GUI layout

ASCAD.m

```

function varargout = ASCAD(varargin)
% ASCAD MATLAB code for ASCAD.fig
%     ASCAD, by itself, creates a new ASCAD or raises the existing
%     singleton*.
%
%     H = ASCAD returns the handle to a new ASCAD or the handle to
%     the existing singleton*.
%
%     ASCAD('CALLBACK',hObject,eventData,handles,...) calls the local
%     function named CALLBACK in ASCAD.M with the given input arguments.
%
%     ASCAD('Property','Value',...) creates a new ASCAD or raises the
%     existing singleton*. Starting from the left, property value pairs are
%     applied to the GUI before ASCAD_OpeningFcn gets called. An
%     unrecognized property name or invalid value makes property application
%     stop. All inputs are passed to ASCAD_OpeningFcn via varargin.

% Last Modified by GUIDE v2.5 07-Apr-2015 16:20:01

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                  'gui_Singleton',   gui_Singleton, ...
                  'gui_OpeningFcn', @ASCAD_OpeningFcn, ...
                  'gui_OutputFcn',  @ASCAD_OutputFcn, ...
                  'gui_LayoutFcn',  [], ...
                  'gui_Callback',    []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

% --- Executes just before ASCAD is made visible.
function ASCAD_OpeningFcn(hObject, eventdata, handles, varargin)
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to ASCAD (see VARARGIN)

% Choose default command line output for ASCAD
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

```

```

% UIWAIT makes ASCAD wait for user response (see UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function varargout = ASCAD_OutputFcn(hObject, eventdata, handles)
% Get default command line output from handles structure
varargout{1} = handles.output;

function P_name_Callback(hObject, eventdata, handles)

% --- Executes during object creation, after setting all properties.
function P_name_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function P_age_Callback(hObject, eventdata, handles)

% --- Executes during object creation, after setting all properties.
function P_age_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function D_name_Callback(hObject, eventdata, handles)

% --- Executes during object creation, after setting all properties.
function D_name_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function img_path_Callback(hObject, eventdata, handles)

% --- Executes during object creation, after setting all properties.
function img_path_CreateFcn(hObject, eventdata, handles)

```

```

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
handles.output = hObject;
[fn pn] = uigetfile('*.jpg','select dicom file');
complete = strcat(pn,fn);
set(handles.img_path,'string',complete);

% --- Executes on button press in Go.
function Go_Callback(hObject, eventdata, handles)
P= get(handles.P_name, 'String');
A= str2num(get(handles.P_age, 'String'));
D= get(handles.D_name, 'String');
IP= get(handles.img_path, 'String');

if isempty(P)      % checking that the patient name field is not left blank
    errordlg('Please enter the Patient Name.', 'Error');
else if isempty(A) % checking that the patient age is not left blank
    errordlg('Please enter the age of the patient.', 'Error');
else if A<0        % checking if the user has entered a negative age
    errordlg('Please enter a valid age of the patient.', 'Error');
else if isempty(D);
    errordlg('Please enter the name of the doctor or medical practioner
who advised the test. Enter N.A. in case of self test.', 'Error');
else if isempty(IP)
    errordlg('Image path empty. Please browse to the input
image.', 'Error');
else
    Code(get(handles.P_name, 'String'), get(handles.P_age, 'String'),
get(handles.D_name, 'String'), get(handles.img_path, 'String'));
end
end
end
end
end
end
end
end

```

### Code.m

```

function Code(P, A, D, IP)
warning off;

P_name= P;    %input('Enter Patient name : ','s');
P_age= A;     %input('Enter Patient age : ');
D_name= D;    %input('Enter Doctor (medical practioner) name : ','s');
%img_input = input('Enter the image index : ');
img_path= IP; %strcat('C:\AAA\Img\',int2str(img_input),'.jpg');
img= imread(img_path);

