AUTOMATIC DETECTION OF MGD LEVEL (MEIBOMIAN GLAND DYSFUNCTION)

ALP EREN BÜTÜN

MASTER'S THESIS

Submitted to the School of Graduate Studies of
Kadir Has University in partial fulfillment of the requirements for the degree of
Master of Science in Computer Engineering

İSTANBUL, May, 2019

DECLARATION OF RESEARCH ETHICS /
METHODS OF DISSEMINATION

I, ALP EREN BÜTÜN, hereby declare that;

- this master's thesis is my own original work and that due references have been appropriately provided on all supporting literature and resources;
- this master's thesis contains no material that has been submitted or accepted for a degree or diploma in any other educational institution;
- I have followed *Kadir Has University Academic Ethics Principles prepared in accordance with The Council of Higher Education's Ethical Conduct Principles.*

In addition, I understand that any false claim in respect of this work will result in disciplinary action in accordance with University regulations.

Furthermore, both printed and electronic copies of my work will be kept in Kadir Has Information Center under the following condition as indicated below:

- [x] The full content of my thesis will be accessible from everywhere by all means.
- [ ] The full content of my thesis will be accessible only within the campus of Kadir Has University.
- [ ] The full content of my thesis will not be accessible for ____ years. If no extension is required by the end of this period, the full content of my thesis will be automatically accessible from everywhere by all means.

ALP EREN BÜTÜN

21 May 2019

KADİR HAS UNIVERSITY

SCHOOL OF GRADUATE STUDIES

ACCEPTANCE AND APPROVAL

This work entitled AUTOMATIC DETECTION OF MGD LEVEL (MEIBOMIAN GLAND DYSFUNCTION) prepared by ALP EREN BÜTÜN has been judged to be successful at the defense exam on 21 May 2019 and accepted by our jury as

APPROVED BY:

Assoc. Prof. Dr. Tamer Dağ (Advisor)
Kadir Has University .....

Assoc. Prof Dr. Atilla Özmen
Kadir Has University .....

Assoc. Prof. Dr. Tansal Güçlüoğlu
Yıldız Technical University .....

I certify that the above signatures belong to the faculty members named above.

.....

Prof. Dr. Sinem ARGÜL AÇIKMEŞE

Dean of School of Graduate Studies

DATE OF APPROVAL: 21 May 2019

TABLE OF CONTENTS

AUTOMATIC DETECTION OF MGD LEVEL (MEIBOMIAN GLAND DYSFUNCTION)

ABSTRACT

MGD is a common, clinical disease which is the major cause of dry eye. Meibography is a study in order to observe meibomian glands. A meibography image is the infrared photograph of the inner part of eyelids of the patient. Physicians are using meibography images to diagnose MGD. The important result from the infrared meibography image is the level of the disease. Using the level, the appropriate treatment is decided by the physician.

Aim of this thesis is to develop the automatic software to detect MGD level from meibography images. Current methods require manual processing takes a long time for detection of MGD level. Thus our aim is to develop an algorithm running in reasonable time and easier to use than the current methods.

To achieve our goal, image processing techniques supported by manual editing are used. In addition, some mathematical methods such as drawing line chart, finding local maximum, local minimum, processing image pixels, finding mean, ratio and other methods are used. Manual editing and automatic determination feature is provided to the user of the software. Physician can load a meibography image to the software and can get results in seconds. Result can also be changed.

A software using the algorithm described in this thesis is developed and tested. Development of the automatic software to process meibography images for MGD level detection is completed. For testing and valdiation, Unit tests on the code are done. In addition, Bland&Altman method is applied on the results of the software.

Keywords: Meibography, MGD, Software, Algorithm

MGD SEVİYESİNİN OTOMASYON YOLUYLA TEŞHİSİ

ÖZET

MGD göz kuruluğunun başlıca nedeni olarak görülür. Kronik ve klinik bir hastalıktır. Meibography, meibomian glandleri gözlemlemek amacı ile geliştirilmiş bir araştırmadır. Bir meibography fotoğrafı ise tedavi edilen kişinin göz kapağının iç kısmının infrared fotoğrafıdır. Infrared meibography fotoğrafından alınabilecek önemli bilgi hastalığın seviyesidir. Meibography fotoğrafları MGD seviyesini tespit etmek amacıyla kullanılır ve hastalığın seviyesine göre bir tedavi uygulanır.

Tezin amacı, meibography fotoğraflarından MGD hastalığının seviyesini tespit eden bir algoritma ile birlikte bir yazılım geliştirmektir. Mevcut yöntemler düşünüldüğünde, doktorların zaman alıcı işlemler yapması nedeniyle amaç makul süre içinde çalışan bir algoritma ve mevcut yöntemlerden daha kolay kullanılan bir yazılım geliştirmektir.

Temel yöntem olarak, elle düzenlemenin desteklediği resim işleme teknikleri kullanılmıştır. Çizgi grafiği çizmek, yerel en büyük, en küçük değer tespiti, resim pixel işlemleri, oran bulma ve diğer matematiksel yöntemler kullanılmıştır. Ek olarak, tezde açıklanan algoritmayı kullanan bir yazılım geliştirilmiştir. Doktor, geliştirilen yazılımı kullanarak bir meibography fotoğrafı yükleyebilir ve sonucu saniyeler içinde alabilir. Ek olarak, sonuç doktor tarafından düzenlenebilir.

Tezde açıklanan algoritma ve yöntemleri kullanan bir yazılım geliştirilmiş, testler uygulanmış ve revizeler yapılmıştır. Bir meibography fotoğrafını işlemek amacıyla kullanılacak bir uygulamanın geliştirilmesi ve testleri tamamlanmıştır. Ek olarak, yazılımın sonuçları üzerinde Bland&Altman testleri uygulanmıştır.

Anahtar Sözcükler: Meibography, MGD, Yazılım, Algoritma

ACKNOWLEDGEMENTS

I thank Prof. Dr. Banu Bozkurt for allowing me to use undisclosed meibography images for testing and validating the software.

I thank Assoc. Prof. Dr. Tamer Dağ for starting this thesis and giving the idea the thesis is based on. Also, Bland&Altman Excel document is shared by Tamer Dağ.

LIST OF FIGURES

LIST OF SYMBOLS/ABBREVIATIONS

MGD	Meibomian Gland Dysfunction
RGB	Notation for Color Representation
R	Red
G	Green
B	Blue
Greyscale	Monochromatic
IR	Infrared
PHP	Personal Home Page
JS	Javascript Programming Language
NodeJS	Server Side Javascript Programming Language
AJAX	Asynchronous JavaScript and XML
HTTP	Hyper Text Transfer Protocol
URL	Uniform Resource Locator
Pre-defined	Built-in
B&A	Bland-Altman
HTML	Hypertext Markup Language
GD	GIF Draw / Graphic Draw
px	Pixel
OS	Oculus Sinister(Left eye)
OD	Oculus Dextrus(Right eye)
OSDI	Ocular Surface Disease Index
Non-Obvious MGD	The condition that MGD can not be determined
Frontend	Visible part of the application
Backend	Server side of the application
jpeg	Joint Photographic Expert Group
gif	Graphic Interchange Format
bmp	Bitmap
png	Portable Network Graphic image format

1. INTRODUCTION

MGD is a disease which is common and clinical and the major cause of dry eye. MGD is the abnormal condition of meibomian glands and is known as chronic [1][2][3]. The superficial tear film is using lipids contributed by glands for tear production. Disfunction of meibomian glands is resulting in a disease called as MGD that causes dry eye [4]. Effects can be counted as discomfort, visual disturbance and contact lens intolerance [6]. Determination of MGD level is important for the treatment of the disease. Fortunately, meibography helps physicians to take infrared photographs of the inner part of the eyelid of the patient. This process is called as meibography which is a study developed for the purpose of observing meibomian glands [4]. Inner area of the eyelid is including signs of MGD. Disease can be observed by looking at the images but calculating the level is challenging.

The way for determination of the MGD level is using meibography, the imaging study developed for the purpose of observing meibomian glands [4]. Physician takes the infrared photograph of the inner part of the eyelid of the patient in order to observe. Using the image, the physician calculates the level of the disease. According to level, appropriate treatment for the patient is decided.

Following images are examples of meibography images

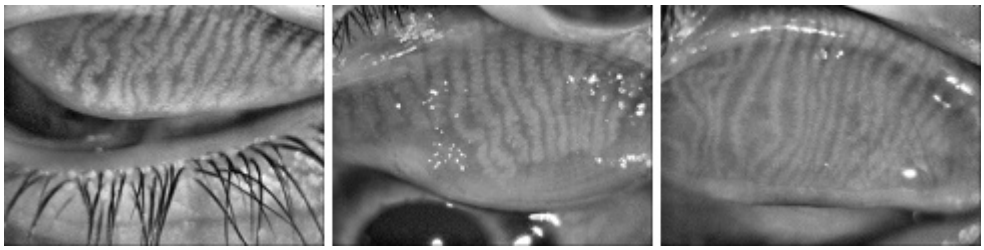

Figure 1.1 Example meibography images (Scaled to 25%)

MGD level is calculated using the percentage of "Area of Loss" in the interest area. In this thesis, interest area of a meibography image is the region indicating the focus area for the software. According to the output of the software, interest area have two types of pixels. One type is MGD pixel and the other is NO-MGD. Area of Loss is the percentage of NO-MGD pixels. From 0% to 100%, each percentage has a corresponding level according to the metric. There are different metrics for calculating the level of the disease. One of the metrics is three level using intervals with %33.3 length. Another is four level, intervals with %25 length.

Aim of this thesis is to automate the calculation of MGD level by calculating Area of Loss inside the interest area of the meibography image.

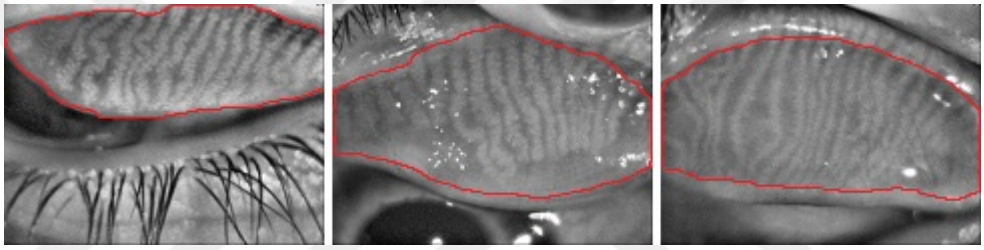

Figure 1.2 Interest area of meibography images (Scaled to 25%)

MGD can be observed by looking at the vertical channels inside the interest area of the image. Vertical light color channels are the indicators of MGD. Physicians are using meibography images not only for diagnosing the disease but also calculating the level for finding the appropriate treatment. There are multiple metrics for the level of the disease, finding the percentage of MGD pixels, thus finding the "Area of Loss" will give the level of the disease according to one of the metrics.

In Chapter 1 (Introduction), the subject of thesis along with the organization is presented. Chapter 2 (Meibomian Gland Dysfunction) is about the definition, the metrics, diagnosis and treatment of MGD. In Chapter 3 (Automatic Detection of MGD Level), methods for automatic level detection are developed, pseudocode of methods are presented. Chapter 4 (Software of This Thesis) presents the software of this thesis. In Chapter 5 (Testing and Validation), Bland&Altman test is applied to results. In Chapter 6, conclusions of this thesis is presented.

2. MEIBOMIAN GLAND DYSFUNCTION

2.1 Definition of MGD

"Meibomian gland dysfunction (MGD) is a chronic, diffuse abnormality of the meibomian glands, commonly characterized by terminal duct obstruction and/or qualitative/quantitative changes in the glandular secretion. This may result in alteration of the tear film, symptoms of eye irritation, clinically apparent inflammation, and ocular srface disease." This recommended definition is taken from the Report of the Definition and Classification Subcommittee on The International Workshop on Meibomian Gland Dysfunction.

MGD is a disease because of the abnormal condition of meibomian glands. MGD is known as common, clinicial and chronic. Tears are important for protecting the eye from unwanted conditions. Tears are produced by superficial tear film using the lipids contributed by meibomian glands. Abnormal condition of glands resulting in dysfunction and blocking tear production.

People who have MGD are seeing the effects of dry eye. These effects can be counted as discomfort, visual disturbance and contact lens intolerance. Treatment of MGD is possible. Patient gets a treatment from the physician according to level of disease.

MGD is diagnosed using infrared meibography image of the inner part of eyelids of patient. The image is processed by the physician in order to determine level of the disase. There are different metrics for determination of level and methods for processing the image. Aim of all methods and metrics is to determine the correct level and decide the appropriate treatment.

2.2 Metrics for Calculating MGD Level

Two metrics can be counted for calculating MGD level. One of the metrics is using four levels while another is using three levels. Metrics are using the value of area of loss from processed meibography images. According to value of area of loss, metrics are giving the level as output. Difference between metrics is that one of them is using four intervals with 25% length, other is using three intervals with 33.3% length.

Both metrics are using the value of "Area of Loss" for level determination. Area of Loss is the percentage of pixels marked as NO-MGD in the interest area of the processed meibography image. Other set of pixels in the interest area is MGD pixels.

One metric is using following intervals:

```
0% <= Area of Loss <= 25%   LEVEL 1
25% < Area of Loss <= 50%   LEVEL 2
50% < Area of Loss <= 75%   LEVEL 3
75% < Area of Loss <= 100%  LEVEL 4
```

Other metric is using following intervals:

```
0%   <= Area of Loss <= 33.3%   LEVEL 1
33.3% < Area of Loss <= 66.6%   LEVEL 2
66.6% < Area of Loss <= 100%    LEVEL 3
```

Metrics are checking the value of "Area of Loss". Since there are only two types of pixels in the interest area, finding the ratio of MGD pixels is giving the ratio of NO-MGD pixels (Area of Loss). Algorithm of this thesis is trying to fing MGD pixels in the interest area but the important value is Area of Loss which the algorithm calculates by substracting MGD pixel ratio from one hundred.
(Area of Loss = 100 - Percentage of MGD pixels)

2.3 Diagnosis of MGD

There are current methods used by physicians for determination of the MGD level. One of the methods is putting pins around the glands and around the interest area, then giving the image to a software for level determination. This process is time taking and requiring effort because the number of pins are many. Another method is using image processing softwares to apply filters on the image in order to get MGD area and pixels. In this thesis, a different method is used. Meibography images are processed using increasing & decreasing color valuess and no filter applied to images.

One of the current methods is putting pins around glands and the interest area. In this technique, the physician takes the meibography image of inner part of eyelids of patient to check for existence of MGD and the level of the disease. The physician puts pins around glands and around the interest area. Then the image is processed by the software which draws MGD and NO-MGD pixels with different colors and returns level of the disease. Sofware uses four level metric for deciding the level.

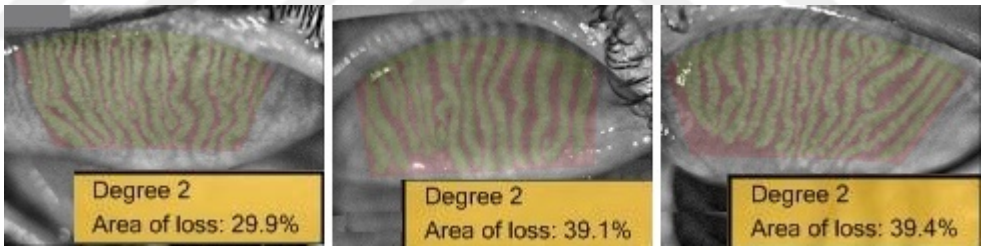

Figure 2.1 Example images from the current method

Although the current method is giving a trusted result to the physician about the level of the disease, manually putting pins around glands and interest area is time taking. Another disadvantage is that there is no undo for a pin after a new pin.

This thesis is discussing the algorithm for developing a software in order to determine MGD level from meibography images. Comparing with the current method, the physician is going to draw only interest area of the image. After drawing, software of this thesis is designed to convert this area into a two-color structure and give level of the disease as output. Software will use four level metric for deciding the level. Also, the physician can control whole process and edit the results.

Another current method for determination of MGD level is using image filters. In this technique, MGD level is determined by applying filters on the image [1]

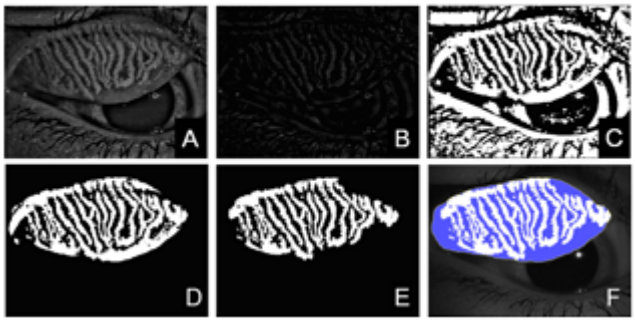

Figure 2.2 Example images from the method [1]

Although this technique determines MGD pixels as shown in Image F., along with other preparations applying filters mentioned in the following list will require time to run the algorithm.

- Wallis Filter:

  The Wallis Filter applies a contrast enhancement to the image in which there are significant areas of bright and dark tones. The Filter adjusts brightness in local areas so that the local mean and standard deviation match target values
- Labelling Process:

  Grouping pixels on the image according to coordinate and color relation. Pixels are grouped to form a component that can be used independent from others.
- Edge Detection:

  Determination of pixels that forms an edge between two components or regions.
- Spline Interpolation:

  A different form of curve-fitting and is similar to linear interpolation.
- Gamma Correction:

  Gamma functions are used for correcting the image's brightness level(luminance).
- Mapping:

  Mapping is transforming pixels of a image into new pixels. Thus transforming the image. Greyscale image can be converted to RGB image using mapping.
- Excluding Misdetermination:

  This method is used to correct pixels that are misdetermined.

2.4 Treatment of MGD

There are traditional methods used for treatment of the disease such as warm compress, lid hygiene, antibiotics and anti-inflammatory agents [3]. There are other methods for the treatment of the disease. These methods [3] can be listed like,

- Intraductal meibomian gland probing
- Emulsion eye drops containing lipids
- Thermal pulsation system
- N-asetil-cysteine
- Topical Azithromycin
- Cyclosporine A

Treatment of the disease is still challenging. Although MGD can be determined by looking at the meibography images, there are some cases that the disease can not be determined. This type of condition is called as "Non-obvious MGD".

There is a case report that is taken from reference 2, A 42 years old female claimed dry eye. Symptoms were assessed by the OSDI (Ocular Surface Disease Index). OSDI score was 37.5. Lid margin and meibomian gland appeared to be normal. Meibomian glands were not expressible easily. Loss of meibomian glands was evaluated by the portable non-contact infra-red (IR) meibograph and computerized grading and resulted in 37% (OD) loss of glands of the upper lid.

Condition of upper and lower eye lids before and after the treatment is listed in Table 1.2. Loss of glands (Area of Loss) is represented with percentages.

	Upper Lid	Lower Lid
Before Treatment	OD: 37% OS: 42%	OD: 45% OS: 51%
After Treatment	OD: 40% OS: 39%	OD: 48% OS: 45%

Table 2.1 Area of loss of the upper and lower lids before and after treatment.

3. AUTOMATIC DETECTION OF MGD LEVEL

In the development of the software of this thesis, PHP programming language and GD image processing library are used in server. Javascript programming language is used for processing the image on the web browser and developing the software interface. The algorithm mentioned in this thesis is from the server side (PHP) code of the software. Algorithm is represented as pseudocode.

PHP is a popular, general-purpose scripting language that is especially suited to web development as php.net describing the language. For years, development with PHP became more popular. Object oriented programming support and the latest version of the language which is version 7 made the language more preferable. Gif Draw library of PHP programming language is the built-in image processing library which is used and tested by many developers. The library provides all needed functions to the developer. Documentation of PHP programming language and GIF Draw library is well prepared and has a lot of example of usage.

Javascript is a programming language which is object-oriented and used for making interactive effects within a web browser. Javascript code is running on the browser. The latest developments are making the language more powerful and preferable. NodeJS is a server side programming language based on Javascript. It would be fast and efficient to process images using NodeJS. In this thesis, PHP is preferred.

Processing with PHP has more advantages than processing with Javascript. PHP is running in the server while Javascript is on the web browser. PHP is using the content of the file while Javascript is using the canvas element for processing the image. For this reason, in this thesis, images are sent to server when possible.

3.1 Image Properties

A meibography image is the infrared photograph of the inner part of the eyelid of the patient. The output images from meibography are the inputs for the software of this thesis. Extension of files is "jpeg". PHP Gif Draw image processing library is able to open, read and understand the image files with a valid image file extension. Openning, reading and writing "jpeg" files is possible.

