A Project Report

On

DETERMINING METRICS FOR RECONSTRUCTION OF CORNEA USING IMAGE PROCESSING AND NEURAL NETWORK MODELLING

BY

ABISHEK KRISHNAN
2011B5A3511H

Under the supervision of

Prof. P.K.THIRUVIKRAMAN
Dr. C JAGADESH REDDY

SUBMITTED IN PARTIAL FULLFILLMENT OF THE REQUIREMENTS OF

BITS F423T: FIRST DEGREE THESIS

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE PILANI (RAJASTHAN)

HYDERABAD CAMPUS

(APRIL 2016)

ACKNOWLEDGMENTS

Firstly, I am grateful to **Department of Physics, BITS-Pilani Hyderabad Campus** for offering a First Degree Thesis as a course in the curriculum.

I would like to express my sincere gratitude to Professor **P.K.Thiruvikraman**, Department Of Physics, BITS–Pilani Hyderabad Campus for giving me the opportunity to carry out a thesis in this esteemed organization.

I would like to express my sincere gratitude to **Dr. C Jagadesh Reddy (L.V.Prasad Eye Institute, Hyderabad)**, mentor for suggesting the project and providing valuable guidance and support throughout the course of work.

Birla Institute of Technology and Science-Pilani,

Hyderabad Campus

Certificate

This is to certify that the project report entitled "**DETERMINING METRICS FOR RECONSTRUCTION OF CORNEA USING IMAGE PROCESSING AND NEURAL NETWORK MODELLING**" submitted by Mr. ABISHEK KRISHNAN (ID No. 2011B5A3511H) in partial fulfillment of the requirements of the course BITS F423T – First Degree Thesis, embodies the work done by him under my supervision and guidance.

Date: 30.04.2016 **(P.K.THIRUVIKRMAN)**

BITS- Pilani, Hyderabad Campus

ABSTRACT

This thesis seeks to predict the thickness of cornea of eyes of a patient who will undergo corneal reconstructive surgery. The thesis focuses on a) Measuring the thickness of a cornea in the pre-surgery & post-surgery stage and those of the donor from images of cornea obtained using OCT (Optical coherence tomography) using image processing techniques and use the results to b) Predict the thickness of cornea of reconstructed eye using neural network modelling implemented in MATLAB environment. Results thus obtained predict the thickness of cornea of a reconstructed eye with error of +1.09 %. The model built in this thesis provides some vital metrics which can be used in future reconstructive corneal surgery.

CONTENTS

Table of Figures

1. Human Eye and Visual Process

Eye is the most valuable sense organ. The below image shows the important parts of an eye.

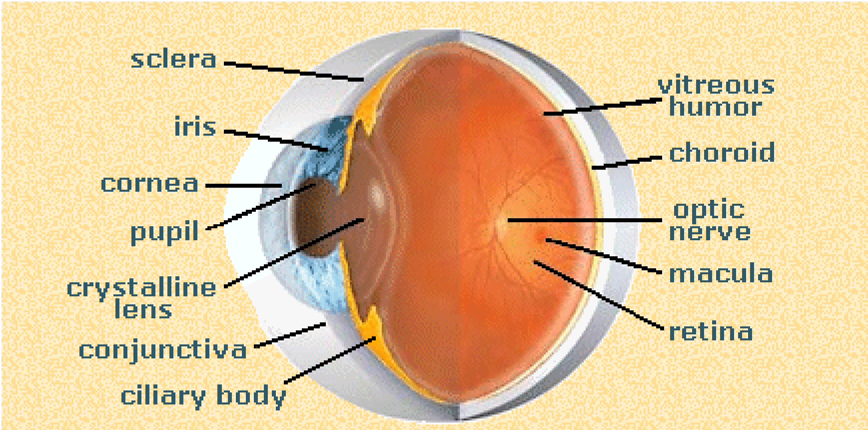

Figure 1.1 Anatomy of Human Eye

Steps involved in the process of visual experience **[1]**

1. The light rays enter the eye through the cornea which converges/bends the light.

2. The light then progress through the pupil the centre part of the coloured iris.

    2. a. Pupil changes its size to accommodate for the change in Intensity from smaller (for Intense light) to large (for dimmer light).

    2. b. Iris changes its curvature to refocus the object on the retina whose images initially were formed behind the retina.

3. The image which was converged first by the cornea is converged further by the crystalline lens where just behind it the image gets inverted.

4. The light rays travels through the vitreous humor and to retina (macula is the central region of the eye where the best visual location resides).

5. The retina converts these light impulses to electrical signals which are sent via the optic nerve to the occipital cortex located at the posterior of the brain.

1.1) Cornea

It is a transparent dome like structure and also a powerful structure which focuses the incoming light rays.