```

```

%figure, imshow(img);

img_grayS = rgb2gray(img);
%figure, imshow(img_grayS);

%histogram stretching
bins = linspace(0,255,256);
H = hist(img_grayS(:), bins);
H(H==0) = eps(sum(H));
cdf = [0,cumsum(H)/sum(H)]; %cumulative distribution function
pct= 0.05; %percent of pixel values to ignore
h_low = interp1(cdf, [0,bins], pct);
h_high = interp1(cdf, [0,bins], 1-pct);
stretchedImg= uint8((double(img_grayS)-h_low)/(h_high-h_low) * 255);
%img_grayS= uint8(imadjust(img_grayS, stretchlim(img_grayS), []));
%histogram stretching using inbuilt function
%figure, imshow(stretchedImg);

BW_c= imcomplement(stretchedImg);
BW_filled= imfill(BW_c,4, 'holes');
BW_refined= imcomplement(BW_filled);
%figure, imshow(BW_refined);

%Gaussian filter
myfilter = fspecial('gaussian',[3 3],16);
filteredImg = imfilter(BW_refined, myfilter, 'replicate');

img_binary= im2bw(BW_refined,graythresh(filteredImg)); % converting the
enhanced image to binary
%figure, imshow(img_binary);

img_binary_c= imcomplement(img_binary); %bwareaopen and bwconncomp works
only on bright pixels
refined_img= bwareaopen(img_binary_c,200, 4); %removing unwanted spots
from the image having a maximum pixel density of 200 pixels
%figure, imshow(refined_img);

%EROSION (EXPERIMENTAL)
se = strel('disk',2);
erodedBW = imerode(refined_img,se); %Shrinks each 'disk' shaped object in
the image
%figure, imshow(erodedBW);

img_binary1 = imclearborder(erodedBW); %remove border objects
%figure, imshow(img_binary1);

CC= bwconncomp(img_binary1, 4); % keeping connectivity 4 to ignore
diagonal connection
CA= regionprops(CC, 'Area'); %returns the areas of all objects in the
image identified by bwconncomp()
CP= regionprops(CC, 'Perimeter'); %returns the perimeters of all objects
in the image identified by bwconncomp()
img_binary2= imcomplement(img_binary1);

```



```

areas= cell(CC.NumObjects,1);    %storing the area of individual components
from CA.Area                    %CA.Area is not feasible to use for
                                computations

for i=1: CC.NumObjects
    areas{i,1}= CA(i,1).Area;
end
areas= cell2mat(areas);    %converting the cell to array

iqr= quantile(areas,0.75)-quantile(areas,0.25);    %inter quartile range

average=iqr*1.5;    %going by the definition of an outlier

%counting the number of valid and invalid cels in the image (only considering
the objects with areas within the inter quartile range)
validCells=0; invalidCells=0;
for i=1:CC.NumObjects
    if CA(i,1).Area >= quantile(areas,0.25)-average && CA(i,1).Area <=
quantile(areas,0.75)+ average
        validCells= validCells+1;
    else
        invalidCells= invalidCells+1;
    end
end

%calculating the metric value for each valid cell
metric= cell(validCells ,1);
k=1;
for i=1:CC.NumObjects
    if CA(i,1).Area >= quantile(areas,0.25)-average && CA(i,1).Area <=
quantile(areas,0.75)+ average
        metric{k,1} = (4*pi*CA(i,1).Area)/(CP(i,1).Perimeter *
CP(i,1).Perimeter);
        k=k+1;
    end
end

normal=0; abnormal=0;

for i=1:validCells
    if metric{i,1}>=0.75    %all cells with a metric value above 0.75 are
considered normal, as they prove to be more circular
        normal=normal+1;
    else
        abnormal=abnormal+1;
    end
end

threshold=0.1*validCells;    % 10 percent of the number of valid cells
(Experimental)
                                % Reference - http://goo.gl/E4NCam