Images from meibography have "640px" width and "480px" height. Figure 3.1 is the example of meibography images which are scaled to 25%.

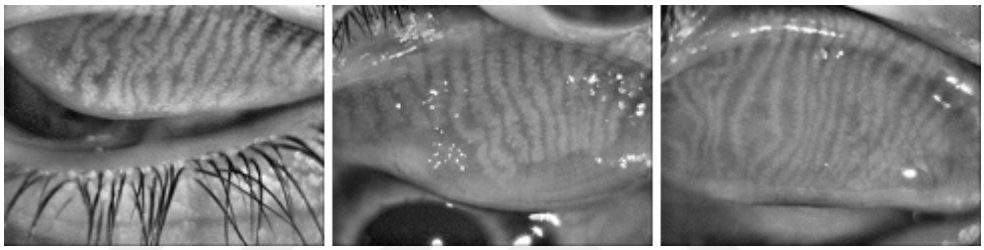

Figure 3.1 Meibography images

Meibography images are greyscale. The characteristic of greyscale images is that Red, Green and Blue values of each pixel of the image are equal.

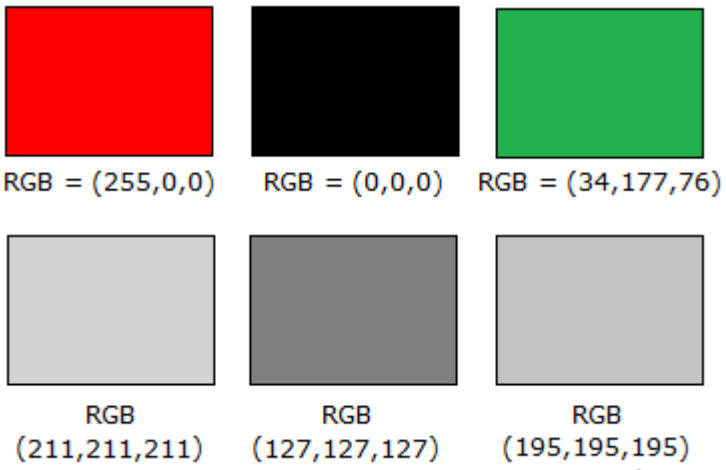

Figure 3.2 Example colors

In Figure 2.3, there are six colors. Red and Green rectangles (First and Third) is not greyscale because of the different values of Red, Green and Blue colors. Black and grey rectangles (Second, fourth, fifth, sixth) are greyscale.

In software or web development, there are mainly two notations that are used in common. One of the notations is HEX. Another one is RGB. Both notations are used for representing a color. Development environments are providing some visual elements to the developer. Using the elements, developers compose a software or a website. There are many visual products that are developed until now. Each product is including visual elements in different colors. Element colors are selected using the interface of the environment or by writing some line of code. All programming environments accept both HEX and RGB notation.

HEX notation represents each of Red, Green and Blue values with two bytes in base sixteen. Length of HEX notation is six. First two bytes represent Red while the second two bytes is value of Green. In a similar way, the third two bytes represent Blue color. Developers put a number sign "#" in front of the notation to tell the compiler or programming environment that the notation is a HEX notation. This notation is accepted by all programming languages and HTML. Examples of HEX notation are: #FF0000, #00FF00, #0000FF (Red, Green, Blue)

RGB notation represents Red, Green and Blue values of a color with three numbers from 0 to 255. First number is representing the value of Red. Second and the third numbers are representing Green and Blue color. Developers write RGB notation with using "rgb" keyword to tell the type of notation. RGB like HEX is accepted and used by all programming languages. In addition to this, there is another notation which is called as RGBA. The fourth value (A) represents alpha channel of color. Alpha channels are between 0 and 1, used for making transparent color. Examples of RGB notation are: rgb(255,0,0), rgb(0,255,0), rgb(0,0,255) (Red, Green, Blue)

Meibography images are greyscale. In HEX notation, value of each two bytes are equal. In RGB notation, value of each three numbers are equal. While developing the software of this thesis, RGB notation is used when processing image pixels.
Example HEX notation for a greyscale color: #8A8A8A
Example RGB notation for a greyscale color: rgb(138,138,138)

3.2 Image Processing

Image Processing is a set of activities in order to get information about or change a digital image. Image files have a unique code representation that describing the image to the computer. In addition to this, programming languages are able to read and understand this representation. For this reason, reading and changing the image without seeing the visual representation is possible. Following list is representing some of image processing functions of PHP programming language.

- imagecreatefromstring:
  Forming a "GD image" object from base64 representation of the image.
- imagesx:
  Returning the width of the image.
- imagesy:
  Returning the height of the image.
- imagecolorat:
  Return is the color value of pixel at given left and top coordinate.
- imagecolorallocate:
  Function preparing a color for drawing a pixel.
- imagesetpixel:
  Function that drawing a pixel with the color from imagecolorallocate
- imagepng:
  Converting a "GD image" object to a PNG Image

Programming languages are useful for image processing. Programs able to process images can be written. All popular programming languages have image processing libraries provided to developers. Pre-defined functions and classes of the language are helping developers open, read, write and save images. In this thesis, PHP and Javascript programming languages are used for image processing. A big percentage of the processing work is done by PHP in the server side. This is for the reason that the performance advantage of PHP programming language.

Images from meibography have "jpeg" extension. Gif Draw image processing library is able to open, read and change "jpeg" files. After openning the file, developer can write code for reading from or writing on the image. Pseudocode for traversing each pixel of the image can be written as following:

```
FOREACH ROW AS M:
   FOREACH COLUMN AS C:
       COLOR = GET_COLOR_AT(C, M);
```

The code is reading each pixel of the image. There will be (width * height) 640 * 460 = 307200 "Get Color at" function call by methods. For this reason, performance of the methods must be considered and tested in this kind of condition.

Color of pixels of the image can be useful for image processing. Color of each column on a row can be drawn onto a line chart to understand the color relation.

```
FOREACH COLUMN AS C:
     COLOR = GET_COLOR_AT(C, 138);
     DRAW_ON_CHART(COLOR, C, 138)
```

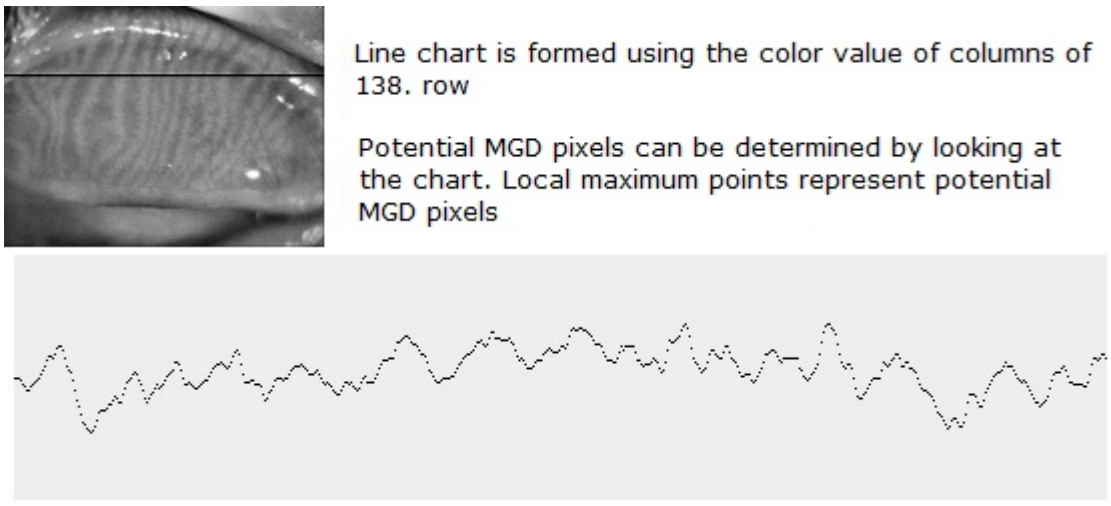

Figure 3.3 Line chart representation of 138. row of a meibography image

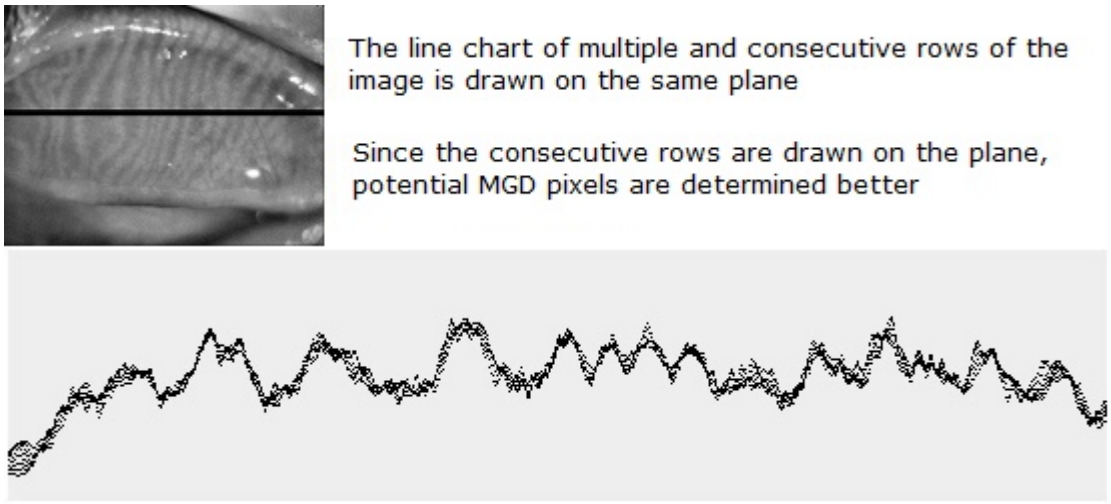

Figure 3.4 Line chart representation of multiple rows of a meibography image

Figure 3.3 and Figure 3.4 are showing line chart representation of some rows of a meibography image. Horizontal line of the charts is representing the columns while the vertical line is representing color values. When the line chart of a row is drawn, potential MGD pixels can be seen by looking at the chart. If color values are increasing and then decreasing, the pixels can be thought as a potential MGD pixel set. Increasing and decreasing values mean that there is a brightness. For this thesis, brightness on a meibography image means a potential MGD pixel set.

There is one more example showing that in some rows, there may not be MGD pixel sets although there are increasing and decreasing color values on the row.

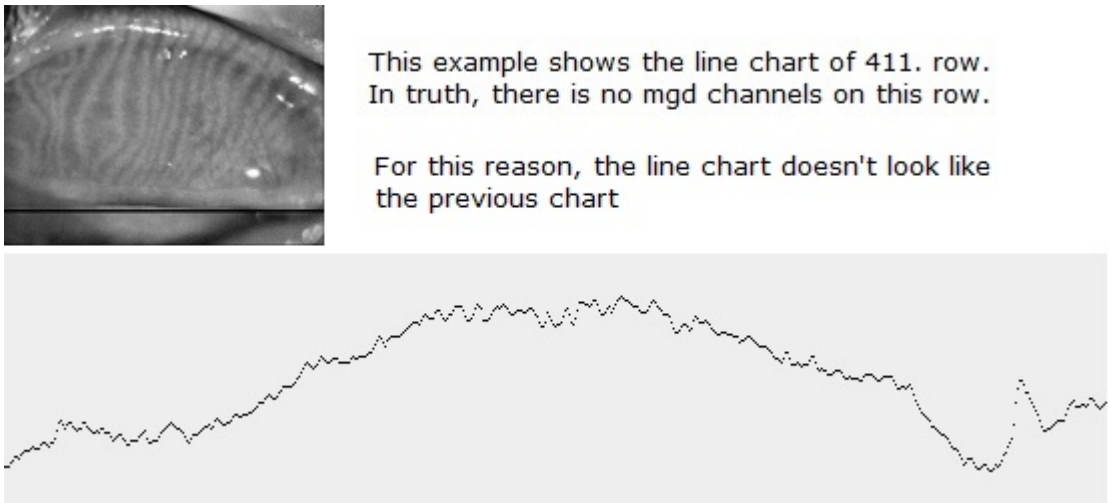

Figure 3.5 Line chart representation of 411. row.

Line chart representations for some rows are telling that potential MGD pixels can be determined using line chart of the row. Potential MGD pixels are brightness on the image. At the same time, are inside increasing & decreasing pixel sets. By looking at the line chart, pixels between a local minimum and consecutive local minimum represent brightness and are potential MGD pixels.

In this thesis, Determination of MGD level using meibography images is based on line charts. Before ending this section, pseudocode showing that the meibography images are greyscale can be written as the following code block. The code doesn't give error on meibography images.

```
FOREACH ROW AS M:
   FOREACH COLUMN AS C:
       COLOR = GET_COLOR_AT(C, M);
       IF (COLOR["R"] == COLOR["G"] AND COLOR["R"] == COLOR["B"]):
           CONTINUE;
       ELSE:
           ERROR("IMAGE IS NOT GREYSCALE");
```

Pseudocode above is traversing each column of each row of the image using the FOREACH COLUMN inside FOREACH ROW. Color of each pixel of the image at the coordinate from (0, 0) to (640, 480) is checked. If Red value is equal to Green and Blue, the pixel can be a part of a greyscale image. Characteristic of a greyscale image is that Red, Green and Blue values of each pixel of the image are equal. For this reason, code does not give error and continue with the new pixel. If Red, Green and Blue values are not equal then the code give error. Because the pixel can not be a part of a greyscale image. If so, the image is not greyscale.

Code will run for each pixel if the image is greyscale. Code will give error whenever a not greyscale pixel is read. Using the code, program for checking images in terms of greyscale or not can be developed.

3.3 Algorithm for Detection of MGD Level

In this section, methods for automatic detection of MGD level is presented. From a meibography image to a processed image of which level is determined, all phases used are mentioned. The first phase is drawing the interest area. Phases end with determining MGD level and displaying the results to the physician.

Phases from drawing the interest area to the result can be listed as following:

- Determination of Interest Area

  Interest area is the region that the algorithm will be aplied. Interest area is drawn by the physician and determined by the software
- Drawing Line Chart

  Forming line chart representation of each row to determine potential MGD pixels. Line charts are imagined by the algorithm instead of drawing.
- Finding Local Maximum, Local Minimum

  Checking brightness on the image to find frames for potential MGD pixels. Pixels between consecutive local minimum points of the row are behaved as potential MGD pixels
- Gathering Frames

  Frames from all rows are gathered for processing. Each frame is containing potential MGD pixels between consecutive local minimum points
- Deciding MGD Pixels from Frames

  From the frames, MGD pixels are determined using the color distance of a pixel to local minimum and to local maximum.

After passing through these phases, the input image becomes the result image of which MGD pixels are determined. The algorithm focuses on the interest area and form the frames that have potential MGD pixels inside. The algorithm detects MGD pixels by using color distance. After detecting MGD pixels, Area of Loss is calculated using the ratio returned from the software. According to Area of Loss, level of the disease is calculated using the four level metric.

3.3.1 Determination of interest area

In this section, the method for determining the interest area in server is presented. Before that, feature of drawing the interest area is mentioned. Following images are examples of the feature.

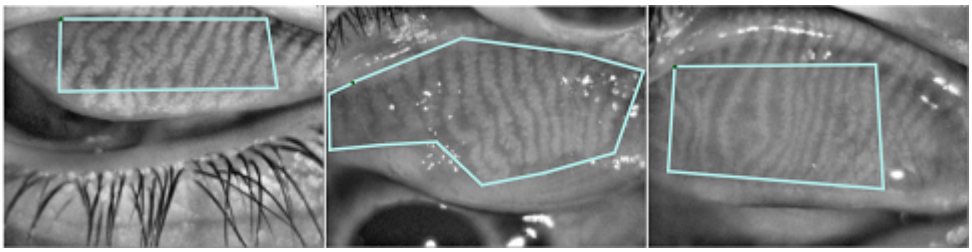

Figure 3.6 Example images for drawing the interest area.

Workflow description of the application,can be usable to understand the feature of drawing the interest area, which can be listed as following.

- A meibography image is uploaded:
  After taking the IR photograph of the inner part of eyelid of the patient, output image is uploaded to the software.
- Interest area is drawn:
  After the image is uploaded, the interest area is drawn by the physician using the tool in the frontend of the software. The interest area means a closed area on the image which the algorithm will be applied on.
- "Process" button is clicked:
  Drawing the interest area is completed when the starting and the ending point of drawing line is connected. Then, the physician clicks to "Process" button.
- Result is shown:
  In about five seconds, the software is giving a result which shows MGD pixels and NO-MGD pixels, along with the number and ratio of the pixels.
- Result is edited:
  Result image is presented in the HTML document using a canvas element along with a tool that is enabling the editing operation on the image. The doctor can change MGD and NO-MGD pixels on the image using the editing tool.

When starting and ending point of drawing line is connected, "Process" button becomes visible to the physician to send the image to server for determining the interest area. The area is determined using the following code.

```
FOREACH COLUMN AS C:

   FOREACH ROW AS M FROM ROW0 TO ROW479:
       COLOR = GET_COLOR_AT(C,M);
       IF (GREYSCALE): PAINT_WITH_RED(C,M);
       ELSE: BREAK;

   FOREACH ROW AS M FROM ROW479 TO ROW0:
       COLOR = GET_COLOR_AT(C,M);
       IF (GREYSCALE): PAINT_WITH_RED(C,M);
       ELSE: BREAK;

FOREACH ROW AS M:

   FOREACH COLUMN AS C FROM COLUMN0 TO COLUMN639:
       COLOR = GET_COLOR_AT(C,M);
       IF (GREYSCALE): PAINT_WITH_RED(C,M);
       ELSEIF (CLARET_RED): CONTINUE;
       ELSE: BREAK;

   FOREACH COLUMN AS C FROM COLUMN639 TO COLUMN0:
       COLOR = GET_COLOR_AT(C,M);
       IF (GREYSCALE): PAINT_WITH_RED(C,M);
       ELSEIF (CLARET_RED): CONTINUE;
       ELSE: BREAK;
```

The method is traversing image pixels from four different direction.

The method can be represented with the following list.

- For each column of the image
  - From Up to Bottom, check the color value of each column of the row
    - If color is greyscale, then paint with a color (Claret red)
    - If color is not greyscale, then break the loop
  - From Bottom to Up, check the color value of each column of the row
    - If color is greyscale, then paint with a color (Claret red)
    - If color is not greyscale, then break the loop
- For each row of the image
  - From Left to Right, check the color value of each column of the row
    - If color is greyscale, then paint with a color (Claret red)
    - If color is claret red, then continue
    - If none of the above, then break the loop
  - From Right to Left, check the color value of each column of the row
    - If color is greyscale, then paint with a color (Claret red)
    - If color is claret red, then continue
    - If none of the above, then break the loop

Output the method mentioned above is as following.

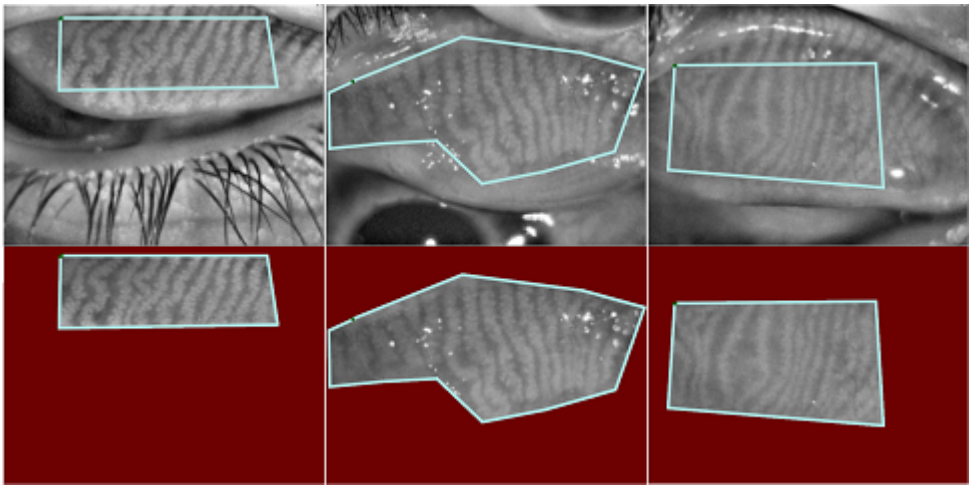

Figure 3.7 Input and Output images of the method

According to the output, the method determines the interest area of images.