The typical dimensions of the cornea can be seen from the below figure:-

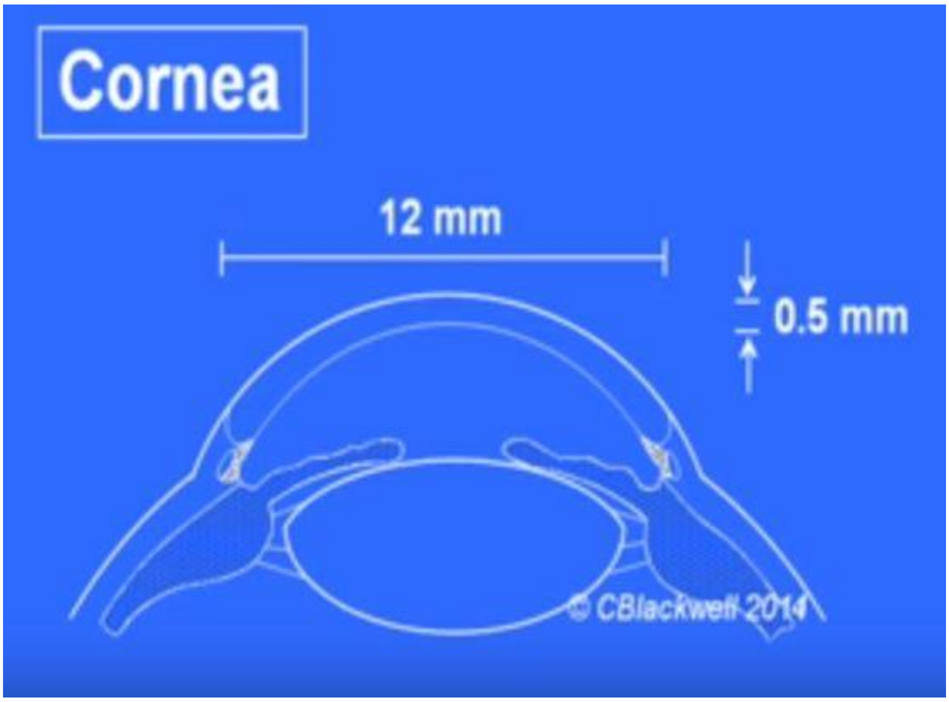

Figure 1.2 Dimensions of a cornea

It consists mainly of 5 parts:-

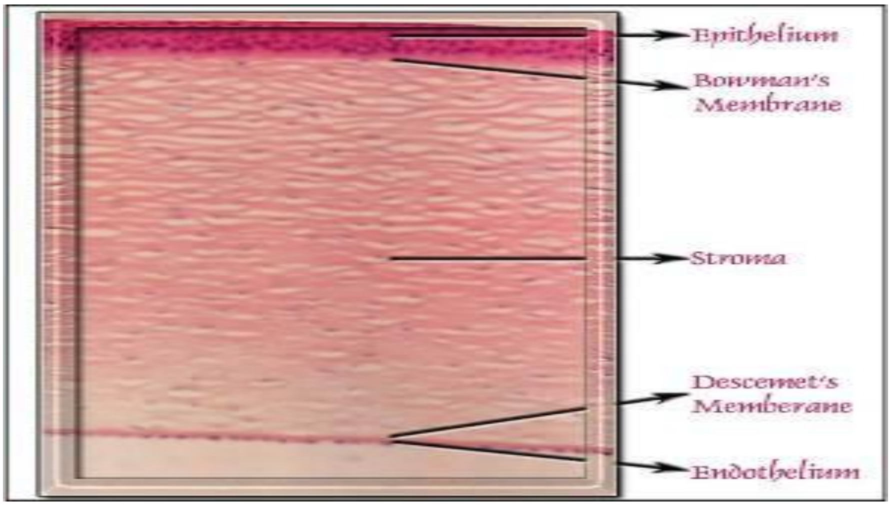

Figure 1.3 Parts of a cornea

1. Epithelium

It is the outer most layer of the cornea and is made up of the tissue's which make up the skin thereby it also has a lifecycle like skin cells which is typically from 7-10 days to form and then to eventually die.

Its functions include:-

a) To protect eye from foreign materials like dust.

b) To absorb oxygen and cell nutrients from tears and then redistribute them.

2. Bowman's Membrane

It is a transparent tissue which has strong layer of protein fibers known as collagen.

3. Stroma

It forms about 90% thickness of the cornea and mainly consists of water and collagen.

The collagen's spatial shape, arrangement leads to the light conducting transparency property of the cornea.

4. Descemet's Membrane

It is strong tissue composed of collagen fibers which are different from the ones present in the Stroma.

Its function is protect the eye from injuries and even injured to recover in a short span of time.

5. Endothelium

It helps in removing excess fluid out of the Stroma so it basically helps in maintaining the balance of the fluid moving in and out of the cornea.

The shape of the cornea is prolate spheroid. To understand the shape of the cornea takes a sphere and one where it is a parabola.

2. DAMAGES TO THE CORNEA AND OCT (OPTICAL COHERENCE TOMOGRAPHY)

2.1 Some of the diseases and disorders of the cornea are [2]

1) Allergies. Allergies affecting the eye are fairly common. The most common allergies are those related to pollen, particularly when the weather is warm and dry. An increasing number of eye allergy cases are related to medications and contact lens wear.

2) Conjunctivitis (Pink Eye). This term describes a group of diseases that cause swelling, itching, burning, and redness of the conjunctiva, the protective membrane that lines the eyelids and covers exposed areas of the sclera, or white of the eye. Conjunctivitis can be caused by a bacterial or viral infection, allergy, environmental irritants, a contact lens product, eye drops, or eye ointments.

3) Corneal Infections. Sometimes the cornea is damaged after a foreign object has entered the tissue, such as from a poke in the eye. At other times, bacteria or fungi from a contaminated contact lens can pass into the cornea. Situations like these can cause painful inflammation and corneal infections called keratitis. These infections can reduce visual clarity, produce corneal discharges, and perhaps erode the cornea. Corneal infections can also lead to corneal scarring, which can impair vision and may require a corneal transplant.

These disorders and diseases lead to cornea being subjected to spherical aberration. The phenomenon can be explained as follows:-

Cornea can be considered as spherical lens, therefore when a ray that is on the axis will go through the focus. A ray that is very close to the axis will still come to the focus very well. But as we go farther out, the ray begins to deviate from the focus, perhaps falling short and a ray striking near the top edge comes down and misses the focus by quite a wide margin. So instead of getting a point image, we get a smear **[3]**