```

```

%printing the report
disp(sprintf('\n\nReport :-'));
disp(sprintf('Patient name : %s', P_name));
disp(sprintf('Patient age : %s', P_age));
disp(sprintf('Doctor / Medical practioner name : %s', D_name));
disp(sprintf('Image Path : %s', img_path));
disp(sprintf('\nTotal number of objects detected\t= %d',CC.NumObjects));
disp(sprintf('Number of valid cells\t\t\t\t\t= %d',validCells));
disp(sprintf('Number of normal cells\t\t\t\t\t= %d',normal));
disp(sprintf('Number of abnormal cells\t\t\t\t\t= %d',abnormal));
disp(sprintf('Threshold \t\t\t\t\t\t\t\t\t= %f',threshold));
if abnormal>threshold
    disp(sprintf('\nResult : \t\t\tYou are diagnosed with Sickle Cell
Anaemia.\n'));
    msgbox(sprintf(' Patient name : %s\n Patient age : %s\n Doctor / Medical
practioner name : %s\n Image Path : %s\n\n Total number of objects
detected\t= %d\n Number of valid cells\t\t\t\t\t= %d\n Number of normal
cells\t\t\t\t\t\t\t= %d\n Number of abnormal cells\t\t\t\t\t= %d\n Threshold
\t\t\t\t\t\t\t\t\t= %f\n\n Result : \t\t\tYou are diagnosed with Sickle Cell
Anaemia.\n', P_name, P_age, D_name, img_path, CC.NumObjects, validCells,
normal, abnormal, threshold),'Report- Positive');
else
    disp(sprintf('\nResult : \t\t\tYou are safe.\n'));
    msgbox(sprintf(' Patient name : %s\n Patient age : %s\n Doctor / Medical
practioner name : %s\n Image Path : %s\n\n Total number of objects
detected\t= %d\n Number of valid cells\t\t\t\t\t= %d\n Number of normal
cells\t\t\t\t\t\t\t= %d\n Number of abnormal cells\t\t\t\t\t= %d\n Threshold
\t\t\t\t\t\t\t\t\t= %f\n\n Result : \t\t\tYou are safe.\n', P_name, P_age, D_name,
img_path, CC.NumObjects, validCells, normal, abnormal, threshold),'Report-
Negative');
end

end    %function Code end

```

## Chapter 7

### Testing

A.S.C.A.D. is a software tool that deals in a medical context, and hence must include minimal errors. To achieve such accuracy, the software needs to be tested well.

Software testing is a process of executing a program or application with the intent of finding software bugs. In other words, testing is executing a system in order to identify any gaps, errors, or missing requirements in contrary to the actual requirements.

#### 7.1 Test cases

Test Case	Expected Result	Actual Outcome	Result
Blank patient name	Error : Please enter the Patient name.	Error : Please enter the Patient name.	Success
Blank patient age	Error : Please enter the age of the patient.	Error : Please enter the age of the patient.	Success
Invalid patient age	Error : Please enter a valid age of the patient.	Error : Please enter a valid age of the patient.	Success
Blank doctor name	Error : Please enter the name of the doctor or medical practitioner who advised the test. Enter N.A. in case of self test.	Error : Please enter the name of the doctor or medical practitioner who advised the test. Enter N.A. in case of self test.	Success
Blank image path	Error : Image path empty. Please browse to the input image.	Error : Image path empty. Please browse to the input image.	Success
Trying to enter an invalid image path.	The user must only be allowed to enter a legal path.	The text box to enter the path to the input image is disabled. As a result, the user has to browse to the image using the 'Browse' button.	Success

Table 7.1.1. Test case table

## 7.2 Type of testing used

Black box testing:

Black-box testing is a method of software testing that examines the functionality of an application without peering into its internal structures or workings. This method of test can be applied to virtually every level of software testing: unit, integration, system, functional and acceptance.

- Functional testing

Functional testing is a quality assurance (QA) process and a type of black box testing that bases its test cases on the specifications of the software component under test. Functions are tested by feeding them input and examining the output, and internal program structure is not considered. Functional testing tests a slice of functionality of the whole system.