3.3.2 Drawing line chart

After the interest area of the image is determined, the output image from the method for determination of the interest area will be sent to server again for final processing. Images are in the following format.

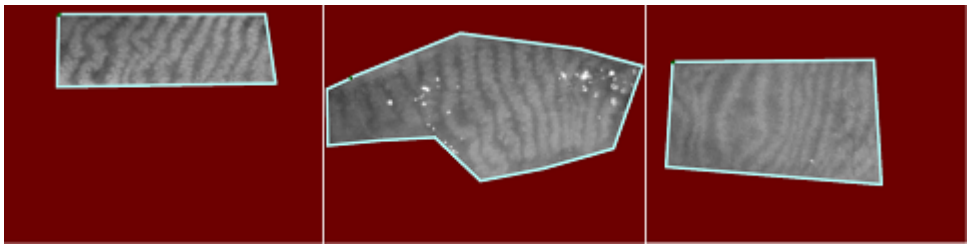

Figure 3.8 Example of images that are inputs for final processing

After the image is sent to server, server starts to process the image. The method for this purpose starts from ROW 0 (First row) of the image, for each column of the row, color value of the column is calculated and drawn on the chart of the row. This process is repeated for each row of the image. This way, the line chart representations of each row are formed.

There will be 480 (height of image) line charts and each chart will represent a row from ROW0 to ROW479. The method can be presented with the following code.

```
FOREACH ROW AS M:
      FOREACH COLUMN AS C:
            COLOR = GET_COLOR_AT(C,M);
            IF (COLOR["R"] == COLOR["G"] AND COLOR["R"] == COLOR["B"]):
                        DRAW_ON_CHART(COLOR,C,M);
```

Method is traversing each column of each row of the image and drawing the color value of the current column on the line chart of the row. In addition, the method is checking the color value of the current column and if color is not greyscale then does not draw the column on the chart. This way, only the interest area of the image is processed and drawn on the line chart representation of the row.

Example output of this method is as following.

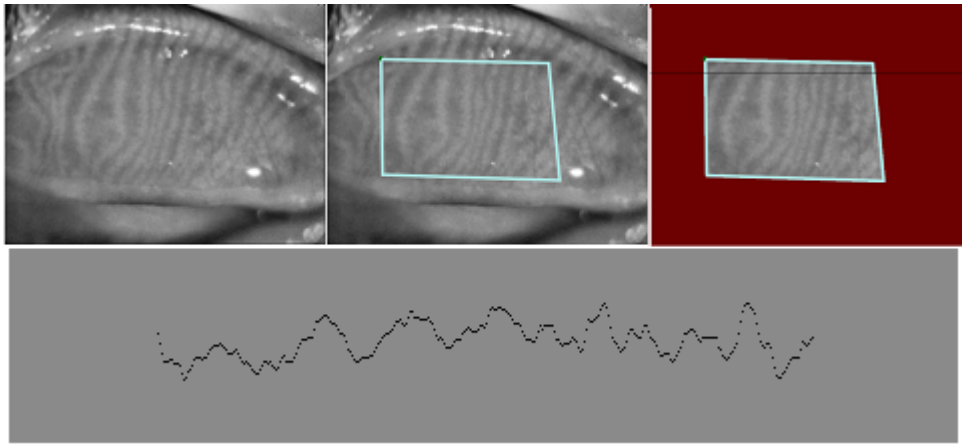

Figure 3.9 Line chart of 138. row of the example image

Another example can be represented as following

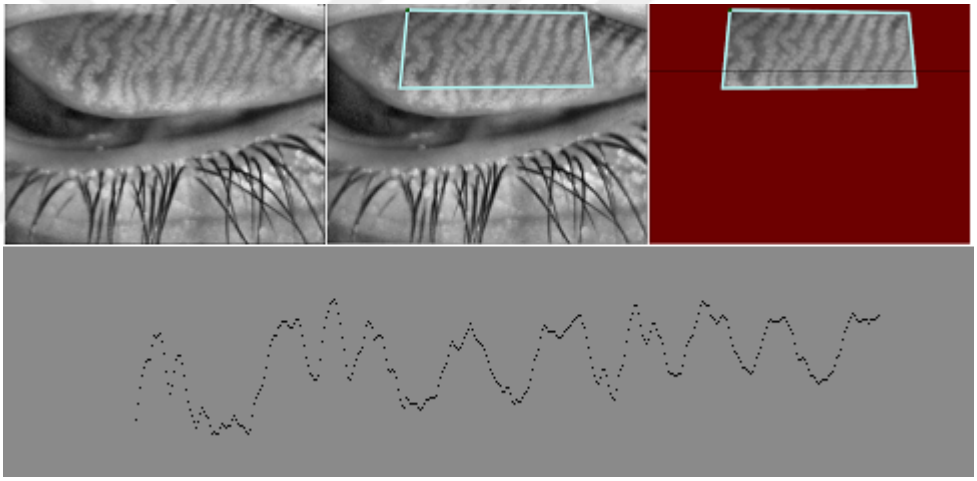

Figure 3.10 Line chart of 138. row of another example image

Figure 3.9 and Figure 3.10 are examples of "Drawing Line chart" method running for 138. row of example images. MGD color sets can be seen by looking at the increasing and decreasing points on the line chart.

Purpose is getting increasing and decreasing pixels in the interest area, keeping in frames and deciding the MGD pixels.

The blank spaces on the left and on the right side of the chart are the areas colored by "Claret Red". The method is focusing on the interest area.

When the following line chart is considered

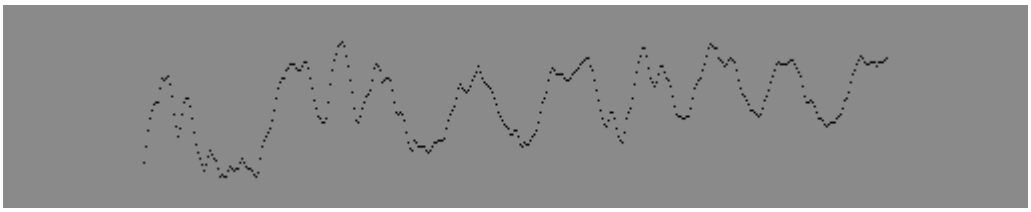

Figure 3.11 Example Line Chart

Pixels which will be used for MGD determination can be marked as following.

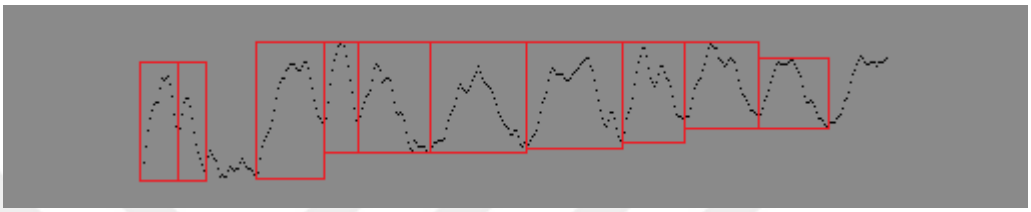

Figure 3.12 Marked pixel sets on Example Line Chart

Pixels (color value of a column) inside the red rectangles are potential MGD pixels. If the color values while going from left to right is increasing then decreasing, that means there is a brightness and the pixels are potential MGD pixels. Marking these pixels on the line chart is simple but for the software, method for getting potential MGD pixels and keeping in a frame is needed. The description for the development of this kind of a method is mentioned in following sections.

Determination of MGD pixels can be done by finding local maximum and local minimum points on the line chart and keeping in a frame.

Frame word means the array containing three elements as following.

```
[ [min_value, index], [max_value,index], [min_value, index] ]
```

Local Minimum	Minimum point of MGD pixel set (frame) on the left
Local Maximum	Maximum point of MGD pixel set (frame)
Local Minimum	Minimum point of MGD pixel set (frame) on the right

Table 3.1 Content of a regular frame

3.3.3 Finding local maximum, local minimum

A local maximum can be described as a point on the graph of the function whose vertical coordinate is larger than all other vertical coordinates on the graph at points "close to" the local maximum point. Local minimum is a point on the graph of the function whose vertical coordinate is less than all other vertical coordinates on the graph at points "close to" the local minimum point[8].

Example of local minimum and local maximum points on a graph can be presented as following figure. The graph is the line chart representation of a row of a meibography image. Software is not generating visual line chart representation of each row of the image. Instead, is behaving the pixels like drawing a line chart while traversing the image pixels.

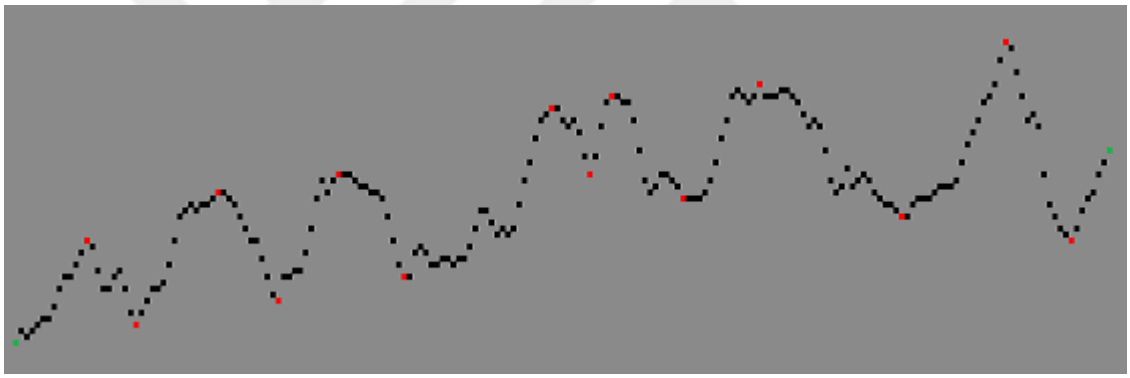

Figure 3.13 Local Maximum and Local Minimum points on a line chart

Local maximum & local minimum points are represented with red color, starting and ending points are represented with green color. While the method is traversing the column pixels of a row of the image, visual representation in Figure 2.13 is imagined by the method and local maximum & local minimum points are determined.

The workflow up to this time can be listed as:

- Physician loaded the image and the interest area was drawn
- Image was processed in server for determining the interest area
- Image was sent back to browser for displaying
- Image is sent to server for final processing

The method in the server side is getting a meibography image that the interest area is determined and is using the code presented as following pseudocode to determine MGD pixel sets (frames). The method is assumed to work on 138. row

```
ROW=138; VAR=10; LEFT_OK=0; RIGHT_OK=0; MINIMUM_UNTIL_NOW=[];
FOREACH COLUMN AS C:
   CURRENT_COLOR = GET_COLOR_AT(C, 138);
   IF (MINIMUM_TIL_NOW == [] OR CURRENT_COLOR < MINIMUM_TIL_NOW[0]):
      MINIMUM_TIL_NOW = [CURRENT_COLOR, C];

   WHILE (THERE IS A PIXEL ON THE LEFT):
    GO_LEFT();
    LEFT_CURRENT_COLOR = GET_COLOR_AT(CURRENT_PIXEL_ON_LEFT, 138);
    IF (CURRENT_COLOR - LEFT_CURRENT_COLOR > VAR):
       LEFT_OK = 1;
       BREAK;
    IF (CURRENT_COLOR < LEFT_CURRENT_COLOR):
        BREAK;

   WHILE (THERE IS A PIXEL ON THE RIGHT):
    GO_RIGHT();
    RIGHT_CURRENT_COLOR = GET_COLOR_AT(CURRENT_PIXEL_ON_RIGHT, 138);
    IF (CURRENT_COLOR - RIGHT_CURRENT_COLOR > VAR):
       RIGHT_OK = 1;
       BREAK;
    IF (CURRENT_COLOR < RIGHT_CURRENT_COLOR):
       BREAK;

    IF (LEFT_OK == 1 AND RIGHT_OK == 1):
       ADD_TO_FRAME(MINIMUM_UNTIL_NOW, C, 138)
```

Method is traversing each column of each row. For each column of the current row, the method is checking the color value of pixels (columns) on the left and on the right of the current column. For columns on the left, if there is a color value that is less than the current color and the difference is more than "VAR" variable(10), the method is breaking the loop and continue with the columns on the right. In a similar way, for columns on the right, if there is a color value that is less than the current color and the difference is more than "VAR" variable(10), then the current point is accepted as local maximum point.

While the method is checking the color value of the previous and the next columns of the current column, at the same time, is keeping the minimum color value seen until that time. If RIGHT OK variable is 1 and LEFT OK variable is 1, then the column along with the minimum point are sent to "ADD TO FRAME" function

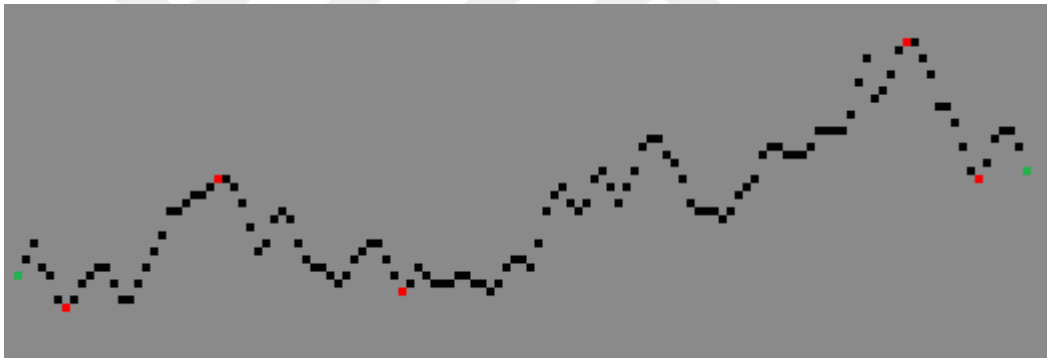

Figure 3.14 Output of Finding Local Maximum, local minimum method

Line chart in Figure 3.14 is the example output of the method for finding local maximum & local minimum point. Red marking is showing local maximum and minimum point while the green marking is showing the starting and the ending point. The gap between the third and the fourth red point is telling that there is no local maximum between the third and the fourth point. This condition is because of the VAR(10) variable of the method. Value of the variable is set to ten by default. Value can be changed to set the precision of the algorithm and the software. If the value is less than ten, there will be more frames gathered, in a similar way, if the value is more than ten, there will be less frames. This condition is telling that VAR variable is setting the precision of the algorithm and the software.

3.3.4 Collecting the frames

Frame word is used for describing the array containing potential MGD pixels. When the framing function is called by local maximum & local minimum method, the minimum point and the maximum point are kept in the array which is called as a frame. In addition, the minimum value is added to the previous frame. This way, each frame will contain three values that are describing the left local minimum point, local maximum point and the right local minimum point.

In this thesis, a frame is independent from other frames. Left local minimum value is representing the starting point and the starting color value of the frame. Local maximum value is representing the maximum color point and the maximum color value of the frame. In a similar way, the right local minimum value is representing the ending point and the ending color value of the frame.

Expectation is to have three different point in a frame. Method for finding local maximum and local minimum point is checking each column and looking for a maximum value using the VAR(10) variable and adding the maximum point to a new frame with the local minimum point until that time. The local minimum point is also added to the previous frame.

There are some conditions that a frame may not contain three points. Instead, there may be two points that are representing the starting and the ending point of the frame, at the same time, the line chart. This condition means that there is no local maximum point on the chart according to the method.

Line charts are showing that there are pixels between the starting point and the first left local minimum point and also there are pixels between the last right minimum point and the ending point. Although the expectation is a frame with three elements, some frames may have two elements. The frames with less than three elements will be determined in a different way. Because, a regular frame has three elements and one local maximum. Frame with two elements has no local maximum.

When the following figure is considered,

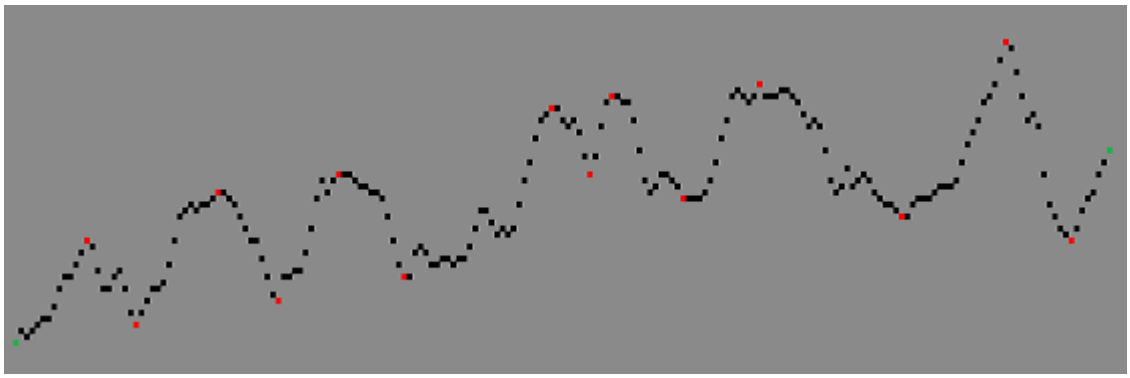

Figure 3.15 Local Maximum and Local Minimum

The local maximum and the local minimum points are marked with red color while the starting and the ending point is marked with a green color. There are seven local maximum, seven local minimum points on the example line chart. Each local maximum point has one local minimum on the left and one local minimum on the right. The starting point marked with green color is the left local minimum point of the first frame. However, the ending point is not the right local minimum point of the last frame. The starting and the ending points may or may not be local minimum points for a frame.

When following figure is considered,

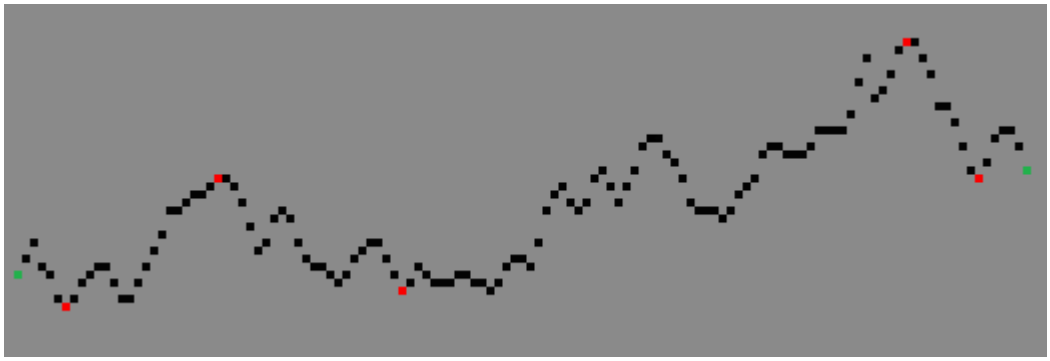

Figure 3.16 Example of Local Maximum and Local Minimum

Starting and ending points are not in the first or the last frame of the line chart. In this kind of condition, the remaining pixels between the starting point and left local minimum of the first frame and between right local minimum of the last frame and the ending point will be behaved different while deciding MGD pixels.

There are four types of frames in total. Combination of frames of a row is the total representation of the row. For this reason, each type of a frame of a row must be processed in order to check all pixels of the row.

```
[[left_minimum,index], [maximum,index], [right_minimum,index]]
```

```
OR
```

```
[[starting point,index], [left local minimum of first frame,index]]
```

```
OR
```

```
[[right local minimum of last frame,index] , [ending point,index]]
```

```
OR
```

```
[[starting point,index], [ending point,index]]
```

As a result, There may be different kind of frames that coming from the framing method. When a frame is thought in the abstract way, each frame is containing the corresponding column pixels of the row. According to the color value of pixels in the frame, the method in the following section will determine MGD pixels looking at the distance of color value of a column to the local minimum and local maximum for a regular frame. For another kind of a frame, a different method will be applied. Following table is describing a regular frame.

Local Minimum	Minimum point of MGD pixel set (frame) on the left
Local Maximum	Maximum point of MGD pixel set (frame)
Local Minimum	Minimum point of MGD pixel set (frame) on the right

Table 3.2 Content of a regular frame

3.3.5 Determination of mgd pixels from frames

After gathering frames, each row will have frames containing column pixels of the row according to local maximum and minimum rule. In the previous section, different kind of frames are mentioned. A regular frame looks like following,

```
[ [left_minimum, index], [maximum, index], [right_minimum, index] ]
```

The frame above is representing that there are three points in the frame: left minimum, local maximum, right minimum. The purpose is traversing each pixel in this frame from left to right and decide if the pixel is MGD pixel.