2.2 OCT (Optical Coherence Tomography)

It is variation of a Michelson Interferometer setup where in

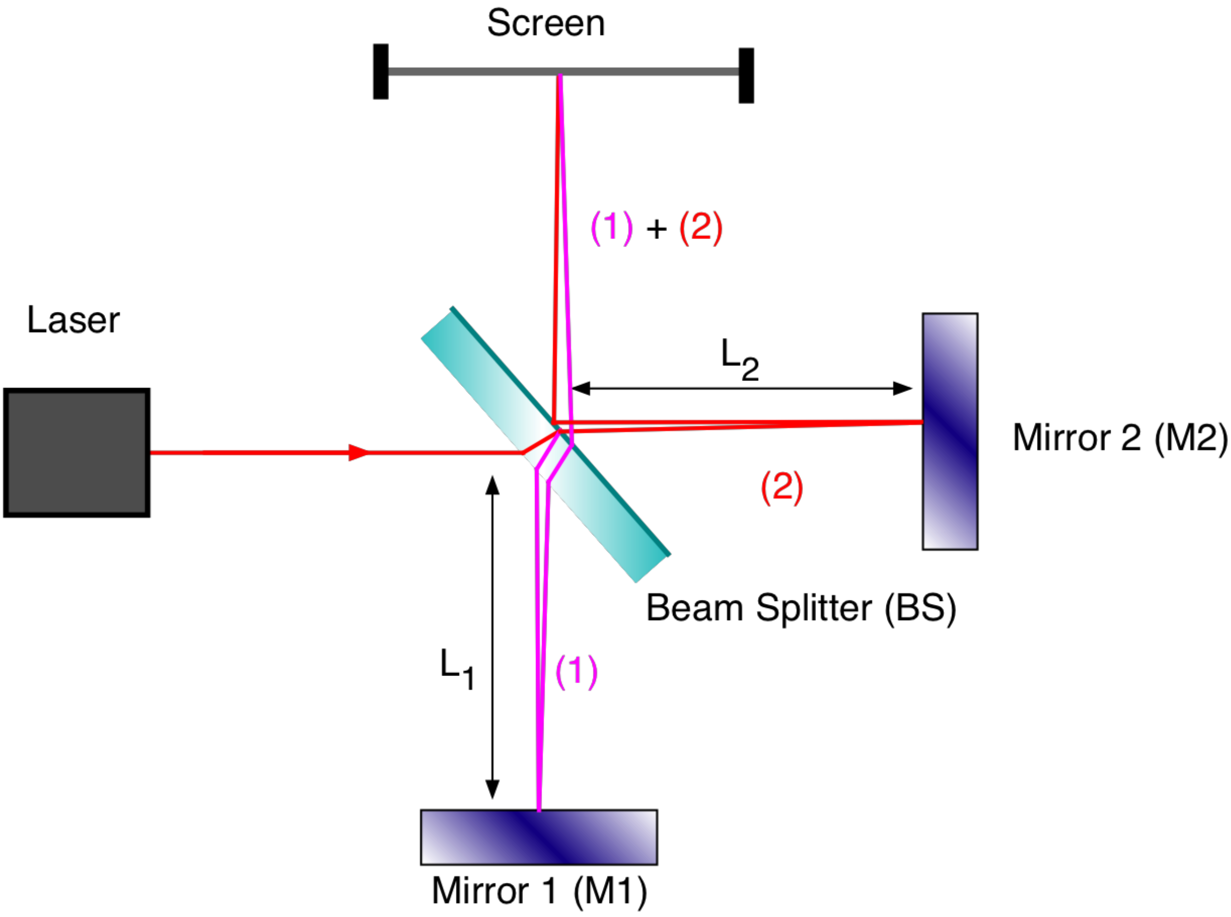

Figure 2.1 Michelson Interferometer Setup

Mirror 2 is replaced by the corneal tissue and Laser is replaced by pulses of laser and image is obtained in Screen.

It divides an incoming beam of radiation into two equal (ideal) parts with each part continuing along a separate path. When the two beams are recombined, a condition is created under which interference can take place. Interference occurs when two beams of radiation are added together or combine to form one summation signal. [4]

OCT is a non-invasive imaging technique that uses light waves to take cross-section pictures of the specimen (Cornea).As a result of diseases and disorders, the cornea which is subjected to spherical aberration tends to display non-uniformity in the thickness between top and bottom portions. With OCT each of the cornea’s distinct layers can be obtained and can be used for mapping and measuring the thickness.

3. THICKNESS OF CORNEA USING IMAGE PROCESSING

As we have seen in Section 2, the result of disorders and diseases to the cornea manifests in the non-uniformity of the distance between the top and bottom portions of the cornea. With the help of OCT we obtain the cross-sectional view of the cornea. This section, details the steps involved to automate the measurement of thickness of cornea from the image obtained via OCT.

This following are the steps involved in MATLAB **[5]** for measuring the thickness of Cornea (Step-1 Edge Detection and Step-2 Measuring the thickness)

3.1.a) A Sample Image of Cornea

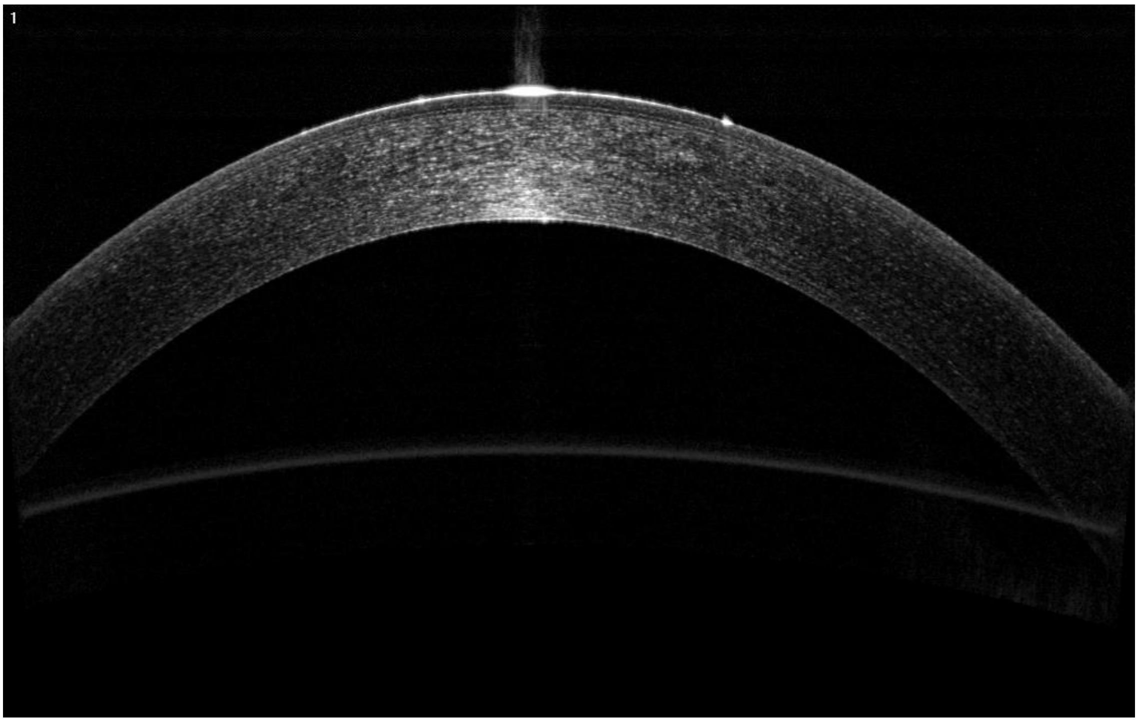

Figure 3.1 A sample image of cornea obtained using OCT.