Test cases:

1. Blank patient name

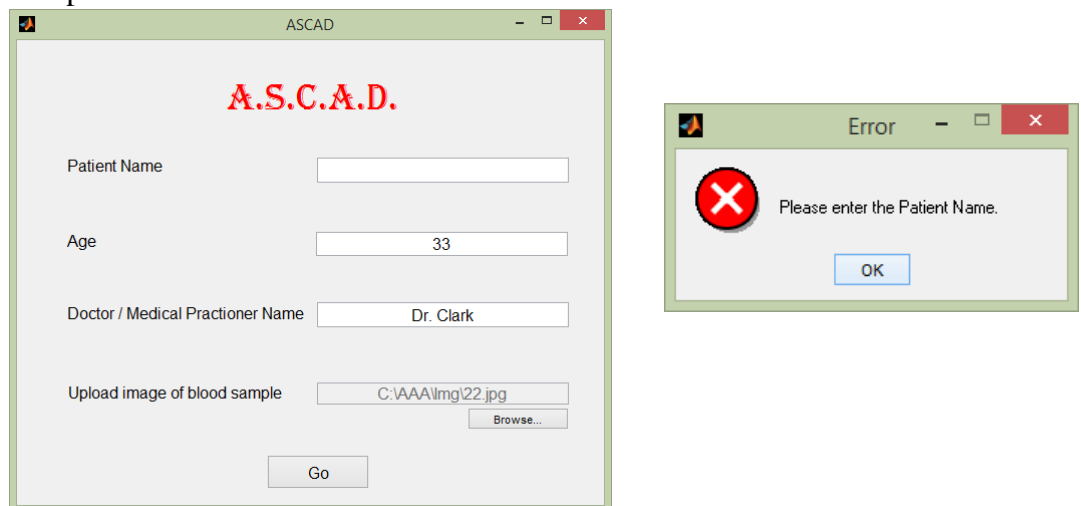


Fig. 7.2.1. Testing : Blank patient name

2. Blank patient age

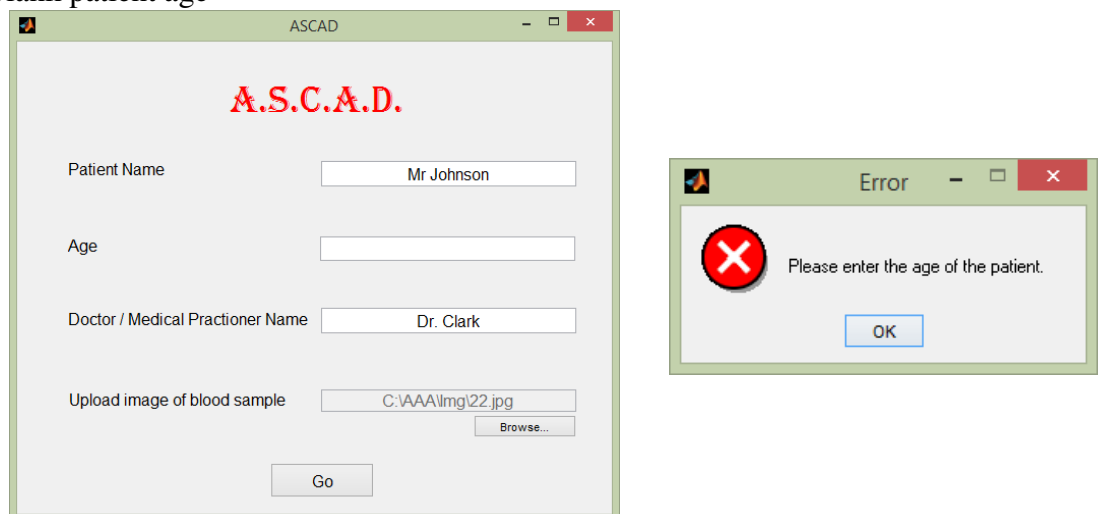


Fig. 7.2.2. Testing : Blank patient age