From left minimum to the maximum, for each pixel, if color distance of the pixel to the left minimum is less than the distance to the local maximum, then the pixel is not representing MGD. If distance to left minimum is greater than the distance to the local maximum, then the pixel is representing MGD.

Following code is representing this method.

```
FOREACH COLUMN AS C FROM LEFT_MINUMUM TO LOCAL_MAXIMUM:
   CURRENT_TO_MIN = CURRENT_COLUMN["COLOR"] - LEFT_MINIMUM["COLOR"]
   MAX_TO_CURRENT = LOCAL_MAXIMUM["COLOR"] - CURRENT_COLUMN["COLOR"]
   IF (CURRENT_TO_MIN > MAX_TO_CURRENT):
       DRAW_WITH_GREEN_COLOR(C, {CURRENT_ROW})

FOREACH COLUMN AS C FROM LOCAL_MAXIMUM TO RIGHT_MINUMUM:
   CURRENT_TO_MIN = CURRENT_COLUMN["COLOR"] - RIGHT_MINIMUM["COLOR"]
   MAX_TO_CURRENT = LOCAL_MAXIMUM["COLOR"] - CURRENT_COLUMN["COLOR"]
   IF (CURRENT_TO_MIN > MAX_TO_CURRENT):
      DRAW_WITH_GREEN_COLOR(C, {CURRENT_ROW})
```

Another kind of a frame can be like following,

```
[[starting point,index],[left local minimum of first frame,index]]
```

In this case, there are two elements inside the frame. One of the elements are representing the starting point of the line chart while the another is representing the left local minimum value of the first frame. The previous method is not applicable on this kind of frames because there is no local maximum value.

Following code is representing the method for processing this kind of a frame.

```
VAR      = 10
MAXIMUM  = {MAXIMUM OF THE FRAME}
MINIMUM  = {MINIMUM OF THE FRAME}
FOREACH COLUMN AS C
   FROM STARTING_POINT TO LEFT_LOCAL_MINIMUM_OF_FIRST_FRAME:
      CURRENT_TO_MIN = CURRENT_COLUMN["COLOR"] - MINIMUM
      MAX_TO_CURRENT =  MAXIMUM - CURRENT_COLUMN["COLOR"]
      IF (CURRENT_TO_MIN > MAX_TO_CURRENT):
         IF (CURRENT_TO_MIN > VAR):
            DRAW_WITH_GREEN_COLOR(C, {CURRENT_ROW})
```

The method knows the maximum and the minimum value of the frame. There are pixels in the frame from starting point of the line chart to the left local minimum of the first frame. For each pixel in the frame, the method is checking the color distance of the pixel to the minimum and to the maximum. Method determines the pixel as MGD if color distance of the pixel to the maximum is less than to the minimum and if the distance to the minimum is more than VAR(10) variable.

Difference from the previous method is that the current method knows maximum and minimum values of the frame in advance.

As different from the previous, a frame can be as following.

```
[[right local minimum of last frame, index],[ending point, index]]
```

In this case, one of the elements of the frame represents the right local minimum of the last frame while the another represents the ending point of the line chart. The method for this kind of a frame is as following.

```
VAR      = 10
MAXIMUM  = {MAXIMUM OF THE FRAME}
MINIMUM  = {MINIMUM OF THE FRAME}
FOREACH COLUMN AS C
   FROM RIGHT_LOCAL_MINIMUM_OF_LAST_FRAME TO ENDING_POINT:
      CURRENT_TO_MIN = CURRENT_COLUMN["COLOR"] - MINIMUM
      MAX_TO_CURRENT =  MAXIMUM - CURRENT_COLUMN["COLOR"]
      IF (CURRENT_TO_MIN > MAX_TO_CURRENT):
            IF (CURRENT_TO_MIN > VAR):
                  DRAW_WITH_GREEN_COLOR(C, {CURRENT_ROW})
```

In addition to the previous kind of frames, a frame can be as following.

```
[ [starting point, index], [ending point, index] ]
```

This condition is telling that there is one frame in the row. In this case, the same method mentioned above can be used by changing,

```
FOREACH COLUMN AS C
        FROM RIGHT_LOCAL_MINIMUM_OF_LAST_FRAME TO ENDING_POINT:
with
FOREACH COLUMN AS C FROM STARTING POINT TO ENDING POINT:
```

After all frames from each row are gathered, frames are processed as independent from other frames by the methods mentioned above according to type of the frame. After processing, MGD pixels are determined and drawn with green color. Other pixels inside the interest area is "Area of Loss".

Example output of determination of MGD pixels method is as following.

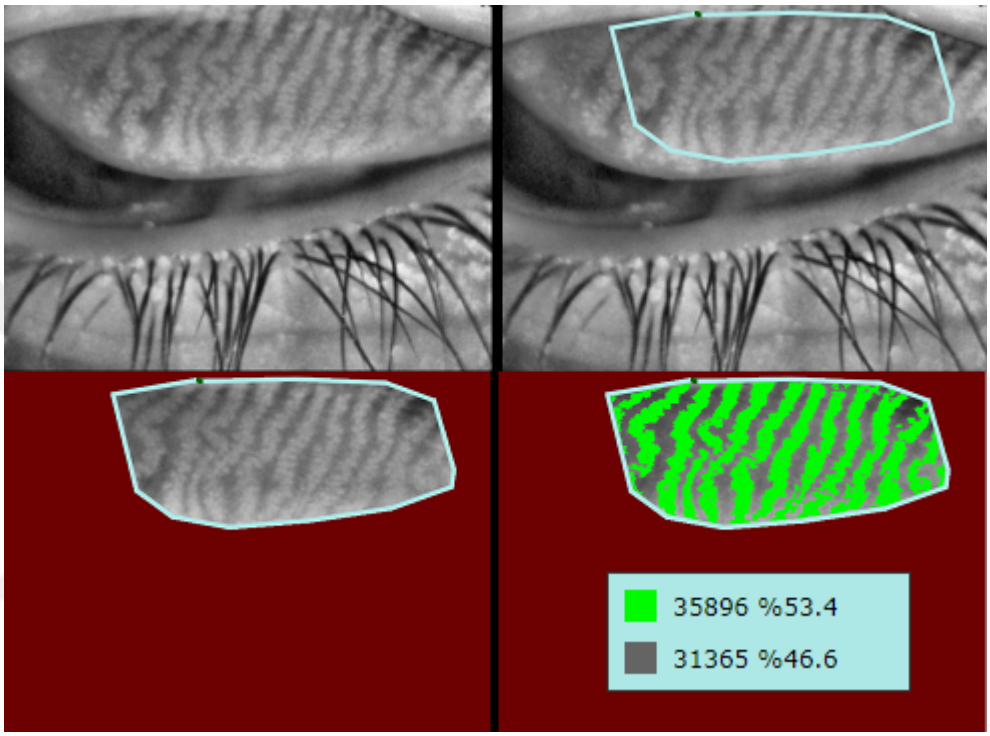

Figure 3.17 Example of determination of MGD pixels

According to result, percentage of MGD pixels inside the interest area is 53.4%. Area of loss from the result is 46.6%. Using Area of loss, MGD level is calculated.

```
0  <= Level 1 < 25
25 <= Level 2 < 50
50 <= Level 3 < 75
75 <= Level 4 < 100
```

According to metric above, the result for the MGD level seems to be "Level 2". Four level metric for detection of MGD level is implemented to the software.

3.4 Revision on the Algorithm

In the previous sections, methods for determination of the interest area and MGD pixels are presented. The algorithm for determination of the MGD level focuses on the interest area of the meibography image and draw the line chart of each row. Drawing line chart, in this thesis, is the abstract operation done by the algorithm. Algorithm is imagining the line chart representation of the row while traversing the row and is marking local maximum and minimum points. After finding local maximum and minimum points, frames are formed. A regular frame is the array keeping one local minimum on the left, one local maximum and one local minimum on the right. Using the frames, the method for determination of MGD pixels is processing the frames and drawing MGD pixels with green color. Methods mentioned in this thesis is resulting with a software.

Following figure is the output of the software.

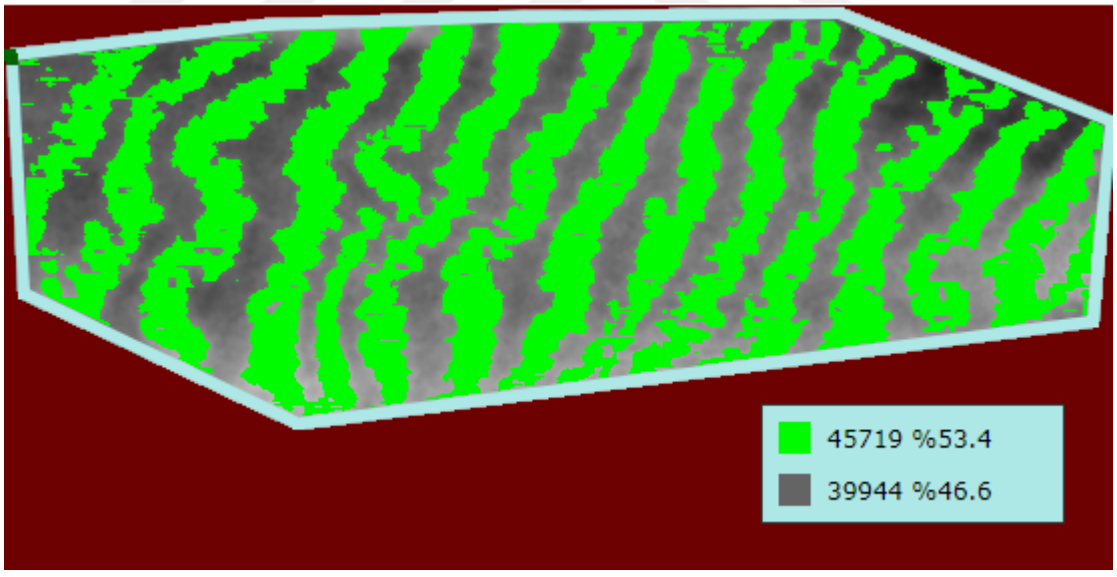

Figure 3.18 Example output of the software

Figure is showing that MGD pixels inside the interest area of the meibography image is determined by the software. The count and the ratio of MGD and NO-MGD pixels are returned from the software as the output. After results are taken, some revisions are needed on the algorithm to make results better. In following, methods for revisions will be developed and used.

Expected revisions on the MGD determination algorithm can be listed as:

Revision: Glands should be wide.
Revision: There should not be a gap inside a gland
Revision: NO-MGD area should not be determined as MGD.
Revision: Brightness on the infrared photograph should be considered.

In order to complete the first revision, method for determination of MGD pixels will be updated. The update should not change the main mechanism of the method. Instead, the update should change just a few line of the code.

For a regular frame, MGD pixels are determined using the following way

```
FOREACH COLUMN AS C FROM LEFT_MINUMUM TO LOCAL_MAXIMUM:
   CURRENT_TO_MIN = CURRENT_COLUMN["COLOR"] - LEFT_MINIMUM["COLOR"]
   MAX_TO_CURRENT = LOCAL_MAXIMUM["COLOR"] - CURRENT_COLUMN["COLOR"]
   IF (CURRENT_TO_MIN > MAX_TO_CURRENT):
      DRAW_WITH_GREEN_COLOR(C, {CURRENT_ROW})

FOREACH COLUMN AS C FROM LOCAL_MAXIMUM TO RIGHT_MINUMUM:
   CURRENT_TO_MIN = CURRENT_COLUMN["COLOR"] - RIGHT_MINIMUM["COLOR"]
   MAX_TO_CURRENT = LOCAL_MAXIMUM["COLOR"] - CURRENT_COLUMN["COLOR"]
   IF (CURRENT_TO_MIN > MAX_TO_CURRENT):
      DRAW_WITH_GREEN_COLOR(C, {CURRENT_ROW})
```

From left local minimum to local maximum and from local maximum to right local minimum, color distance of each pixel in the frame to the minimum and to the maximum is checked. If distance to the maximum is less than the distance to the minimum, then the algorithm is drawing the pixel with green color. This condition can be represented as if the distance to the maximum divided by the distance to the minimum is less than one, the pixel is drawn with green color.

In order to make a gland wide when compared to output of the current method, the following update is added to the code.

```
WIDE_VAR = 0.5
FOREACH COLUMN AS C FROM LEFT_MINUMUM TO LOCAL_MAXIMUM:
   CURRENT_TO_MIN = CURRENT_COLUMN["COLOR"] - LEFT_MINIMUM["COLOR"]
   MAX_TO_CURRENT = LOCAL_MAXIMUM["COLOR"] - CURRENT_COLUMN["COLOR"]
   IF (CURRENT_TO_MIN > MAX_TO_CURRENT OR
       CURRENT_TO_MIN / MAX_TO_CURRENT > WIDE_VAR):
         DRAW_WITH_GREEN_COLOR(C, {CURRENT_ROW})

FOREACH COLUMN AS C FROM LOCAL_MAXIMUM TO RIGHT_MINUMUM:
   CURRENT_TO_MIN = CURRENT_COLUMN["COLOR"] - RIGHT_MINIMUM["COLOR"]
   MAX_TO_CURRENT = LOCAL_MAXIMUM["COLOR"] - CURRENT_COLUMN["COLOR"]
   IF (CURRENT_TO_MIN > MAX_TO_CURRENT  OR
       CURRENT_TO_MIN / MAX_TO_CURRENT > WIDE_VAR):
         DRAW_WITH_GREEN_COLOR(C, {CURRENT_ROW})
```

Update added a new variable to the code. Variable is assigned to the value of ”0.5”. New method again checks the color distance of the pixel to the minimum and to the maximum. If the distance to the minimum is more than the distance to the maximum, the method draws the pixel with green color. In addition, with the new update, algorithm checks the distance ratio of the pixel. For example, if the distance to the minimum is less than or equal to the distance to the maximum, algorithm checks the distance ratio. If the ratio is greater than the value of the variable ”0.5”, the method will determine the pixel as MGD and draw with green color.

The update mentioned above for a regular frame is also applied to other types of frames. For each frame, algorithm checks the distance ratio of the pixels of the frame to the minimum and to the maximum. According to ratio difference, the pixel is determined as MGD or NO-MGD.

The algorithm before the update was checking the distance. In fact, the algorithm was checking the ratio also but the ratio condition was one. New update added a new variable and assigned 0.5 as value. New algorithm after the update checks the distance ratio and determine the pixel as MGD if the distance ratio to local minimum over local maximum is less than one and more than WIDE VAR(0.5).

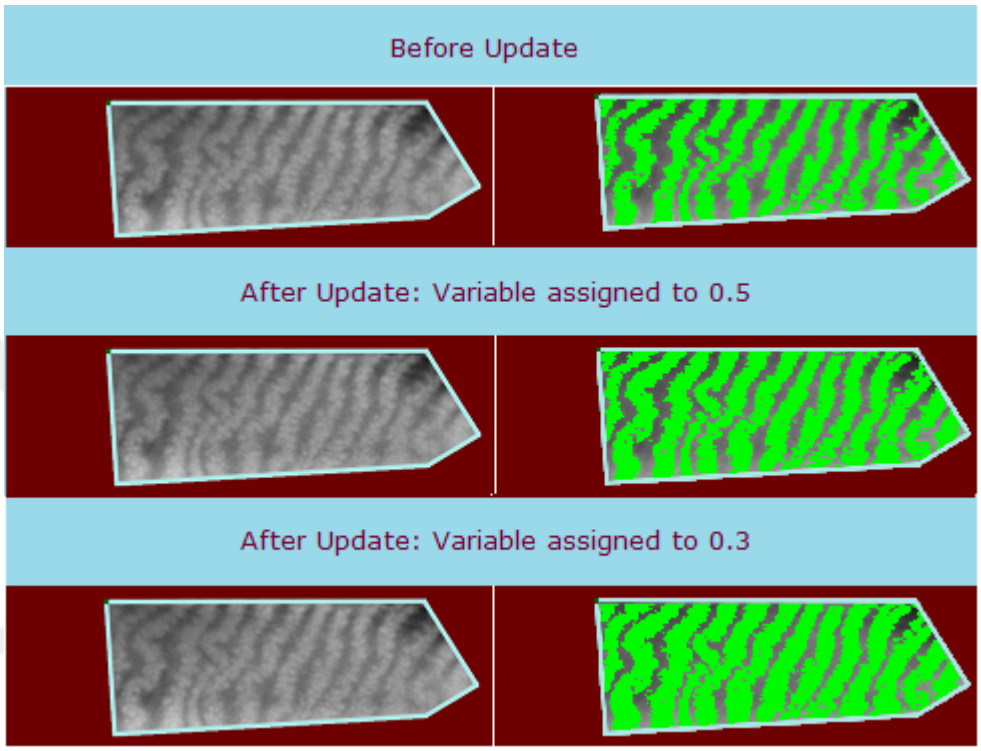

Figure 3.19 Update result for a wide MGD channel

Figure 3.19 represents the result of the update. There are three different output images. The first image is the output of the algorithm before the update. Another image is the output after the update with the variable value assigned to "0.5". The third result represents the result with the variable value assigned to "0.3".

Variable "WIDE VAR" makes the determined glands wide when the value is less than one. Variable will make the glands less wide if the value is more than one. The value of the variable can be assigned to one in order to return to the previous algorithm before the update. The variable can be used for setting the precision of the software. Changing the value will result in the output which has glands with more or less width. Optimum value for the variable seems to be between "0.3" and "1" in order to make the results better.

Another revision requires: "There should not be gap inside a gland."

MGD determination algorithm draws the glands with green color. Algorithm checks local maximum and local minimum points to form frames in order to determine MGD pixels. In some rows, there may be a MGD pixel set and then a NO-MGD pixel set and again a MGD pixel set. In this case, if the width of NO-MGD pixel set is large than that means there are two glands. If the width is not large, five or ten pixels, it can be said that NO-MGD pixels are the part of the glands. For this reason, there is one gland.

In this thesis, frames are processed independent from other frames. A NO-MGD pixel set between two MGD pixel sets can be interpreted in different ways. As mentioned above, according to the width of the NO-MGD pixel set, glands can be behaved as one gland or two glands. As the original attitude of the algorithm, a NO-MGD pixel set between two glands are behaved as NO-MGD. The glands are behaved as two different glands even if the width of NO-MGD pixel set is not large. However, when each row is combined, NO-MGD pixels are forming a gap inside a gland. Current revision requires that there should not be a gap inside a gland.

Update will draw the gap inside a gland with green color. NO-MGD pixels which are not large inside a gland will be behaved as MGD and drawn with green color. The update will be applied on the result image which is shown to the physician and will be written in Javascript because will be applied to the frontend of the software.

Method for this update will group NO-MGD pixels on the result image. For each group, number of pixels will be counted. If the number of pixles is less than a value, like three hundred, the method will determine the pixels in the group as MGD and draw with green color. If the number of pixels is more than the value, pixels will be behaved as NO-MGD and will not be drawn with green color. This way, the method will determine the pixels less than three hundred which make a gap inside a gland as MGD and draw with green color.

The method for this update that will draw gaps which are not large inside the interest area to green color can be written as following

```
GROUP_NOT_MGD_PIXELS();
FOREACH GROUP AS C:
   IF LENGTH(C) < 300:
      DRAW_WITH_GREEN_COLOR(C)
```

Method checks each NO-MGD pixel on the result image. According to enclosing pixels, a NO-MGD pixel is added to a group. After groups are formed, the number of pixels in each group is counted. If the number of pixels is less than three hundred, pixels in the group are determined as MGD and drawn with green color.