3.1b) Convert the given into binary image using grey threshold level

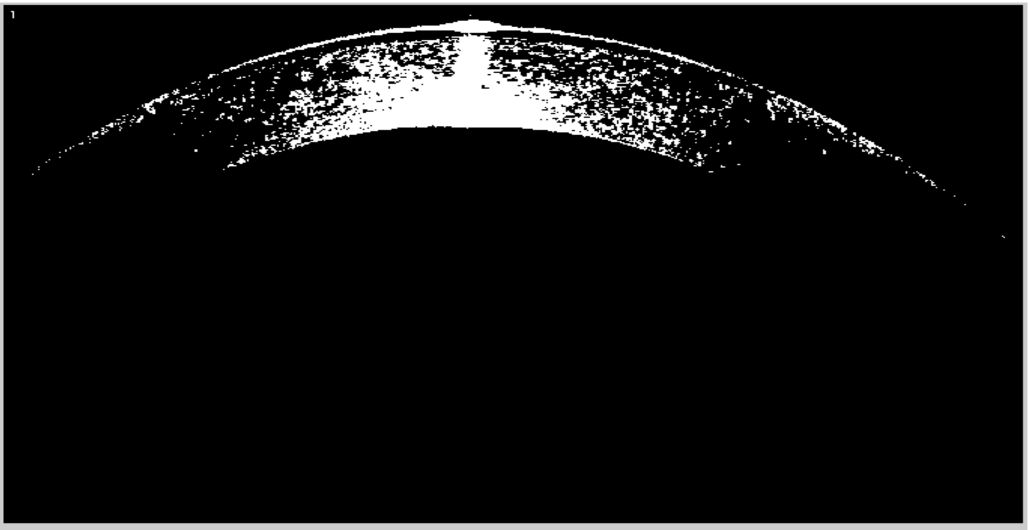

Figure 3.2 Binary image of the sample image

3.1c) Remove salt and pepper noise

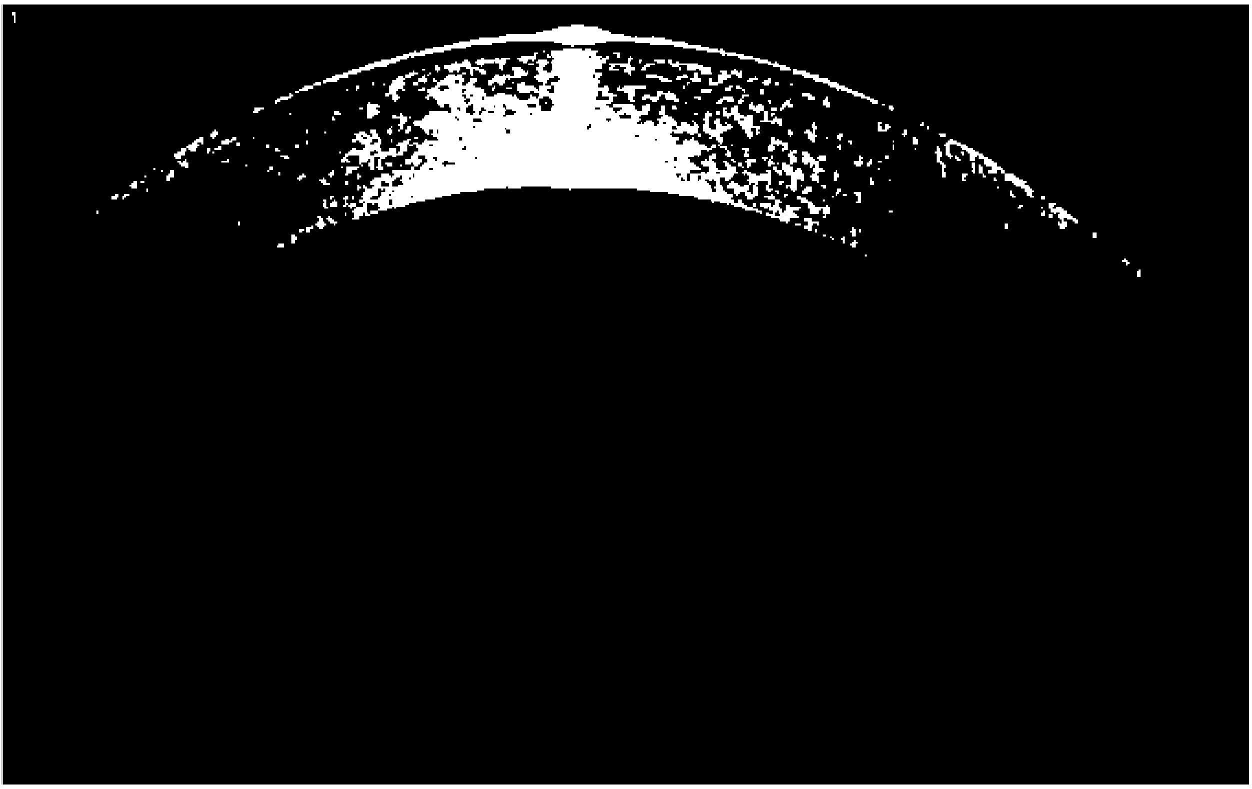

Figure 3.3 Binary image wherein salt & pepper noise has been removed

3.1d) Binary complement the image

Figure 3.4 Binary complement of the noiseless image

3.1e) Apply Sobel operator for edge detection

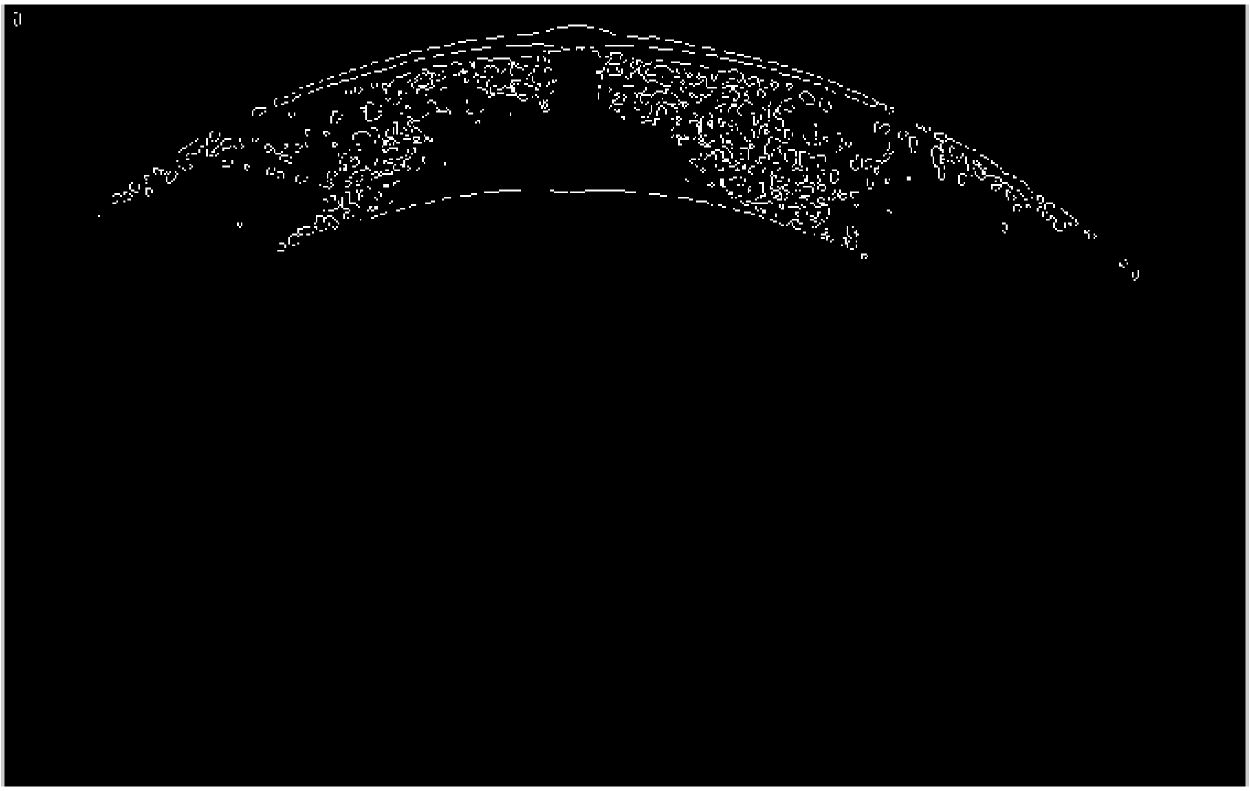