## 3. Invalid patient age

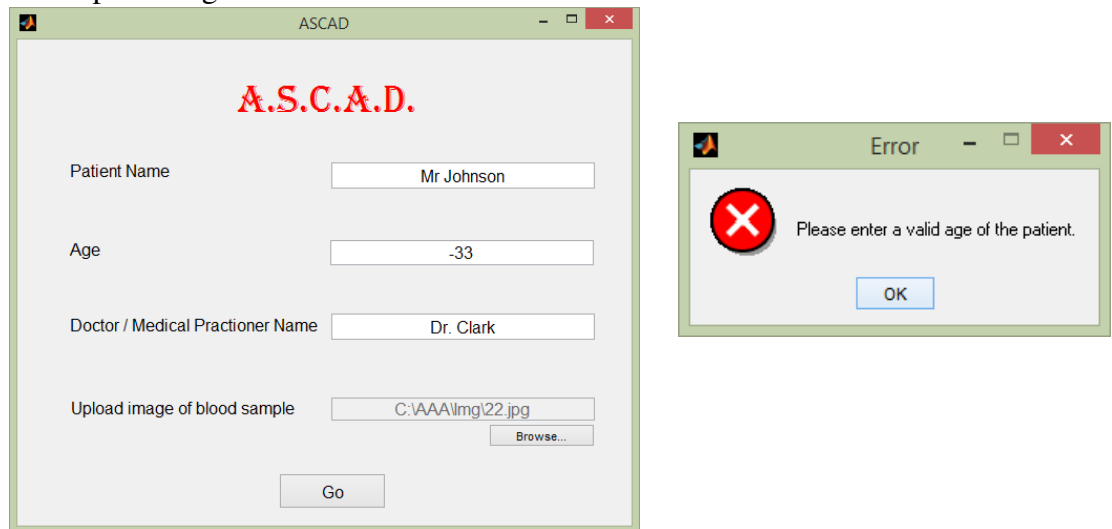


Fig. 7.2.3. Testing : Invalid patient age

## 4. Blank doctor name

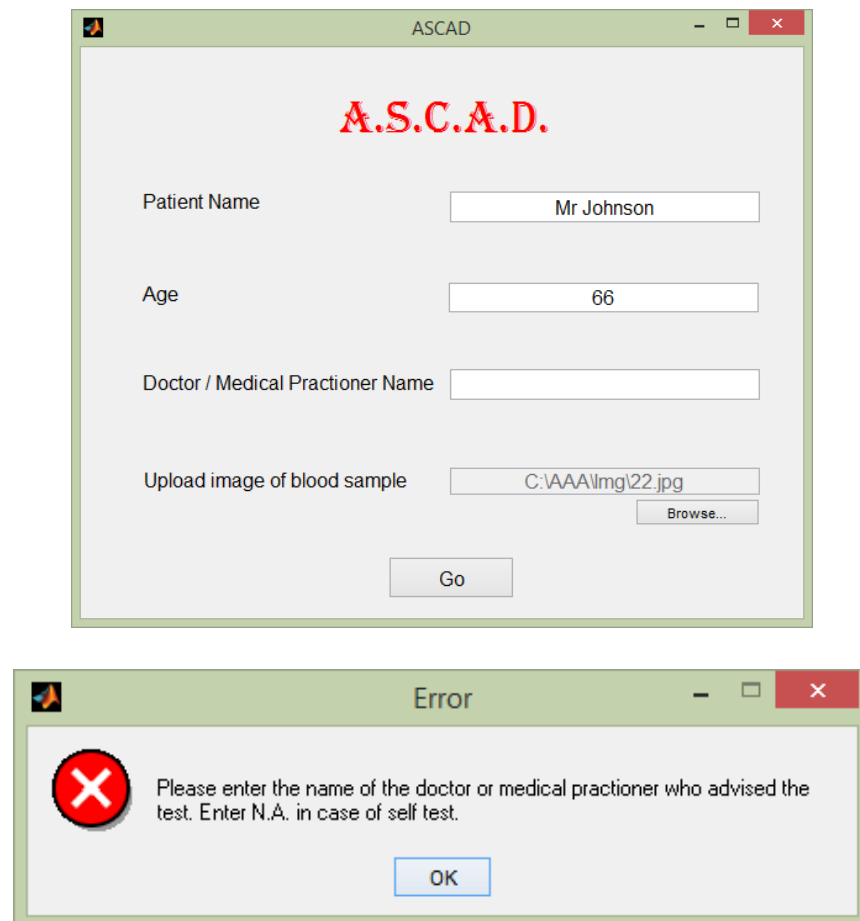
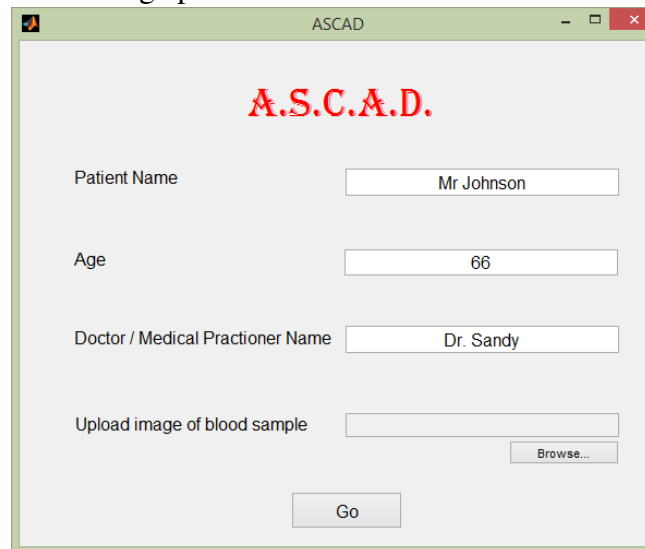