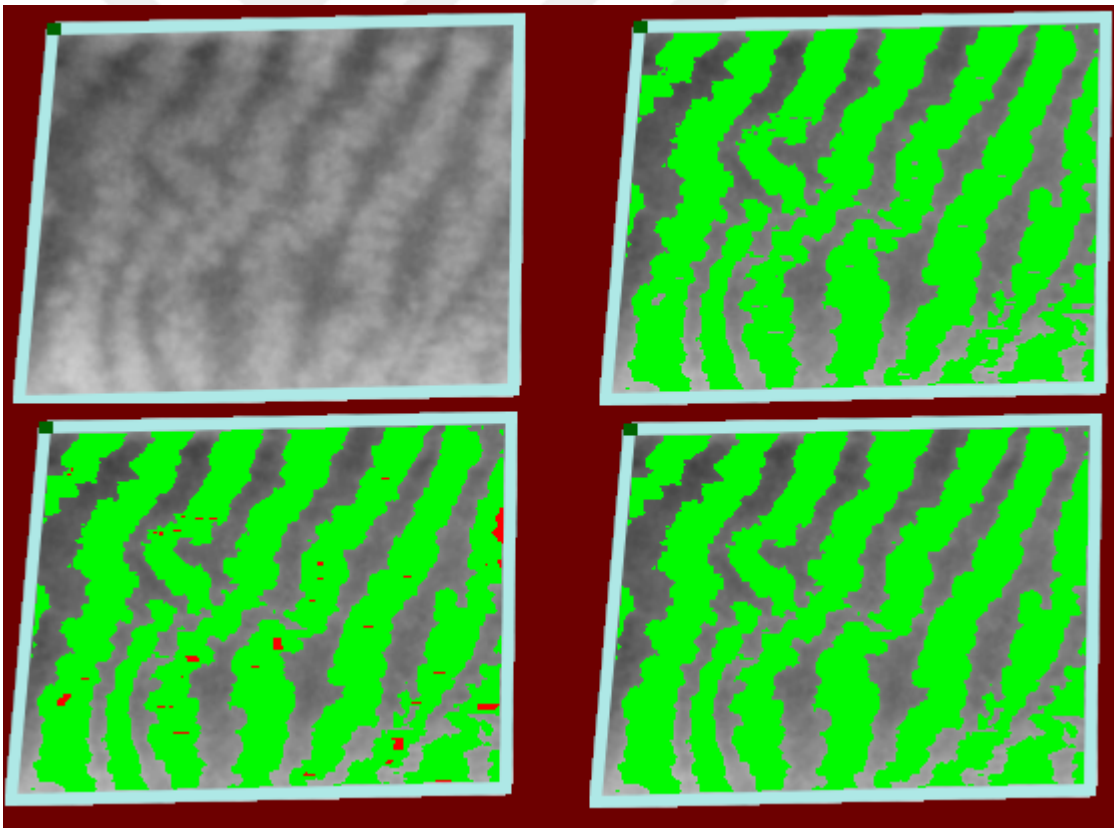

Figure 3.20 Update result for gap inside a gland

Gap which is not large inside a gland is determined as MGD as shown in the figure. The needed revision is completed with this update. Glands are determined better because there is no gap inside.

The third revision requires that:
"NO-MGD area should not be determined as MGD"

Current algorithm uses increasing & decreasing colors of a row to form frames. Algorithm checks the color values of pixels in a frame to decide if the pixel is a MGD pixel. If the color distance of a pixel to local minimum is more than the distance to local maximum, the algorithm determines the pixel as MGD. In addition, This algorithm is revised in Revision 1 to make MGD glands wide.

On a meibography image, there may be some pixels that the algorithm determined as MGD but in fact, the pixels are not MGD. This condition is because of the mechanism of the algorithm. According to current algorithm, brightness on the image means a potential MGD pixel set. Feature of drawing the interest area helps the user of the software focus on the area including real MGD pixels. In fact, the interest area which will be drawn by the physician will be the whole inner part of eyelid of the patient. There will be some pixels determined as MGD by the software but in fact, the pixels are not MGD pixels

Feature of editing result image provides a tool to the user in order to edit the result image. Using the editing tool, new MGD or NO-MGD pixels can be drawn on the image. The user can mark the area will be changed. Using the buttons, new MGD pixels can be drawn on the area or some MGD pixels can be removed. In addition to this feature, following code is added as the update to the algorithm.

```
IF LENGTH(MGD_PIXEL_SET) > 5 AND LENGTH(MGD_PIXEL_SET) < 100:
   DRAW_WITH_GREEN_COLOR({COLUMN},{ROW})
```

The code represents the codition that is added to the algorithm. New algorithm checks the length of the pixel set. A potential MGD pixel set is determined as MGD if the length of the pixels is more than five and less than one hundred. For this reason, length of a gland will be between five and one hundred.

Remaining revision is about considering brightness on a meibography image. There are some cases that IR photograph is containing some unexpected bright area. The example of the case is represented in Figure 3.21

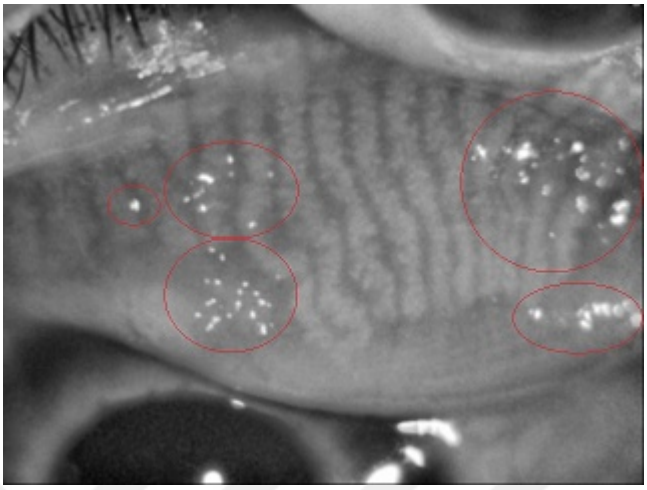

Figure 3.21 Brightness on a meibography image

Brightness on the image is marked with red circle. The reason of this condition is the device taking the IR photograph. Real condition of the pixels in terms of MGD or NO-MGD can be understanded by looking at the image. Algorithms can also determine the brightness on the image using the color values. Bright area pixels will have the higher color values compared to other pixels on the image. When all pixels of the image is gathered, outlier pixels can be determined using the formula:

```
IF (COLOR_VALUE_of_PIXEL < MEAN - (2*STANDARD_DEVIATION)  OR
    COLOR_VALUE_of_PIXEL > MEAN + (2*STANDARD_DEVIATION)):
        BRIGHTNESS()
```

However, the algorithm can not determine the real condition of bright pixels in terms of MGD or NO-MGD. Because according to current algorithm, frames are behaved independent from other frames. Following versions of the software can have "Gland Following" algorithm to follow glands and decide if a bright pixel is representing MGD or NO-MGD. Fortunately, editing feature of the software provides a useful tool in order to edit the result image. The user can change bright pixels on the image to MGD or NO-MGD in seconds using the editing tool.

4. SOFTWARE OF THIS THESIS

Software is designed as a web application and requires a localhost configuration or connection to The Internet. The software provides a drawing feature to the physician for drawing the interest area of a meibography image. After drawing, software is giving the result in about five seconds. After the result is shown, the result image can be edited. User can add or remove some MGD pixels on the result image. This thesis focuses on giving a result which requires minimal editing.

For the development of the software, PHP and Javascript programming languages are preferred. In the backend, a PHP server is processing the image and resulting to the frontend. In the frontend, a set of interactive tools is helping the physician do some browser related work and edit the result.

In the following list, functions and variables of the canvas element are listed. Canvas is the HTML element which allows users to draw on.

- getImageData: Color value at specified index
- fillStyle: Drawing configuration
- fillRect: A rectangle according to fillStyle
- lineTo: A line between two points
- lineWidth: Length of the line will be drawn
- canvas.width: Width of the canvas
- canvas.height: Height of the canvas

Frontend part of the software is using the functions and variables above. Frontend of the software includes features that help the user upload a meibography image, draw the interest area and edit the result.

Flow charts are useful for learning the general mechanism of a system. Flow chart is a diagram that represents how a system is operating. Following diagram is the flow chart representation of the software of this thesis

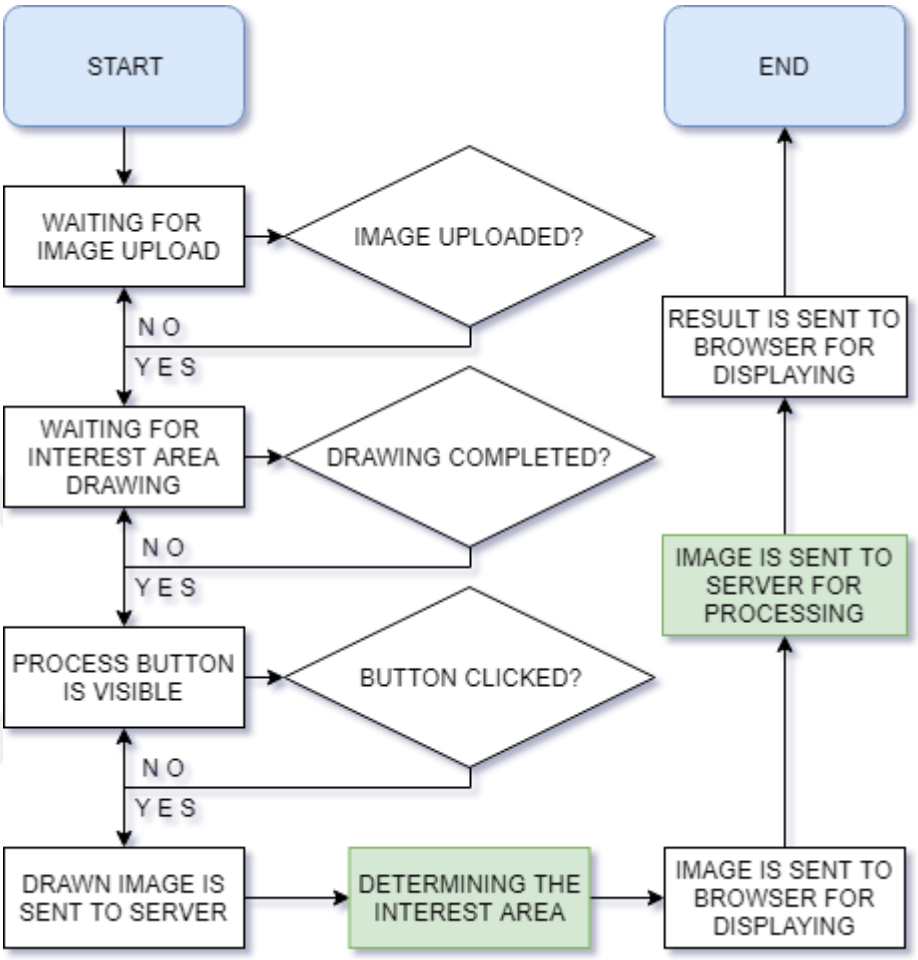

Figure 4.1 Flow Chart of the Application

Rectangles with green background represents the work done in server while the white rectangles are representation of the work done in the frontend. The image is sent to server for processing and sent back to frontend for displaying. Data is transmitted through AJAX calls from Javascript to PHP and PHP to Javascript.

When the software is visited using a web browser, a meibography image is waited to be uploaded. User can use the "Choose File" button for uploading the image. Although the software is waiting for a meibography image, some other files are also allowed. File type checking algorithm is not applied to the software.

Determination of the interest area is important for the correctness of the result. Processing the whole image reuqires multiple algorithms to determine the interest area and give the correct result. Software of this thesis provides a tool for drawing the interest area. A rectangular can be drawn on the image in about ten seconds.

The user clicks on the image and starts drawing, following click draws a line between the points. Each click adds a line between the previous click. This way, a rectangular can be drawn with five click.

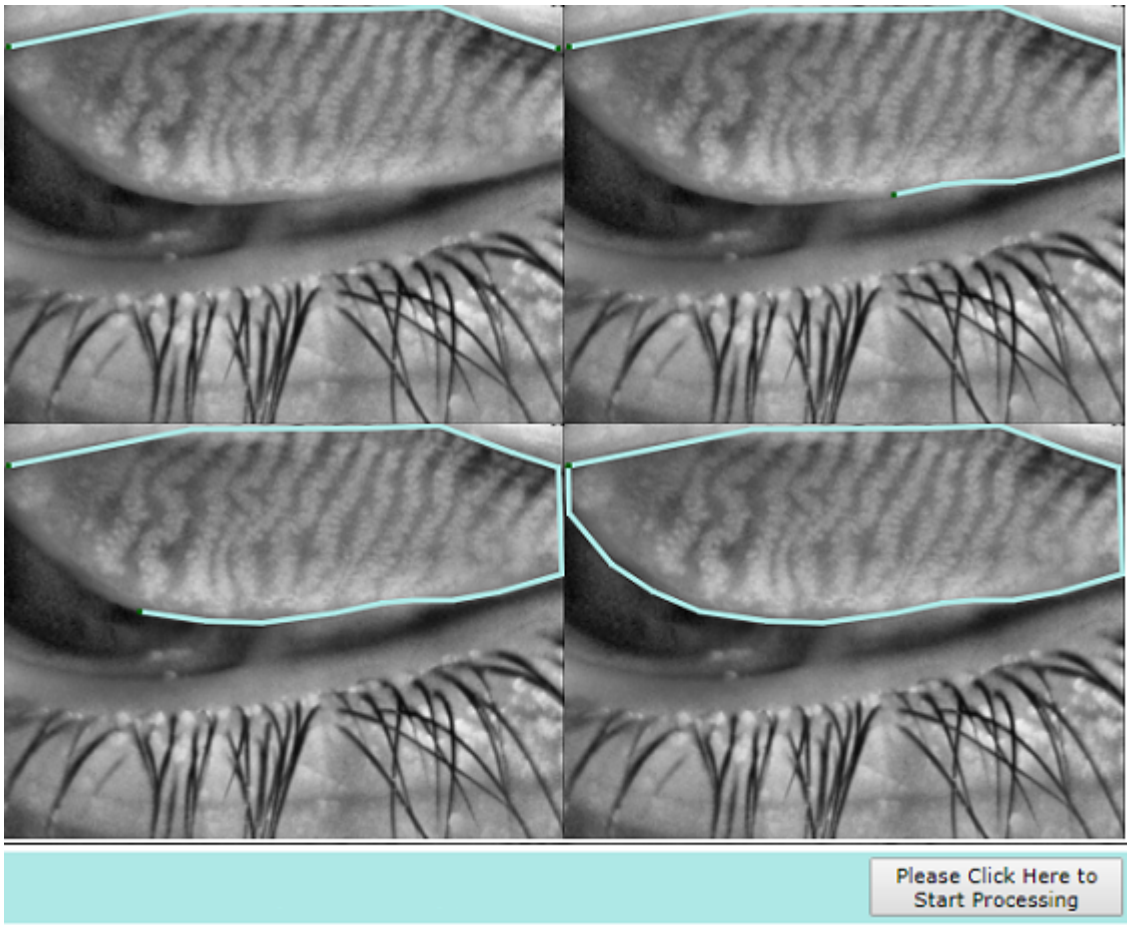

Figure 4.2 Drawing the Interest Area

Figure 4.2 represents the example of interest area drawing feature. When the starting and the ending point is connected, "Process" button becomes visible to the user for processing the image. User clicks the "Process" button and the image is sent to server for processing. After the result is shown, user can return back to interest area drawing feature and can send the image to server to process again.

One of the examples of output of the software is presented in Figure 4.3. Number of MGD and NO-MGD pixels along with the ratio is returned from the software.

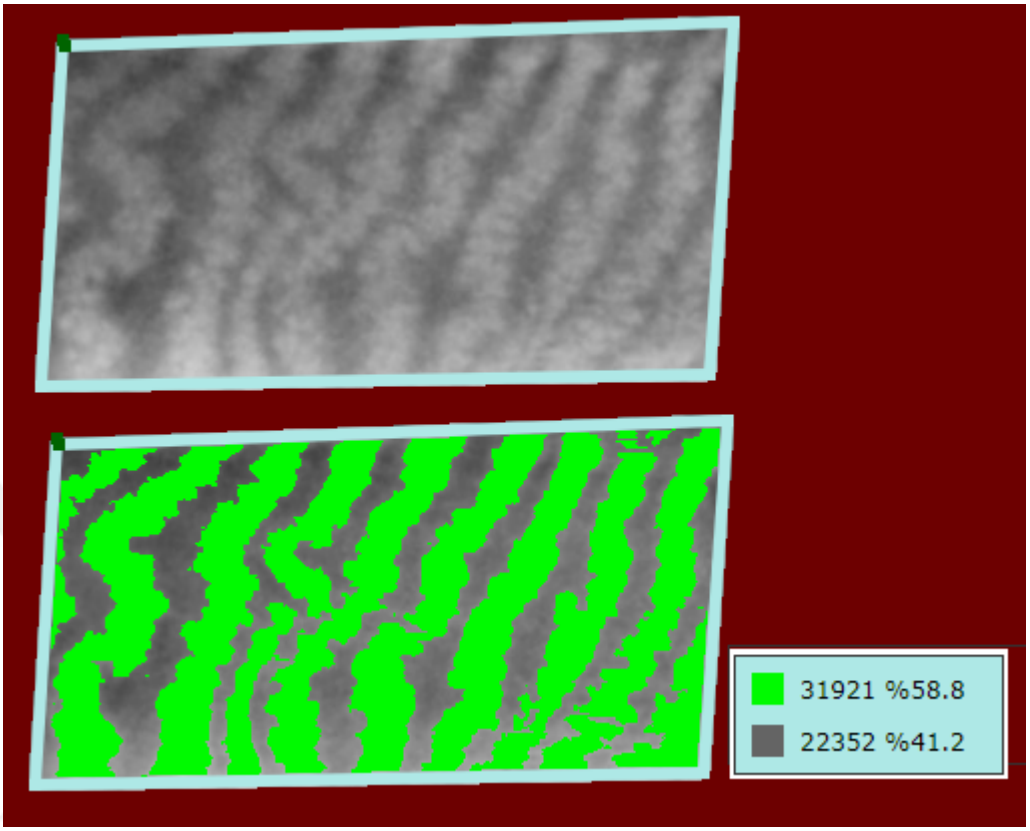

Figure 4.3 Output of the Software

As presented in the following chapter (Chapter 5 Testing and Validation), number of pixels that are MGD or NO-MGD along with the ratio is calculated correct. In addition to this, the physician may need to edit the result like adding or removing MGD pixels to correct some area on the image. There is a tool in the software helping the user edit the result. Physician can mark the area which editing will be applied. Doctor can draw the marked area to MGD or NO-MGD.

Information returned from the software represents the count and the percentage of pixels in the interest area. According to the result,
Number of MGD pixels is: 31.921.
Number of NO-MGD pixels is: 22.352.
Percentage of MGD pixels (58.8%)
Area of loss (41.2%).

After the result is returned from the software, the result image can be edited to correct misdetermined areas. For this purpose, editing feature of the software is used. User checks the correction of the result. Using the editing tool, misdetermined areas can be marked and changed in seconds

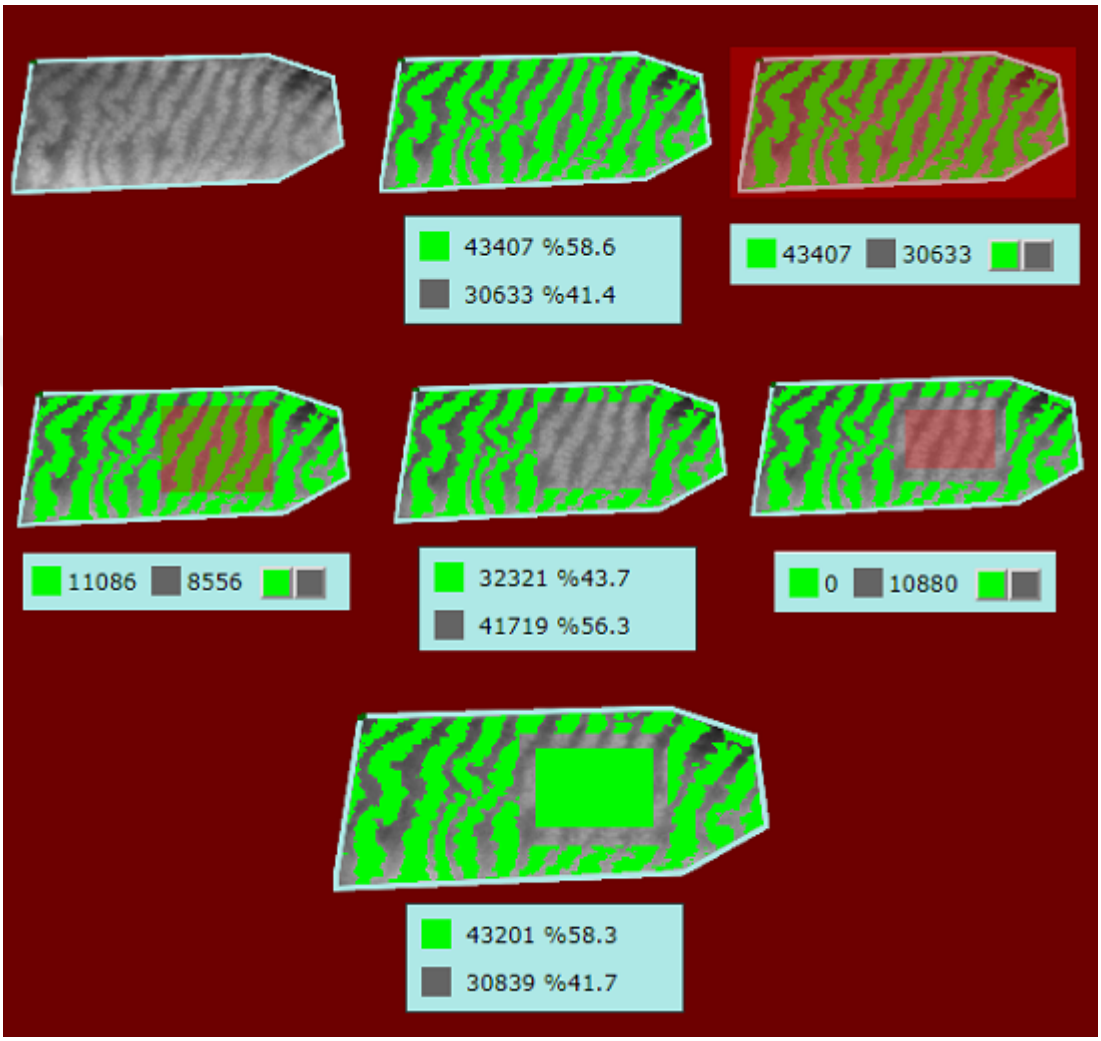

Figure 4.4 Example of editing result

Figure 4.4 is the example of editing the result image. In each editing, pixel count and percentage returned from the software seems to be correct according to results

To edit the result image, user marks the area that editing will be applied. User sees the count of pixels inside the marked area. With the count, button for changing the pixels becomes visible. User can draw the pixels inside the marked area with MGD or NO-MGD color.