Figure 3.5 Edge detected binary image

3.1.f) Fill the gaps in the resultant image

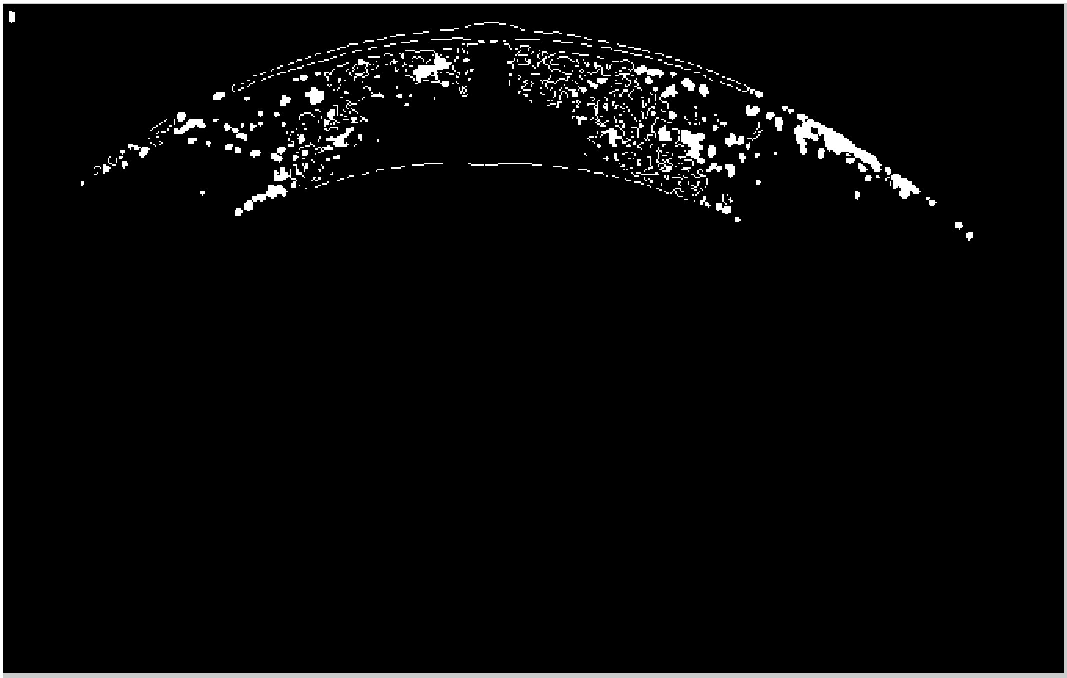

Figure 3.6 Edge detected binary image with gaps filled

3.1.g) Apply the Sobel filter once again and dilate the image for second step which is to measure thickness of the Cornea

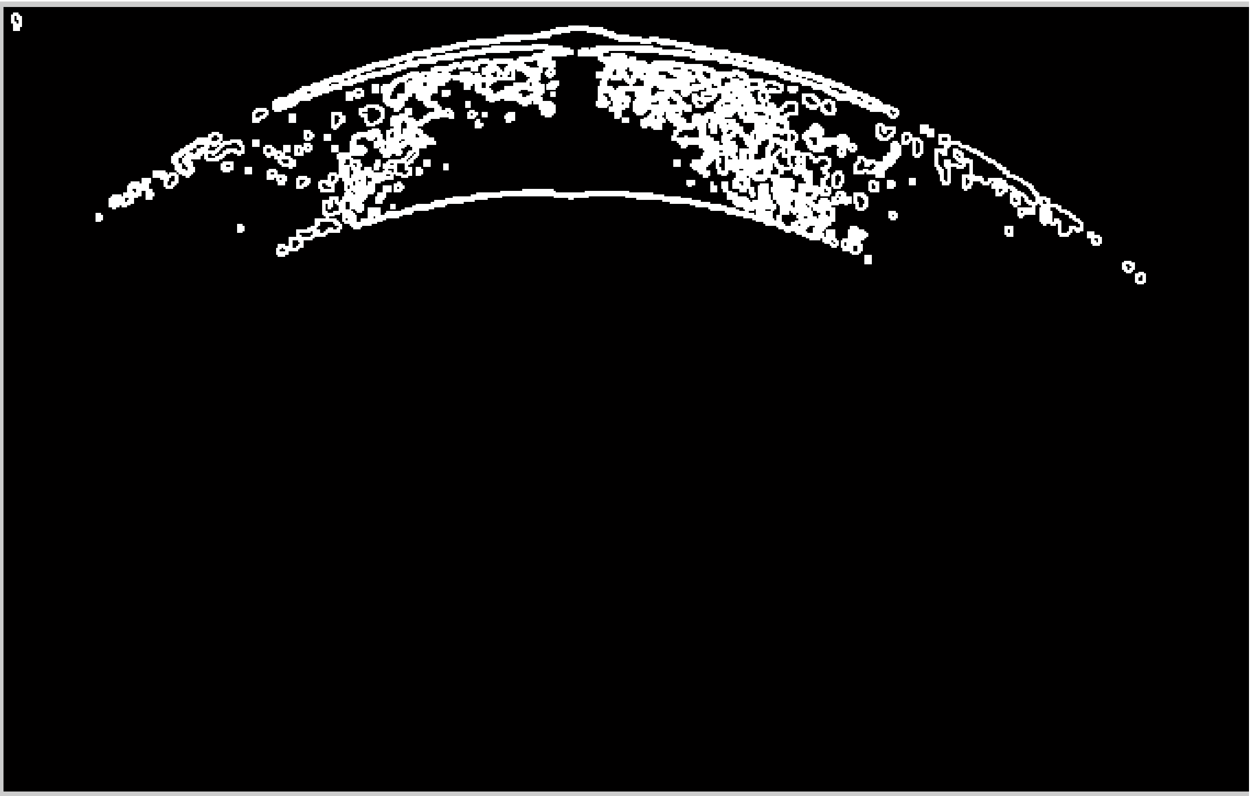

Figure 3.7 Dilated image to improve edge detection

Refer to APPENDIX A (Page 32) for the MATLAB code for the Edge Detection.