Fig. 7.2.4. Testing : Blank doctor name

## 5. Blank image path



ASCAD

**A.S.C.A.D.**

Patient Name

Age

Doctor / Medical Practitioner Name

Upload image of blood sample

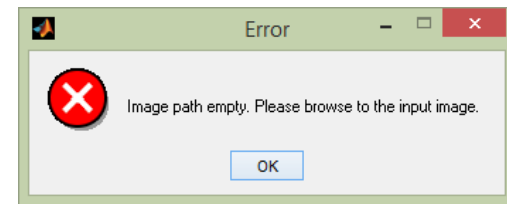
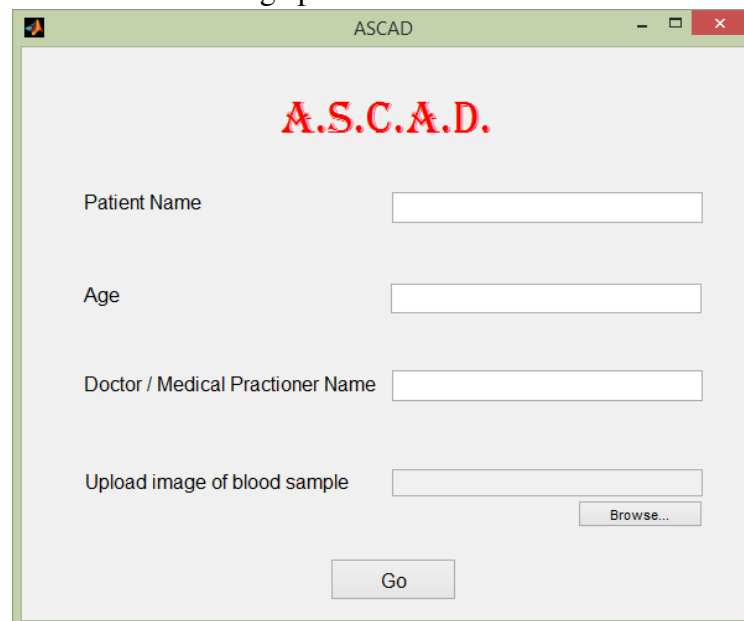


Fig. 7.2.5. Testing : Blank image path

## 6. Trying to enter an invalid image path.



ASCAD

**A.S.C.A.D.**

Patient Name

Age

Doctor / Medical Practitioner Name

Upload image of blood sample

Fig. 7.2.6. Testing : Trying to enter an invalid image path

## Chapter 8

### Result and Discussion

The result was thoroughly analyzed and recorded at each step. The following are the results obtained after providing an input with of a blood smear containing sickle cells.

#### 1. Result of histogram stretching

Histogram stretching (or normalization), is the process of improving or enhancing the contrast of an image. Contrast is the difference between maximum and minimum pixel intensity. This is performed in the software to obtain a clear distinction between a cell and its background.

Given Input: A gray scale image with poor contrast.