5. TESTING AND VALIDATION

For validation, knowing the correctness of the numbers and the percentages returned from the software is important. As the final product, a reliable software is expected. For this reason, numbers and percentages must be checked using a logical validation method. Following figure represents results from different methods in the backend and the frontend of the software. Result on the left is from server side while on the right is from frontend. Both method are giving the same result.

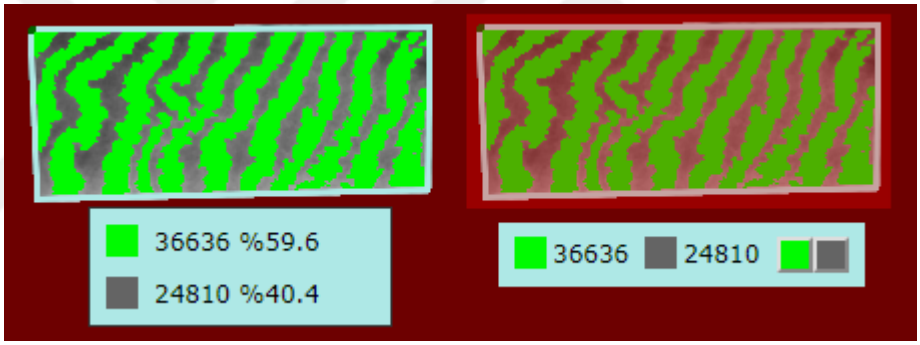

Figure 5.1 Validation of Number and Ratio

Calculation of percentages for the method in frontend is as following. According to percentages, software is giving a consistent output.

```
Percentage of Green = (36636 / (36636 + 24810)) * 100 = 59.6%
Percentage of Area of Loss = (24810 / (36636 + 24810)) * 100 = 40.4%
```

In addition to testing and validation, a formal method is needed for comparing the results of manual determination with the result of automated detection. As the method, correlation plot and Bland&Altman method are used. Using the Area of loss, needed variables are calculated and plot is drawn.

Evaluating the agreement between quantitative results is common. Correct approach to evaluate this degree of agreement is not obvious. Correlation and regression are proposed often. Correlation checks the relationship between one variable and another, not the difference. [9] In 1983, Bland&Altman developed the alternative method based on the quantification of the agreement between two quantitative result by checking the mean difference and constructing the limit of agreement. [9]

Bland&Altman method is a simple way to evaluate a bias between mean differences, and to estimate the confidence interval. Data can be studied both as unit differences plot and as percentage differences plot. The method does not tell whether the limit of interval is acceptable or not. Acceptable limit must be decided based on clinical need and biological idea [9].

In this thesis, Bland&Altman method is applied to different tests represented with the following list. There are four tests in total. Each test has twenty results. Results of tests can be represented as following.

- Result from manual determination
- Result from automated detection on day one
- Result from automated detection on day two
- Result from automated detection by other user

There are twenty results from manual determination. A person will get twenty results from the software on day one. Same person will get another twenty results on day two. In addition, another person will get twenty results from the software. Results will be used in Bland&Altman method for calculating the variables and drawing the plot.

Each test containing twenty results will be compared to each other. There will be six different correlation and Bland&Alman plot in total. Combination of four tests can be represented as ((1,2),(1,3),(1,4),(2,3),(2,4),(3,4)). For each match, there are three example images presented as following.

Figure 5.2, Figure 5.3 and Figure 5.4 are showing the difference between results of manual determination and automated detection on day one.

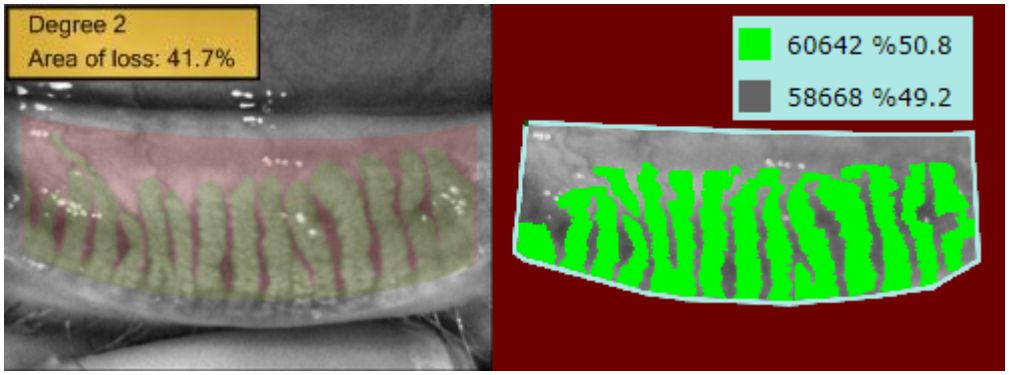

Figure 5.2 Manual determination & Automated detection on day one

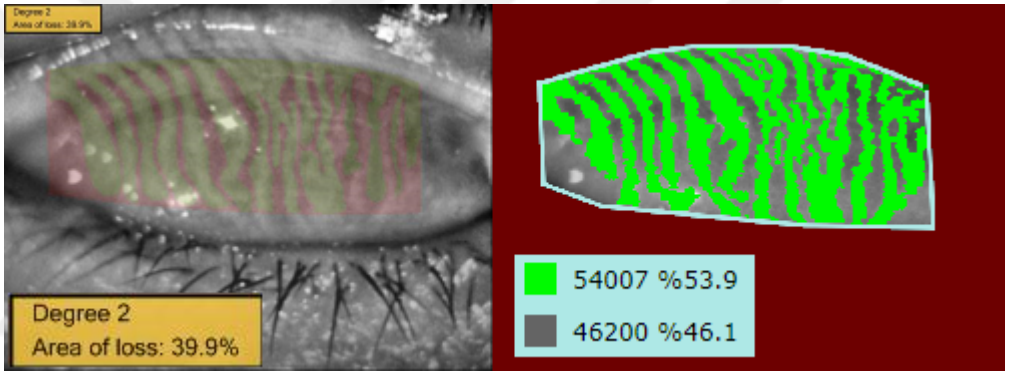

Figure 5.3 Manual determination & Automated detection on day one

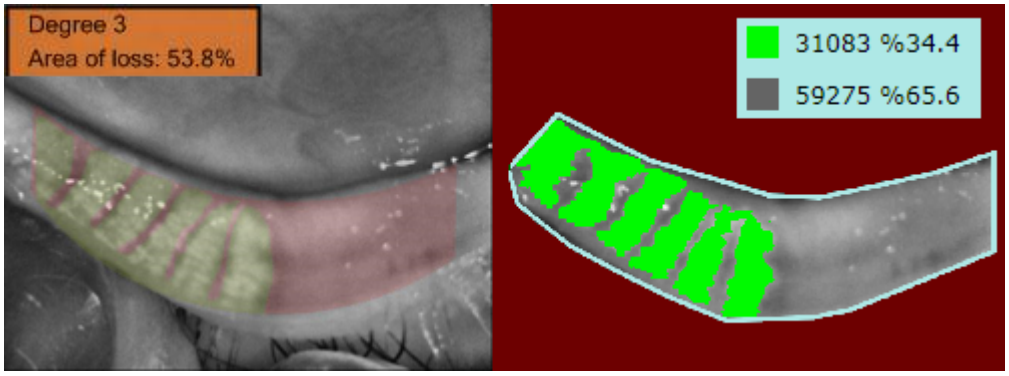

Figure 5.4 Manual determination & Automated detection on day one

Manual determination Area of Loss: 41.7%, 39.9%, 53.8%

Manual determination MGD pixels: 58.3%, 60.1%, 46.2%

Automated detection Area of Loss: 49.2%, 46.1%, 65.6%

Automated detection MGD pixels: 50.8%, 53.9%, 34.4%

Document is calculating variables and drawing correlation and Bland&Altman plot using the results of manual determination and automated detection on day one.

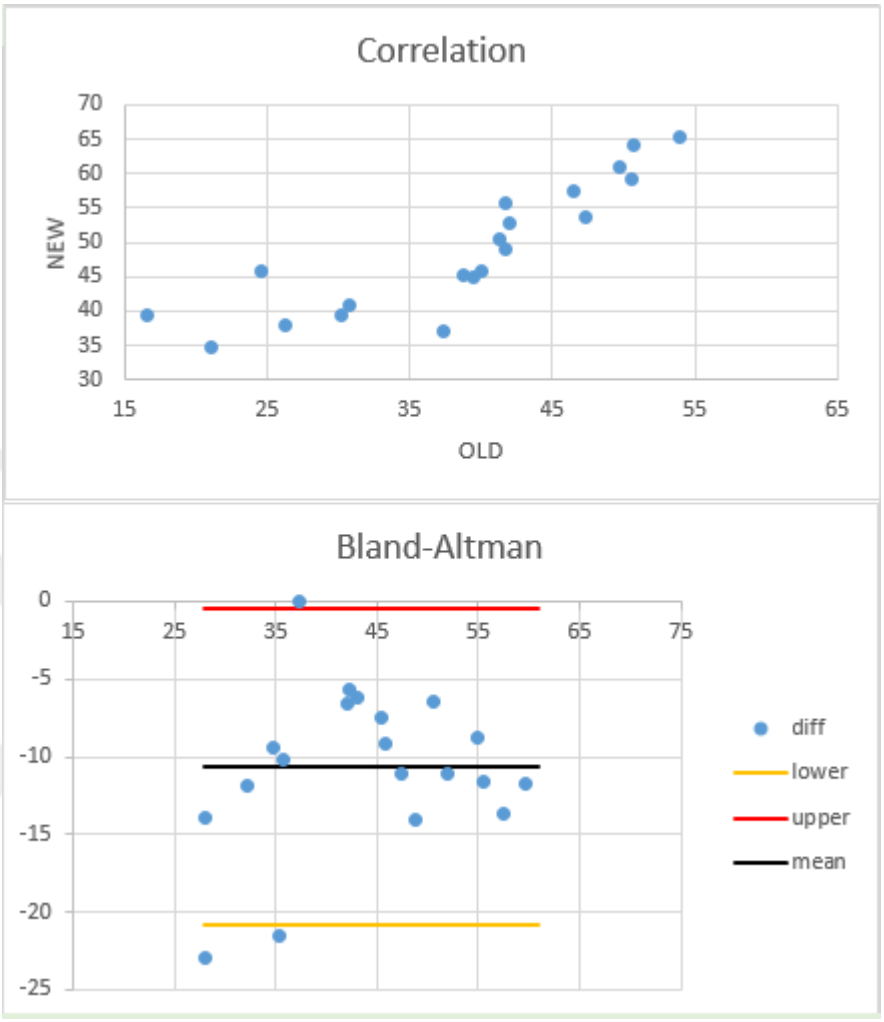

Figure 5.5 Manual determination and Automated detection on day one

Up line: -0.475096483

Low line: -20.89490352

Mean line: -10.685

Maximum difference: 23

Minimum difference: 0

Confidence interval is the region between the lower and the upper line. According to the Bland&Altman plot, seventeen results are in the confidence interval. Maximum difference is "23". Minimum difference is "0". Zero difference means that one of the result of manual determination and automated detection on day one is same.

Bland&Altman test is also applied to result of manual determination and automated detection on day two. Following images represent three of twenty results.

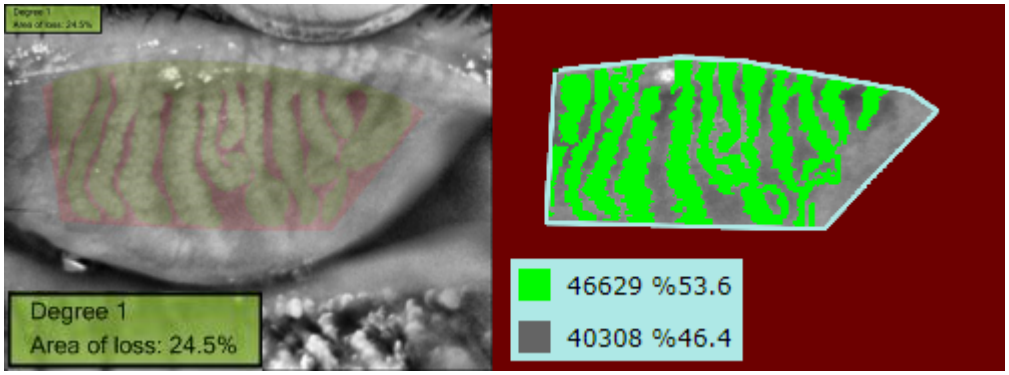

Figure 5.6 Manual determination & Automated detection on day two

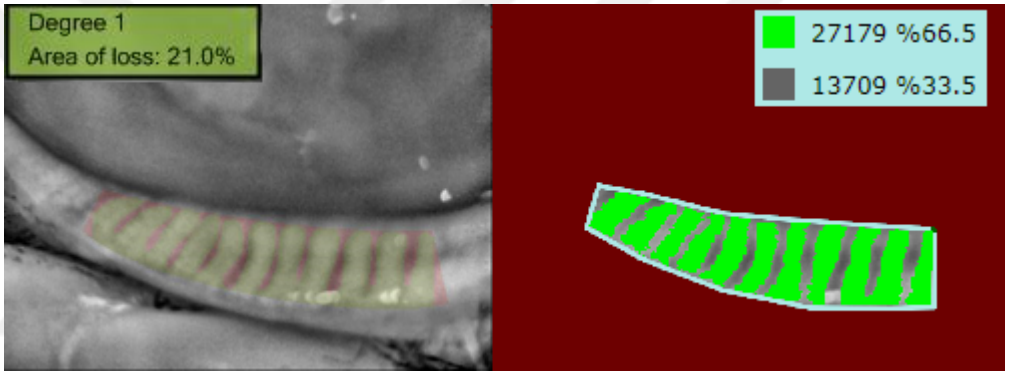

Figure 5.7 Manual determination & Automated detection on day two

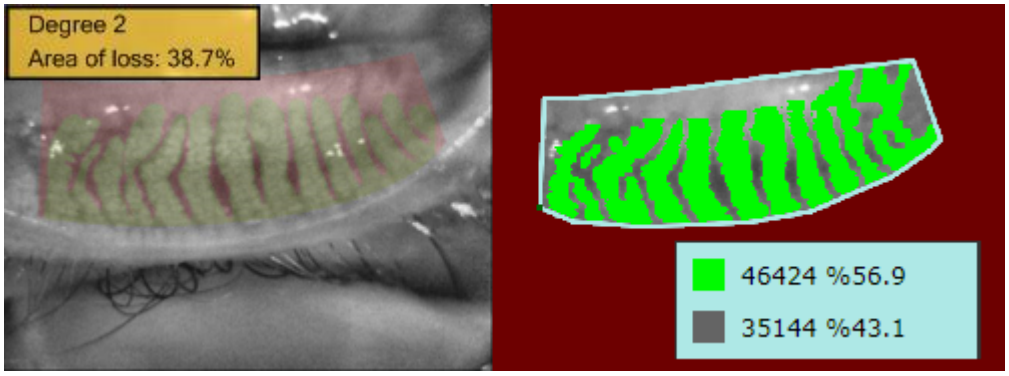

Figure 5.8 Manual determination & Automated detection on day two

Manual determination Area of Loss: 24.5%, 21%, 38.7%

Manual determination MGD pixels: 75.5%, 79%, 61.3%

Automated detection Area of Loss: 46.4%, 33.5%, 43.1%

Automated detection MGD pixels: 53.6%, 66.5%, 56.9%

Twenty results from manual determination, twenty results from automated detection on day two are used by the document for drawing the following plot.

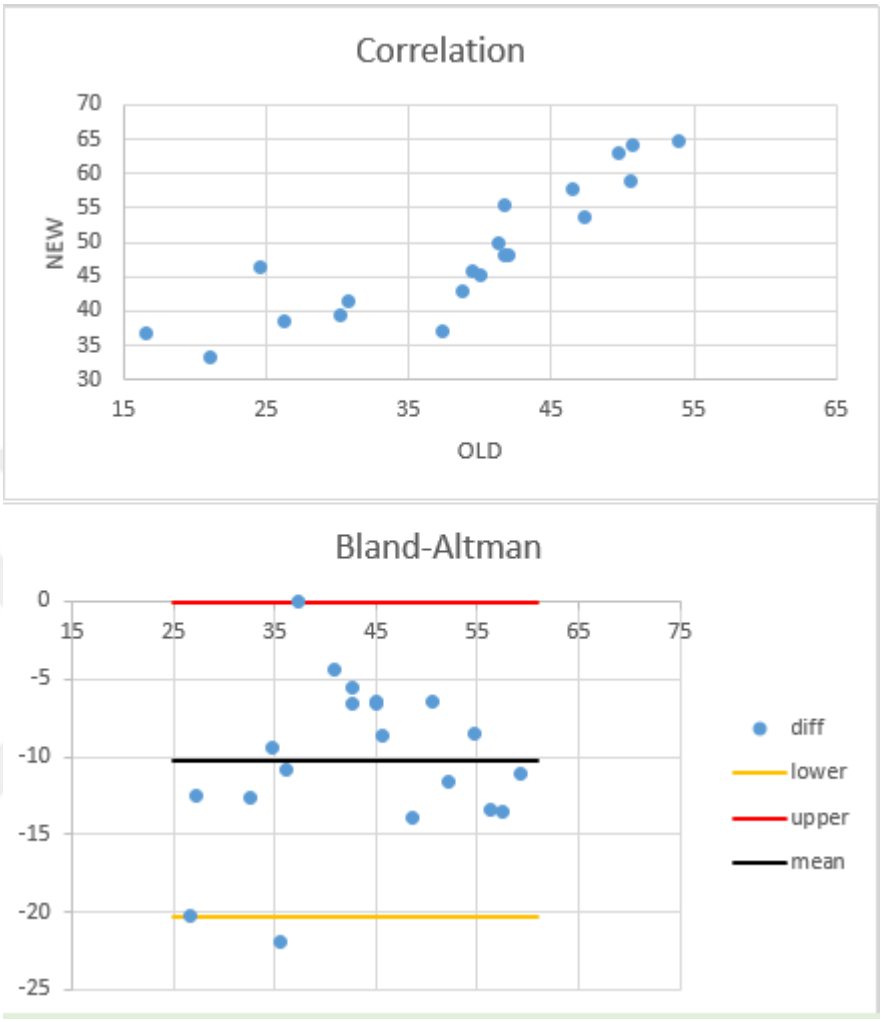

Figure 5.9 Manual determination and Automated detection on day two

Up line: -0.08062551

Low line: -20.34937449

Mean line: -10.215

Maximum difference: 21.9

Minimum difference: 0

According to the plot, confidence interval is including seventeen results. Maximum difference is "21.9". As similar to the previous test, the minimum difference is "0". One of the results is same.