Step-2 involved extracting the thickness of the cornea after detecting the edges the algorithm is as follows:-

1) First, move row wise from the top until we encounter an white pixel and store the coordinate value and move to the next column
2) Repeat Step-1 from bottom.
3) Polynomial fit the top and bottom lines obtained and measure the thickness from the middle of the column or as suited for certain images.
4) Plot the distance between the top and bottom portion of the cornea .This information is needed for predictive analysis.
5) Variations of the above procedure were used in situations where the results obtained were not satisfactory.

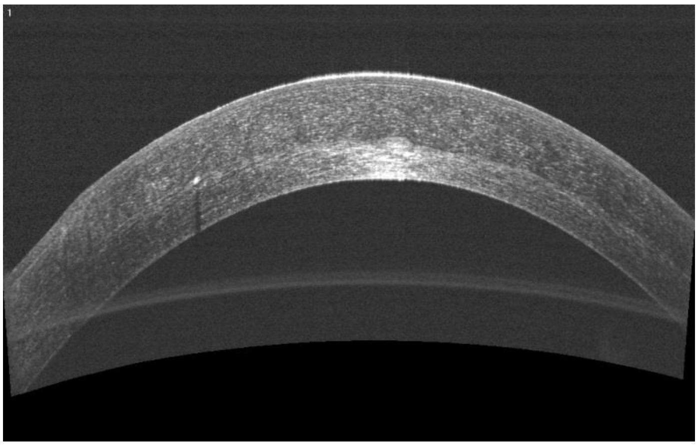

Figure 3.8 Sample image with Gaussian noise

3.2.1) For images of cornea with Gaussian noise, noise was removed by using a wiener filter and then the algorithm was implemented on the resultant image.

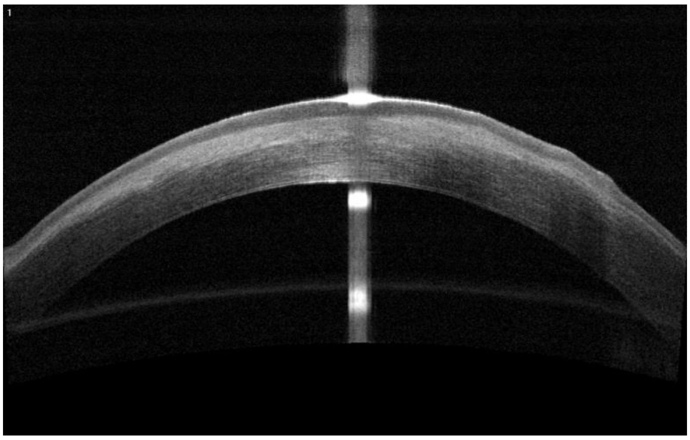

Figure 3.9 Sample image with Instrument Noise

3.2.2) There were many images of the cornea of the patient wherein the central part of the image was corrupted by noise arising due to instrument. Here the thickness was calculated as the average of the values obtained 50 pixels to the right and left of the centre of the image.