Expected Output: Improved contrast.

Observed Output: Better dynamic range and noticeable enhancement in contrast.



Fig. 8.1. Result of histogram stretching

#### 2. Resulting image after imfill operator

The cells contain void spaces within their centers. These have to be filled up, so that the whole cell can be detected as one object and its area can be calculated. So we use the imfill operator to achieve this.

Given Input: An image with cells having while nucleus.

Expected Output: Uniformly coloured cells.

Observed Output: The centers of the cells get filled with appropriate colours.

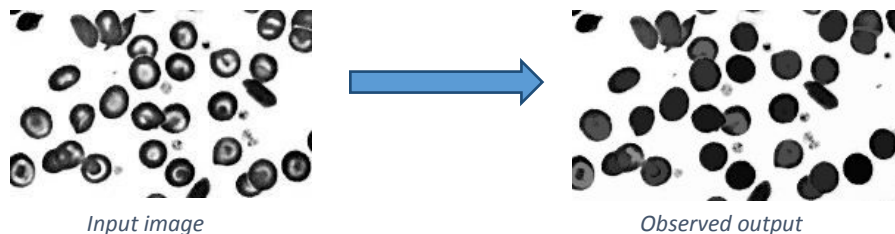


Fig. 8.2. Resulting image after imfill operator

The result obtained from the above were analyzed and verified with the expected output. A randomized dataset consisting of 6 normal and 6 abnormal were used to estimate the accuracy of the software.

### 3. Result of filtering the image

There were small non RBC objects and other minute particles in the background of the sample image. These had to be eradicated to obtain appropriate result. This was done using bwareaopen.

Given Input: An image small objects and noise.

Expected Output: Only RBCs are retained.

Observed Output: The smaller objects are eradicated retaining only the actual RBCs.

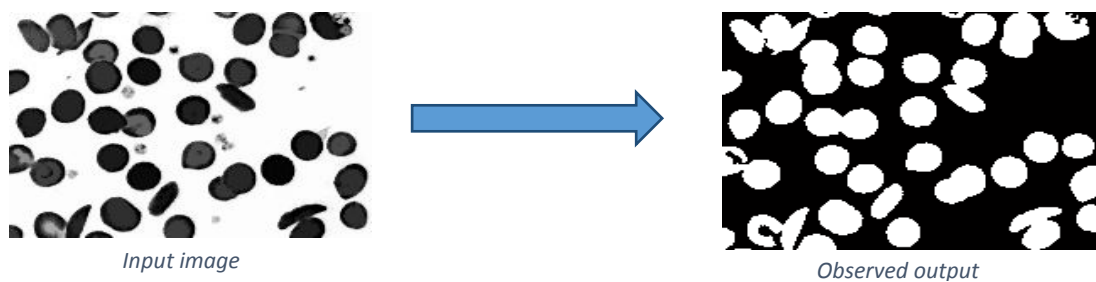


Fig. 8.3. Result of filtering the image

### 4. Result of erosion and clearing the border objects

The image was eroded and the border object were cleared. The erosion was performed to shrink individual objects with circular shape so that the ones being slightly overlapped become separated. The objects touching the border contain less information and is thus eliminated by using the imclearborder function.

Given Input: An image with overlapping objects and objects touching the boundary.

Expected Output: Individual cells being shrunk and objects touching the boundary eliminated.

Observed Output: Output as expected.