Third Bland&Altman test is applied to the result of manual determination and automated detection by the other user.

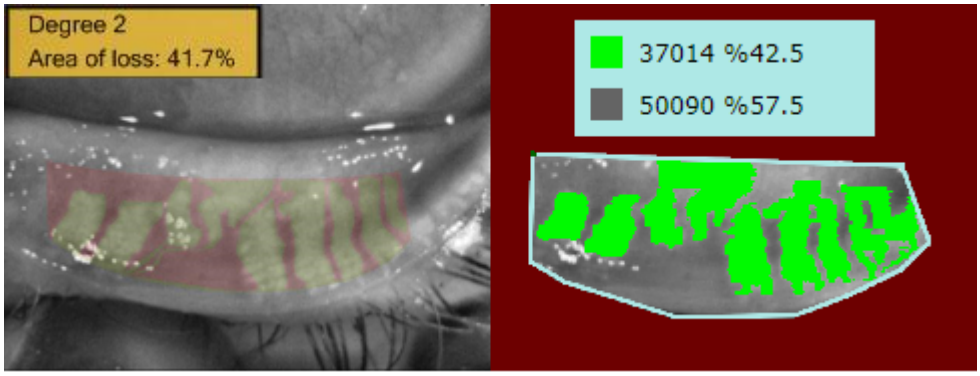

Figure 5.10 Manual determination & Automated detection by other user

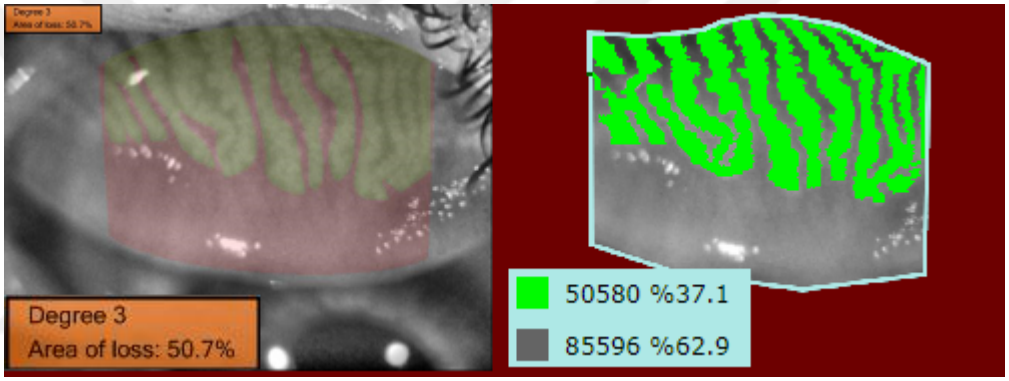

Figure 5.11 Manual determination & Automated detection by other user

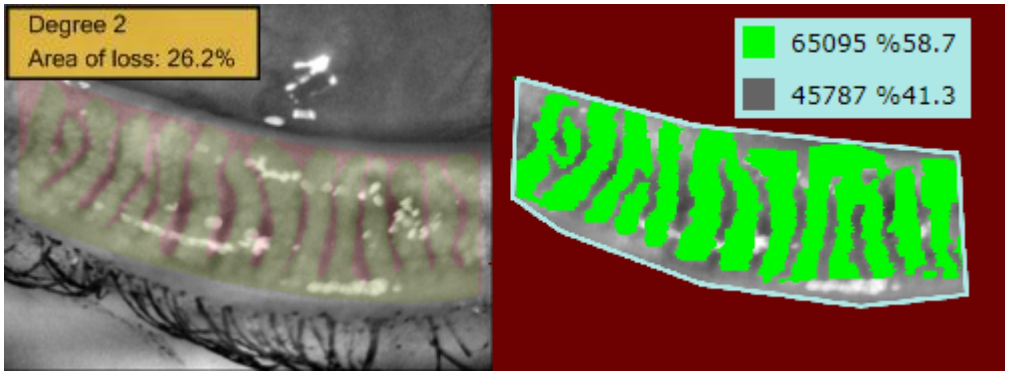

Figure 5.12 Manual determination & Automated detection by other user

Manual determination Area of Loss: 41.7%, 50.7%, 26.2%

Manual determination MGD pixels: 58.3%, 49.3%, 73.8%

Automated detection Area of Loss: 57.5%, 62.9%, 41.3%

Automated detection MGD pixels: 42.5%, 37.1%, 58.7%

Correlation and Bland&Altman plot for the test is as following. Correlation plot is similar to the plot of the previous test.

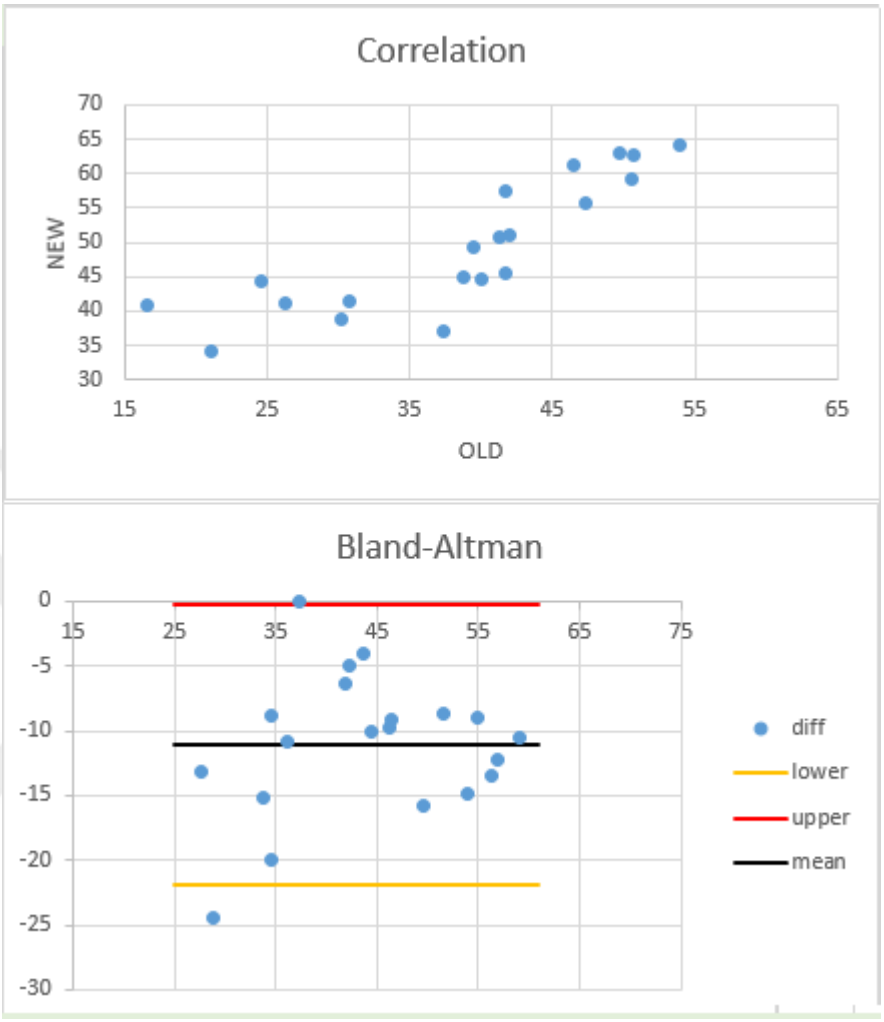

Figure 5.13 Manual determination and Automated detection by other user

Up line: -0.233989458

Low line: -21.88601054

Mean line: -11.06

Maximum difference: 24.5

Minimum difference: 0

According to the Bland&Altman plot, eighteen of twenty results are in confidence interval. Maximum difference is "24.5". Minimum difference is "0". One of the results is the same. Correlation and Bland&Altman plots for manual determination and automated detection look like the previous.

In addition to results of manual determination, results of automated detection on day one and on day two will be evaluated using Bland&Altman method.

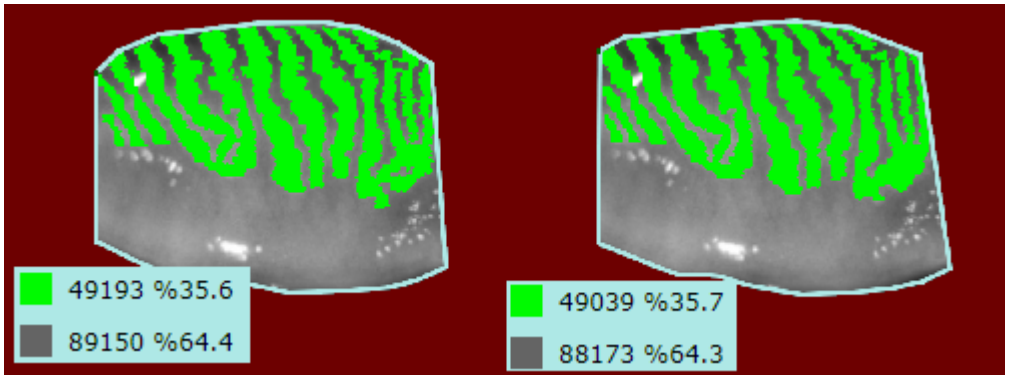

Figure 5.14 Automated detection on day one & on day two

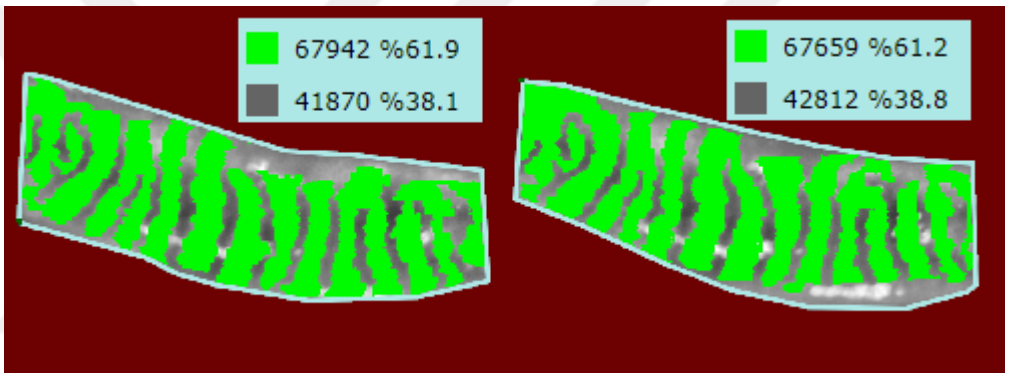

Figure 5.15 Automated detection on day one & on day two

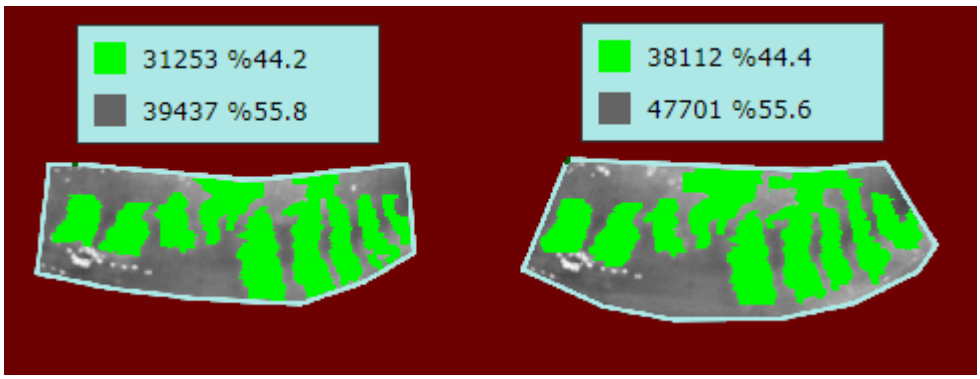

Figure 5.16 Automated detection on day one & on day two

Automated detection on day one Area of Loss: 64.4%, 38.1%, 55.8%

Automated detection on day one MGD pixels: 35.6%, 61.9%, 44.2%

Automated detection on day two Area of Loss: 64.3%, 38.8%, 55.6%

Automated detection on day two MGD pixels: 35.7%, 61.2%, 44.4%

Correlation plot for this test looks like a 45° line. This condition represents that results on day one and day two are almost same.

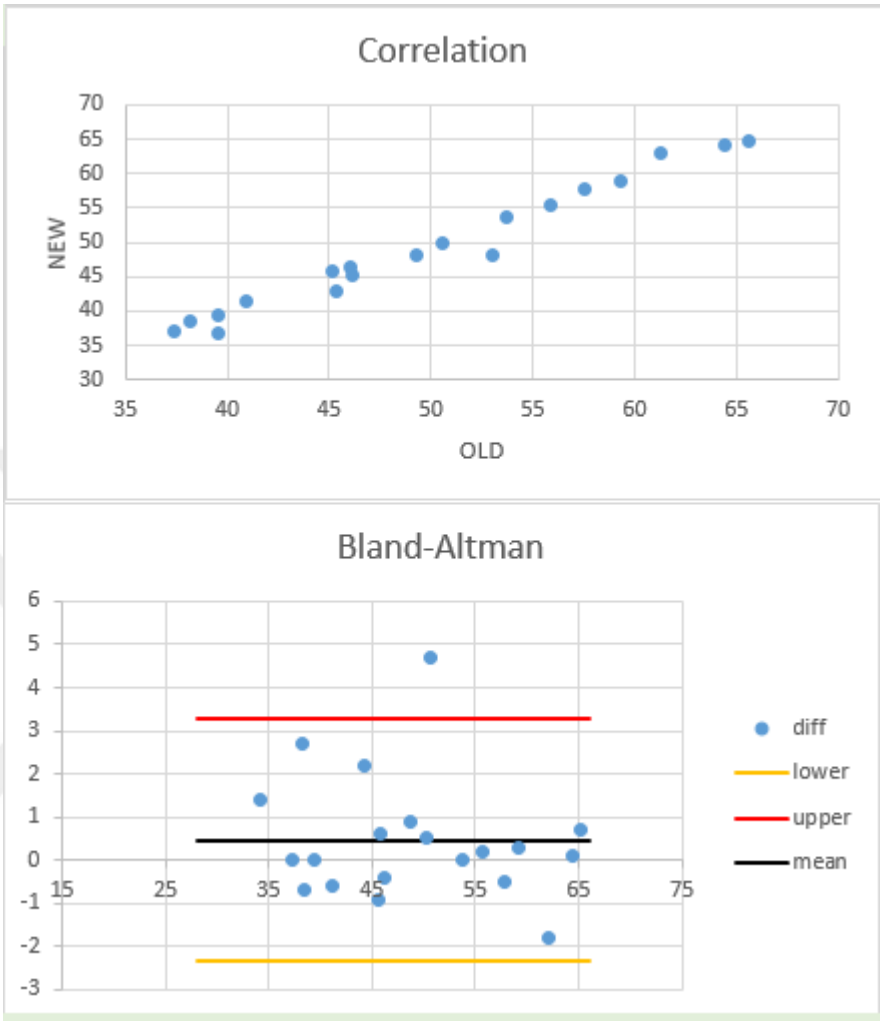

Figure 5.17 Automated detection on day one and on day two

Up line: 3.282430602

Low line: -2.342430602

Mean line: 0.47

Maximum difference: 4.7

Minimum difference: 0

Purpose of this test is to check if the software is reliable. According to plot, nineteen results are in the confidence interval. Maximum difference is "4.7". Minimum difference is "0". All results are the same or similar.

The results of automated detection on day one and by other user will be evaluated. Software must give the same output to the different user for the same input.

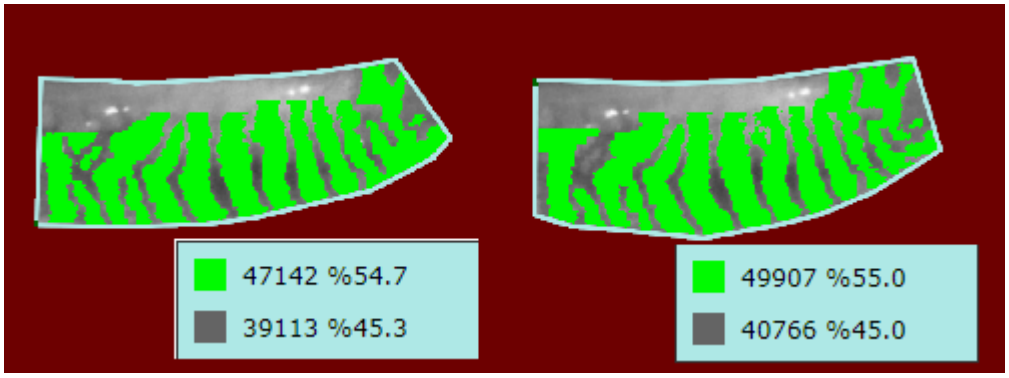

Figure 5.18 Automated detection on day one & by other user

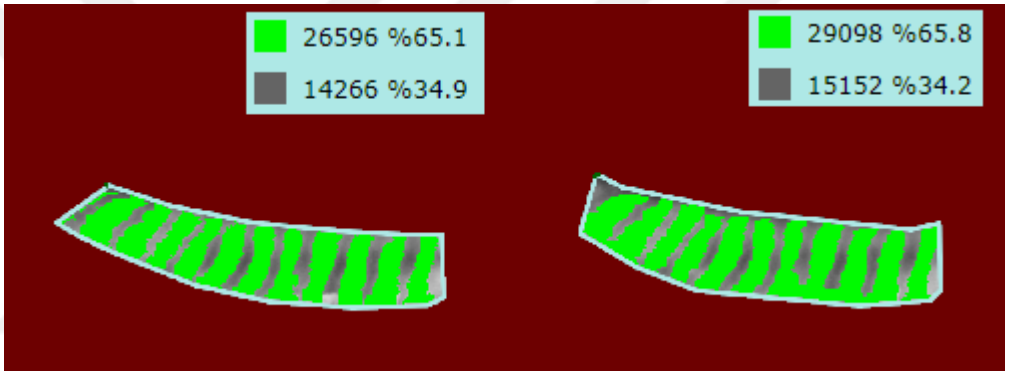

Figure 5.19 Automated detection on day one & by other user

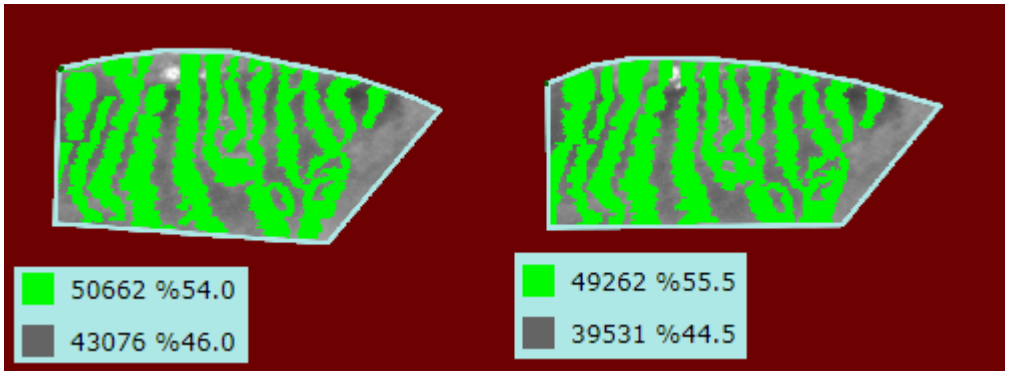

Figure 5.20 Automated detection on day one & by other user

Automated detection on day one Area of Loss: 45.3%, 34.9%, 46%

Automated detection on day one MGD pixels: 54.7%, 65.1%, 54%

Automated detection by other user Area of Loss: 45%, 34.2%, 44.5%

Automated detection by other user MGD pixels: 55%, 65.8%, 55.5%

Correlation plot for the results of automated detection on day one and by other user represents that results of both user tests are almost the same.