3.2.3a) Extract and dilate

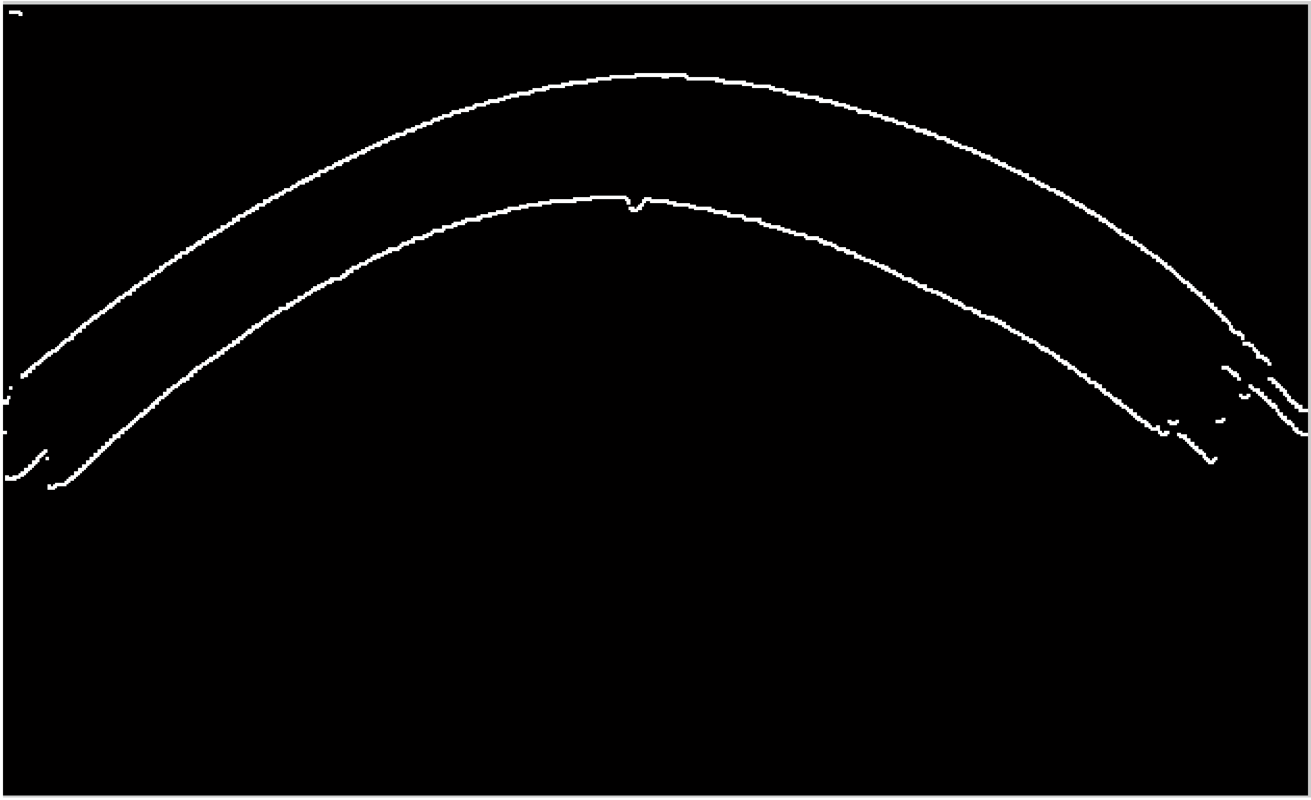

Figure 3.10 Extracted and dilated binary image

3.2.3b) Polynomial Fitting and dilation

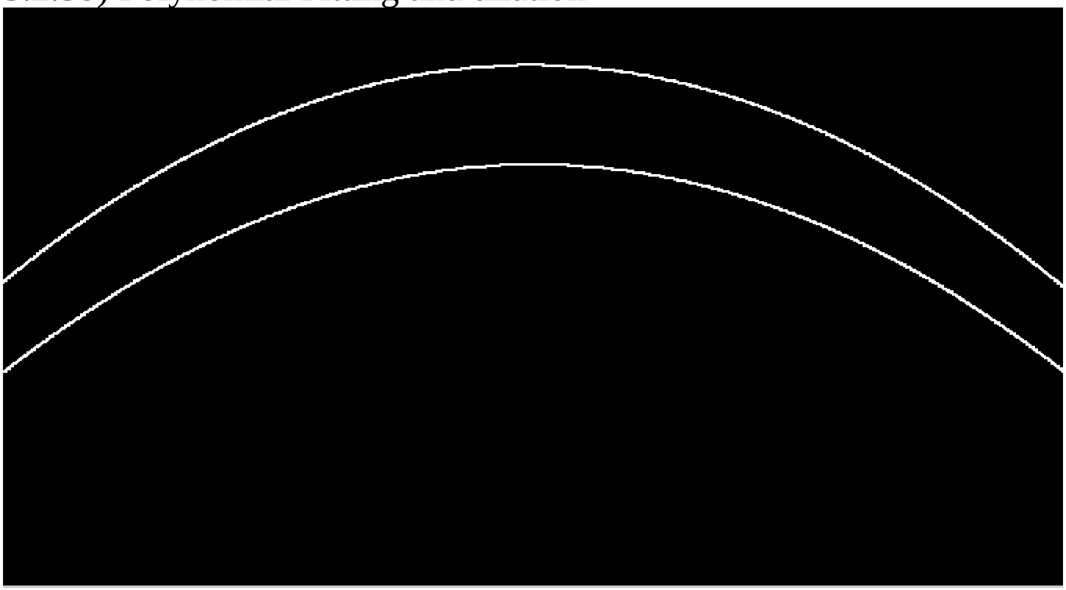

Figure 3.11 Polynomial fitted image of a cornea

3.2.3c) Plot of distance variation (Distance from L->R vs. Pixel Value of the width)

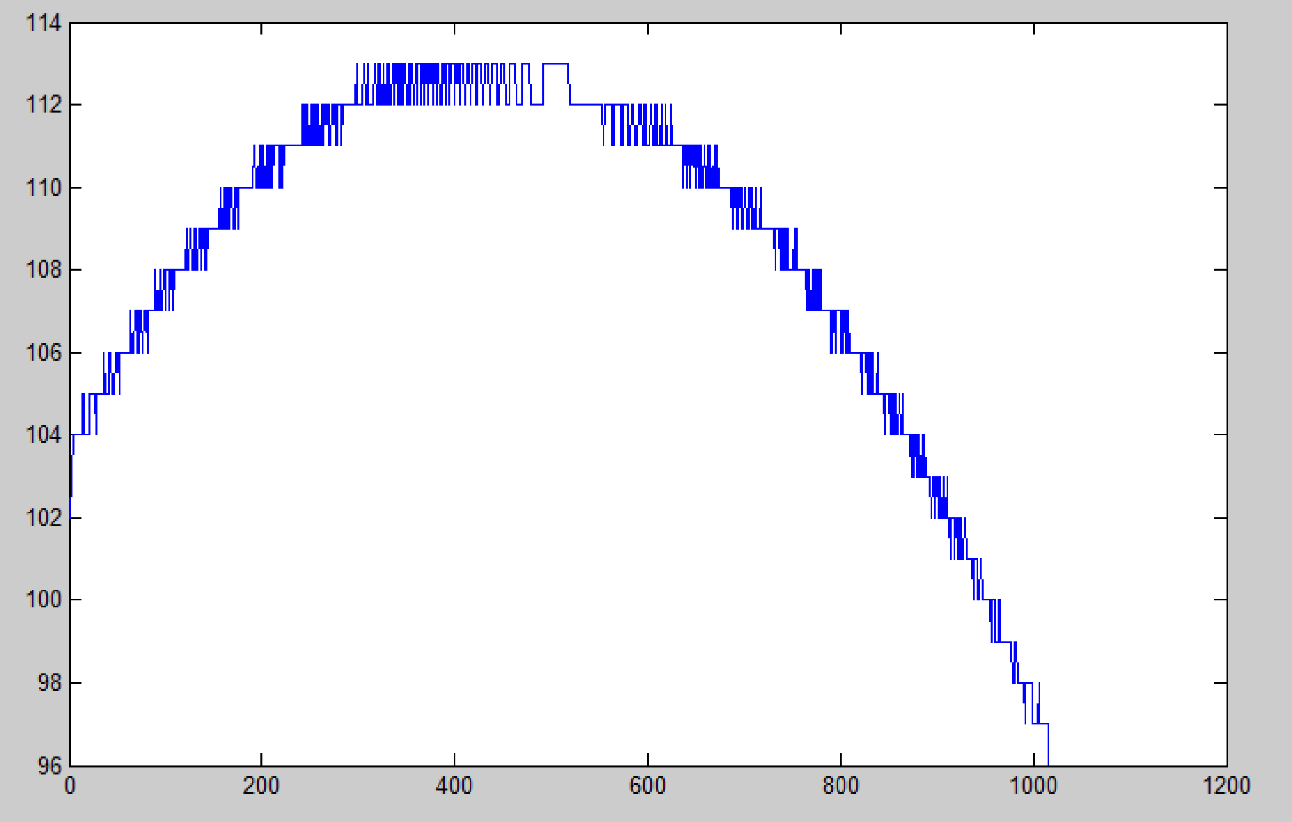

Figure 3.12 Plot of distance variation between top & bottom portions of cornea

Refer to APPENDIX A (Page 33) for the MATLAB code for the measuring the thickness of cornea.