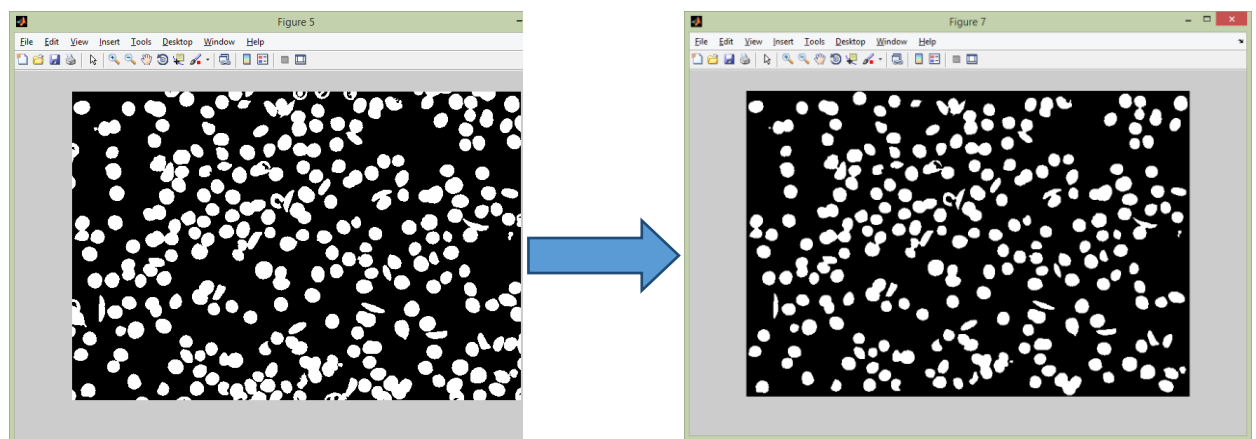


Fig. 8.4. Result of erosion and clearing the border objects

The final result obtained was recorded for a dataset of 10 randomized images. The findings were analyzed and the accuracy was calculated from it.



The following results were yielded from the dataset:

Image	Valid Cells Found/Total	Abnormal cells	Threshold	Observed Result	Actual Status
Exhibit 1	267/341	9	26.7	Healthy	Healthy
Exhibit 2	256/333	11	25.6	Healthy	Healthy
Exhibit 3	188/210	26	18.8	Anaemic	Anaemic
Exhibit 4	164/188	18	16.4	Anaemic	Anaemic
Exhibit 5	244/295	14	24.4	Healthy	Healthy
Exhibit 6	189/216	29	18.6	Anaemic	Anaemic
Exhibit 7	65/73	1	6.5	Healthy	Healthy
Exhibit 8	46/53	4	4.6	Healthy	Anaemic
Exhibit 9	186/209	25	18.6	Anaemic	Anaemic
Exhibit 10	46/63	1	4.6	Healthy	Healthy

Table 8.1. Result analysis

From the above analysis, it is seen that ASCAD provided the expected results in 9 out of the 10 samples used. We can therefore conclude that the software has an accuracy of about 90%. However, a thorough analysis with a much larger dataset would be required before we can determine its real accuracy. This was unfortunately not possible in our project owing to insufficient data available to us.

The data obtained by us were from foreign organizations that had made the medical data available to public domain. One of the reasons for insufficient data regarding sickle cell anaemia in India was found to be due to lack of digitalization of medical data and reluctance of both public and private institutions in sharing information for R&D. This hinders the software from realizing its true potential. But it wouldn't be a stretch to assume that this would change in the near future as digitalization spreads deeper into India and digital records become mainstream.

## **Chapter 9**

### **Conclusions and Future Scope**

India is a country with a population of over 1.27 billion people. It is estimated that out of this, 200 million suffer from some kind of disease at any given point of time [6]. On top of the world second largest sick population, a poor health-care funding and lack of R&D in medicine paints a deplorable picture of the prevalent health-care standard in our country.

In such a scenario, our solution to sickle-celled disease is what we believe, an initiative in the right direction. By expediting the diagnostic processes, we not only seek to improve the health individually but also curtail unnecessary industry-wide expenditure involved in long & manual laboratorial work. Furthermore, A.S.C.A.D, if successful as a consumer product, has the potential to open up a gateway to private research and development investment into automation in health-care sector. Such solutions that target the root causes are indispensable in improving the health-care standard in our country.

Looking beyond, this software has greater prospects when it can be optimized to exploit a larger sickle cell database as mentioned above. Optimization can also be done in the way certain pre-defined values in A.S.C.A.D can be generalized to work for a much wider range of input possibilities. However, such modifications would need more knowledge from a medical database, much wider than what is currently available. But on the whole, with an ever increasing influence of digital technologies, we can confidently envision a rising trend of the utility and use of A.S.C.A.D and similar software.