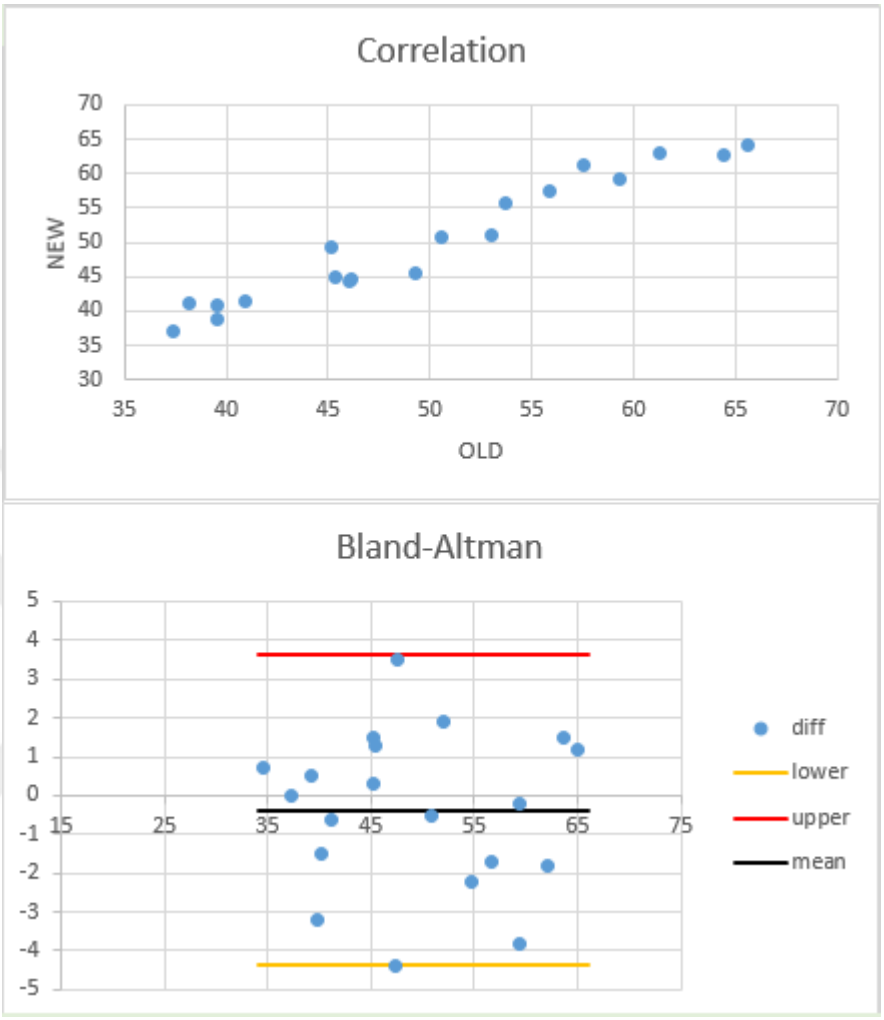

Figure 5.21 Automated detection on day one and by other user

Up line: 3.632092764

Low line: -4.382092764

Mean line: -0.375

Maximum difference: 4.4

Minimum difference: 0

There is no result that out of the Bland&Altman confidence interval. Maximum difference is "4.4". Minimum difference is "0". Results of two differents tests from different users are almost same.

After tests above are completed, Bland&Altman test will be applied to the results of automated detection on day two and by other user.

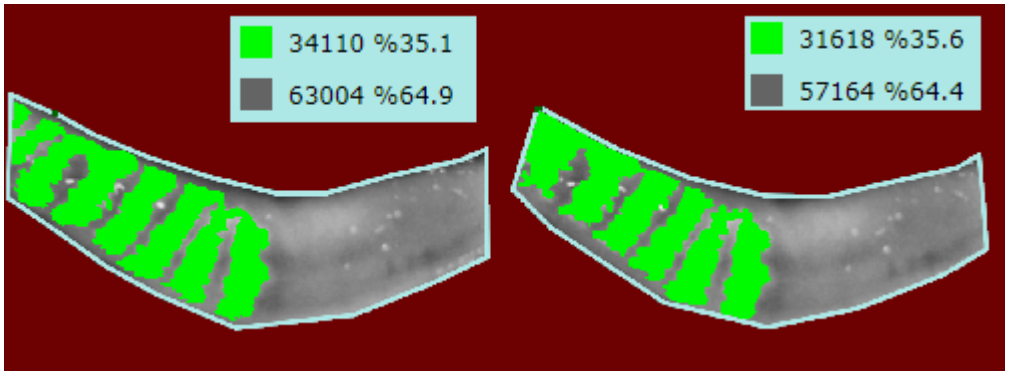

Figure 5.22 Automated detection on day two & by other user

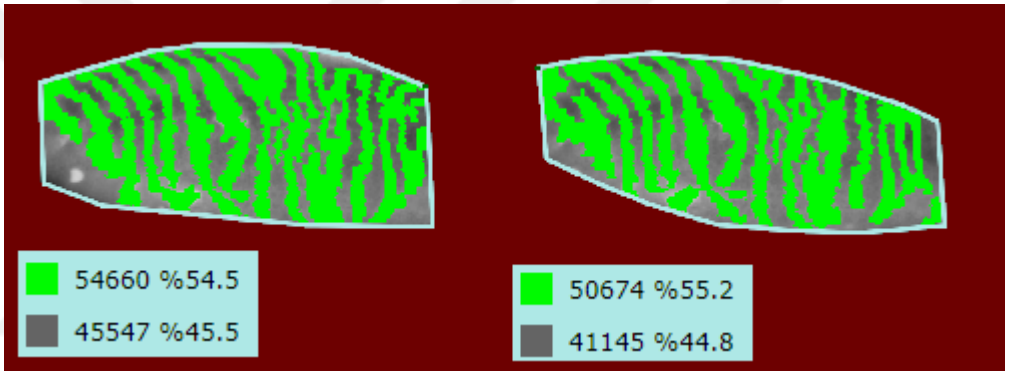

Figure 5.23 Automated detection on day two & by other user

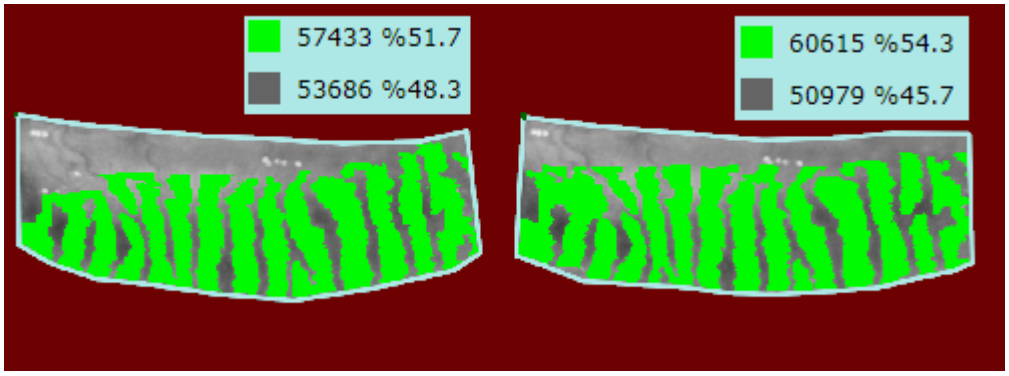

Figure 5.24 Automated detection on day two & by other user

Automated detection on day two Area of Loss: 64.9%, 45.5%, 48.3%

Automated detection on day two MGD pixels: 35.1%, 54.5%, 51.7%

Automated detection by other user Area of Loss: 64.4%, 44.8%, 45.7%

Automated detection by other user MGD pixels: 35.6%, 55.2%, 54.3%

The sixth Correlation and Bland&Altman test is applied on the results of automated detection on day two and by other user

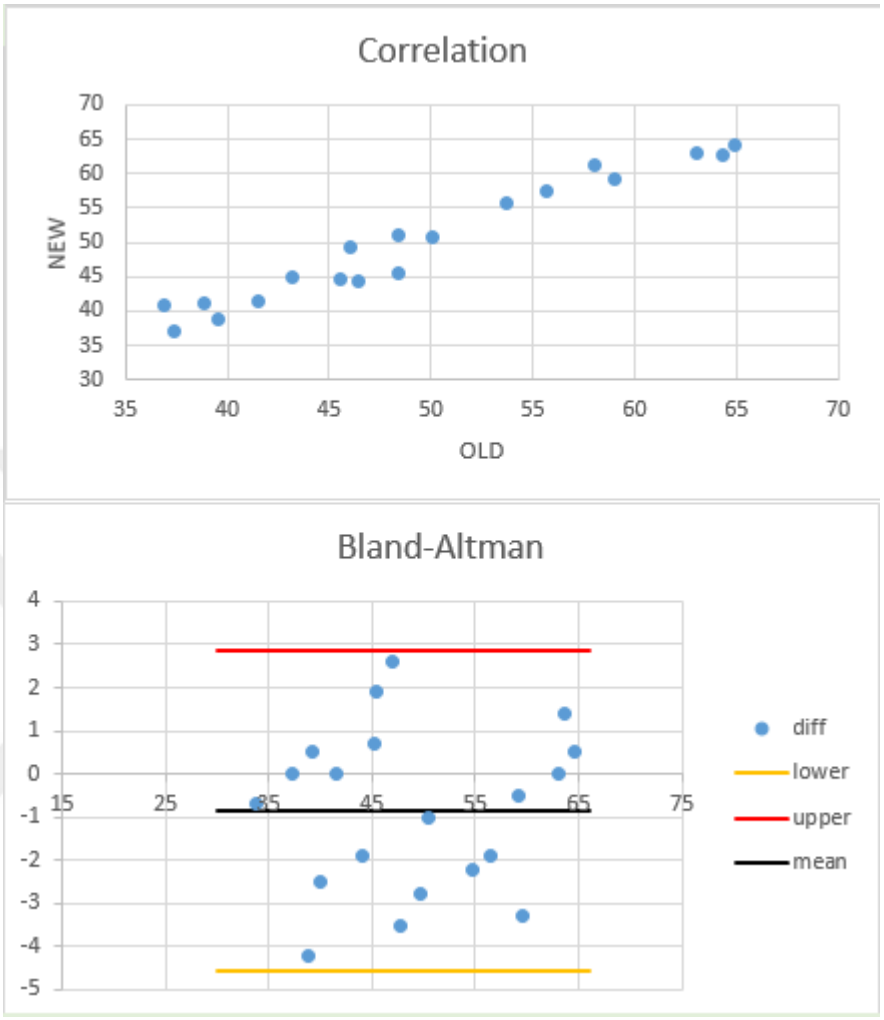

Figure 5.25 Automated detection on day two and by other user

Upper line: 2.854952034

Low line: -4.544952034

Mean line: -0.845

Maximum difference: 4.2

Minimum difference: 0

Like the previous plot, there is no result that out of the confidence interval. Maximum difference is "4.2". The minimum difference is "0". The results of two different tests from different users are almost same.

Bland&Altman method is applied to each combination of four different tests. Tests are including twenty results. One test is compared to each other. For this reason, there are six correlation and Bland&Altman plots.

Variables and plots are presented using the Excel document which is programmed to calculate variables for the method and draw correlation and Bland&Altman plot. In this thesis, six Bland&Altman document is using the following kind of tests.

- Manual Determination & Test done by user on day one
- Manual Determination & Test done by user on day two
- Manual Determination & Test done by other user
- Automated detection on day one & Automated detection on day two
- Automated detection on day one & Automated detection by other user
- Automated detection on day two & Automated detection by other user

The plots represent that the difference between results of manual determination and automated detection is about 10%. The difference between the results of automated detection on day one, day two and by other user is not high, about %1. This condition is telling that the software of this thesis is detecting more "Area of Loss" compared to the manual determination. The software is giving almost the same result to the user on different day and to the other user.

Bland&Altman plots made from the results of user tests are telling that the software is giving consistent output. User can get the same result from the software on different day. In addition, different users are getting similar or the same result.

Software of this thesis is providing editing tool to change the result. User can correct areas that misdetermined using the tool. Testing represent that some results of manual determination is the same with automated detection. For this reason, it can be said that user of the software can get the same result with manual determination using the editing tool. Software is appropriate for maintenance to make the result

6. CONCLUSIONS

In conclusion, the software using the algorithm mentioned in this thesis is developed. Purpose of the software is taking a meibography image as input and giving the MGD level as output. In the development of the software, PHP and Javascript programming languages are preferred.

Image processing methods are used for processing infrared meibography image. When deciding the MGD pixels, line chart representations of rows are used. For each chart, algorithm is considering pixels between two consecutive local minimum point as a potential MGD pixel set.

Meibography images have the characteristic of being greyscale. Since Red, Blue and Green value of each pixel on the image is equal, there are 255 colors in total on a infrared meibogprahy image. Algorithm is using the difference between color value of the columns to understand brightness. According to this thesis, brightness inside the interest area of a meibography image represent a potential MGD pixel set.

Software is requiring localhost configuration or connection to The Internet.
There are visible and server sides of the software.

In the visible side, also known as frontend, the physician is uploading a meibography image. The frontend provides a tool for drawing the interest area. Using the tool, the physician is drawing the area which will be processed by the software for MGD pixel determination. After drawing completed, a button which is for starting to process the image in server is visible. When the button is clicked, the image is sent to server for processing.

Purpose of the frontend is to provide tools to the physician in order to draw and edit a meibography image. Drawing tool is able to draw a rectangular with five click and also able to undo the drawing. Tool is making the software easier to use than the current method. Frontend of the software is developed with Javascript.

In the server side, image with drawn interest area is accepted and processed. After processing, the result is sent to the frontend. The PHP programming language is preferred to develop the server side of the application. Pseudocodes mentioned in this thesis are representation of PHP code running in the server.

The physician can edit the result image returned from the software. Tool of editing result image is another which is making the software easier to use than the current method.

Instead of processing the image in frontend using Javascript, the images are prepared in frontend and sent to server for processing. This is because of the performance difference of the languages on image processing. PHP code is running in the server while Javascript code is running on the browser. PHP is using the power of server when processing the image while Javascript is using the canvas element of HTML.

Although some performance refinements are applied to Javascript code, PHP is ten times faster than Javascript on image processing. For this reason, the images are sent to the server when possible. In addition, image processing work which does not require fast processing like drawing the interest area and editing the result image are done in frontend of the software using Javascript.

A formal validation method (Bland&Altman test) is applied to the result of the automated detection by the software and manual determination. The results of the method are presented. The software is giving a reliable output and working with a good performance and is able to determine the MGD level from the infrared meibography image. The software is appropriate for maintenance.

REFERENCES

[1] Arita R., Suehiro J., Haraguchi T., Shirakawa R., Tokoro H., Amano S. (2013). *Objective image analysis of the meibomian gland area*, Article in The British journal of ophthalmology

[2] Pult H., Riede-Pult H. (2012). *Non-contact meibography in determination and treatment of non-obvious meibomian gland dysfunction*, Journal of Optometry Vol. 5

[3] Qiao J., Yan X. (2013). *Emerging treatment options for meibomian gland dysfunction*, www.ncbi.nlm.nih.gov NCBI Pubmed

[4] Wise R., Sobel R., and Allen R. (2012). *Meibography: A review of techniques and technologies* www.ncbi.nlm.nih.gov, NCBI Pubmed

[5] Nicholas DL, Gillan, WDH. (2015). *Meibomian gland imaging: A review*, ResearchGate

[6] Foulks, Borchman D. (2010). *Meibomian gland dysfunction: the past, present, and future*, www.ncbi.nlm.nih.gov NCBI Pubmed

[7] Vigo L., Giannaccare G, Sebastiani S, Pellegrini M., Carones F. (2019). *Intense Pulsed Light for the Treatment of Dry Eye Owing to Meibomian Gland dysfunction.*, www.ncbi.nlm.nih.gov NCBI Pubmed

[8] Whitman.edu, 5.1 Maxima and Minima. [online] Available at: https://www.whitman.edu/mathematics/calculus_online/section05.01.html [Accessed 3 Apr. 2019].

[9] Giavarina D. (2019). *Understanding Bland Altman analysis*, www.ncbi.nlm.nih.gov BIOCHEMIA MEDICA

[10] Nelson J., Shimazaki J., Benitez-del-Castillo M., Craig P., McCulley P., Den S., Foulks N (2011). *The International Workshop on Meibomian Gland Dysfunction: Report of the Definition and Classification Subcommittee*, www.ncbi.nlm.nih.gov IOVS

[11] Wallis Filter. [online] Available at: https://www.microimages.com/documentation/TechGuides/55Wallis.pdf [Accessed 1 May 2019].

CURRICULUM VITAE

Personal Information

Name Surname	: Alp Eren Bütün
Place and Date of Birth	: Samsun / 13 Feb 1989

Education

Undergraduate Education	: Atılım University Software Engineering
Graduate Education	: Kadir Has University Computer Engineering
Foreign Languages Skills	: English

Work Experience

Name of Employer and Dates of Employment:

- Noktacom Medya İnternet Hizmetleri A.Ş. (2011 - 2013)
- Comodo Yazılım A.Ş. (2013 - 2014)
- Brilliant Bridge (2014)
- Neptün Yazılım (2014 - 2015)
- W3 Digital (2015)
- XTB Trade A.Ş. (2016)

Contact:

Telephone	: 05301783445
E-mail Address	: alperenbutun@gmail.com

APPENDIX

Test Results for Manual Determination

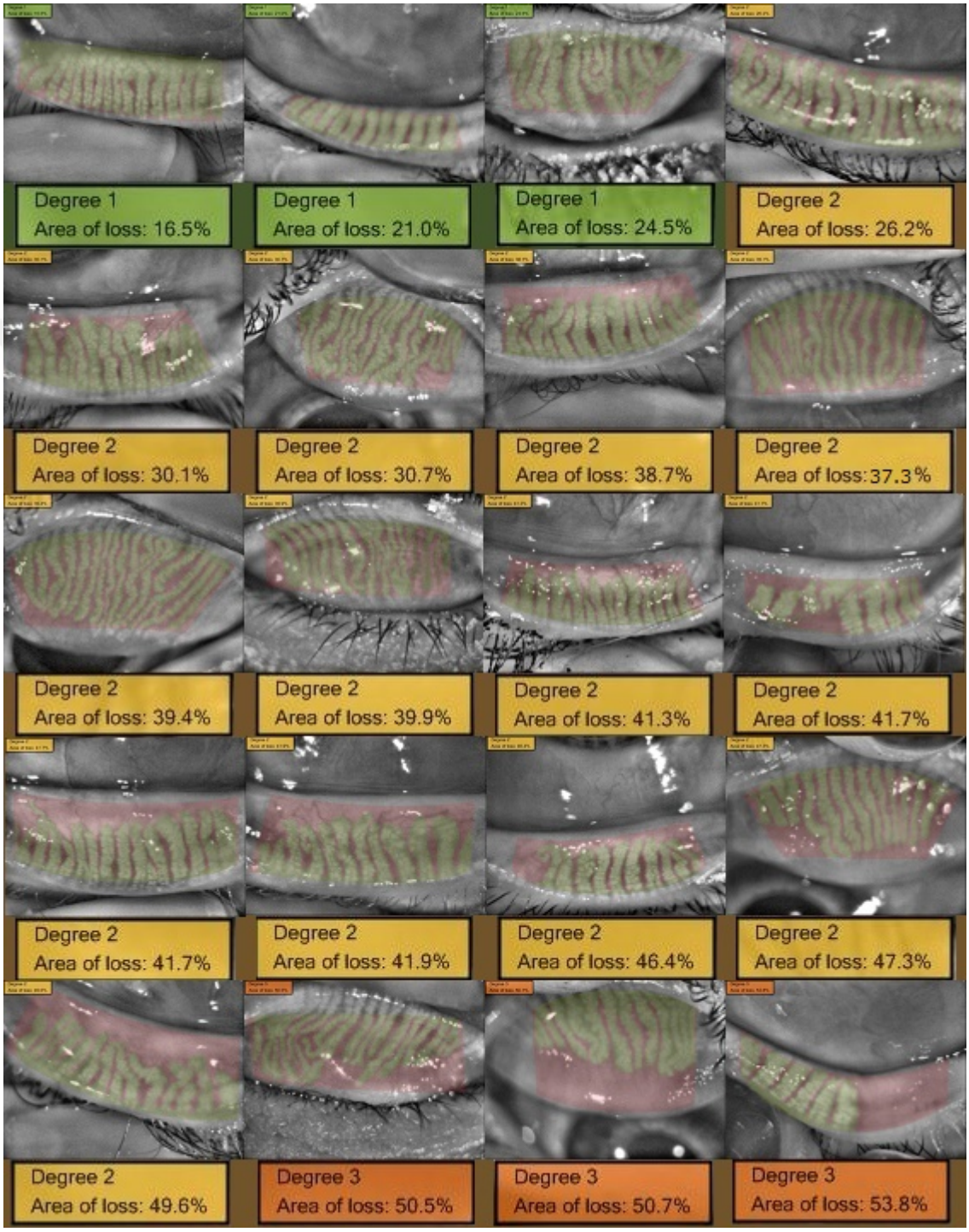

Figure 6.1 Test Results for Manual Determination