3.3a) The following tables indicate the thickness of the cornea obtained by applying the image processing algorithm

Patient No.	Thickness of Normal (Other Eye) (µm)	Thickness of Post-Surgery (µm)	Percentage Change (%)
1	492.1063	461.2228	-6.27
2	469.4584	487.9885	3.95
3	499.3	494.1652	-1.03
4	499.3	547.6966	9.69
5	499.3	560.05	12.16
6	499.3	485.93	-2.67
7	499.3	490.05	-1.85
8	499.3	537.40	7.63
9	499.3	535.34	7.21
10	499.3	520.93	4.33

3.4) The following table compares the thickness of cornea between a donor (or normal) eye and post-surgery eye.

Patient No.	Thickness of Normal (Other Eye) (µm)	Thickness of Post-Surgery (µm)	Percentage Change (%)
1	492.1063	461.2228	-6.27
2	469.4584	487.9885	3.95
3	499.3	494.1652	-1.03
4	499.3	547.6966	9.69
5	499.3	560.05	12.16
6	499.3	485.93	-2.67
7	499.3	490.05	-1.85
8	499.3	537.40	7.63
9	499.3	535.34	7.21
10	499.3	520.93	4.33

Overall Average Change (%) = **+3.32** (Increase from the Normal Thickness)

4) Neural Network modelling for predicting the thickness of cornea

4.1) Review of basic approximation techniques

4.1a) Response surface methodology

Response surface methodology (RSM) is a method of constructing approximations of the system behavior using results of the response analysis carried out at a series of points in the design variable space. The approximation functions are obtained by the least-squares method. The strength of the technique is in application to problems where the design sensitivity information is difficult or impossible to obtain, as well as in cases where the response function values contain some level of computational noise. [6]

4.1b) Genetic Programming

Genetic programming (GP) is an automated method for creating a working computer program from a high-level problem statement of a problem. Genetic programming starts from a high-level statement of “what needs to be done” and automatically creates a computer program to solve the problem. [7]

4.1c) Neural Networks

Neural Networks (NN) are important data mining tool used for classification and clustering .It is an attempt to build a machine that will mimic brain activities and be able to learn. NN usually learns by examples. If NN is supplied with enough examples, it should be able to perform classification and even discover new trends or patterns in data. [8]

4.1d) Universal approximation theorem

In the mathematical theory of artificial neural networks, the universal approximation theorem states that a feed-forward network with a single hidden layer containing a finite number of neurons (i.e., a multilayer perceptron), can approximate continuous functions on compact subsets of Rn, under mild assumptions on the activation function. The theorem thus states that simple neural networks can *represent* a wide variety of interesting functions when given appropriate parameters; however, it does not touch upon the algorithmic learnability of those parameters. [9]

One of the first versions of the theorem was proved by George Cybenko in 1989 for sigmoid activation functions.

Given the reasoning behind the Universal approximation theorem and the implementation of neural networks in MATLAB environment we choose to perform the function approximation for predicting the thickness of cornea using neural networks.

4.2) Basics of the neural networks

Neural networks are typically organized in layers. Layers are made up of a number of interconnected 'nodes' which contain an 'activation function'. Patterns are presented to the network via the 'input layer', which communicates to one or more 'hidden layers' where the actual processing is done via a system of weighted 'connections'. The hidden layers then link to an 'output layer' where the answer is output as shown in the graphic below. [10]

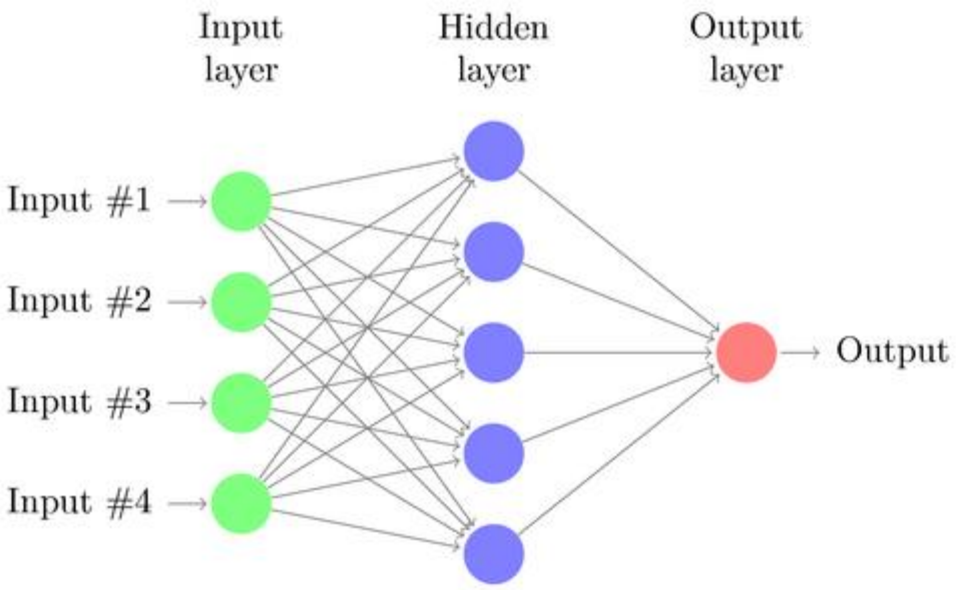

Figure 4.1 Simple Model for Neural Network

4.3) Back-Propagation Algorithm

Most ANNs contain some form of 'learning rule' which modifies the weights of the connections according to the input patterns that it is presented with. In a sense, ANNs learn by example as do their biological counterparts; a child learns to recognize dogs from examples of dogs.

Although there are many different kinds of learning rules used by neural networks, this demonstration is concerned only with one; the delta rule. The delta rule is often utilized by the most common class of ANNs called 'back-propagation neural networks' (BPNNs). Back-propagation is an abbreviation for the backwards propagation of error.

With the delta rule, as with other types of back-propagation, 'learning' is a supervised process that occurs with each cycle or 'epoch' (i.e. each time the network is presented with a new input pattern) through a forward activation flow of outputs, and the backwards error propagation of weight adjustments. More simply, when a neural network is initially presented with a pattern it makes a random 'guess' as to what it might be. It then sees how far its answer was from the actual one and makes an appropriate adjustment to its connection weights.

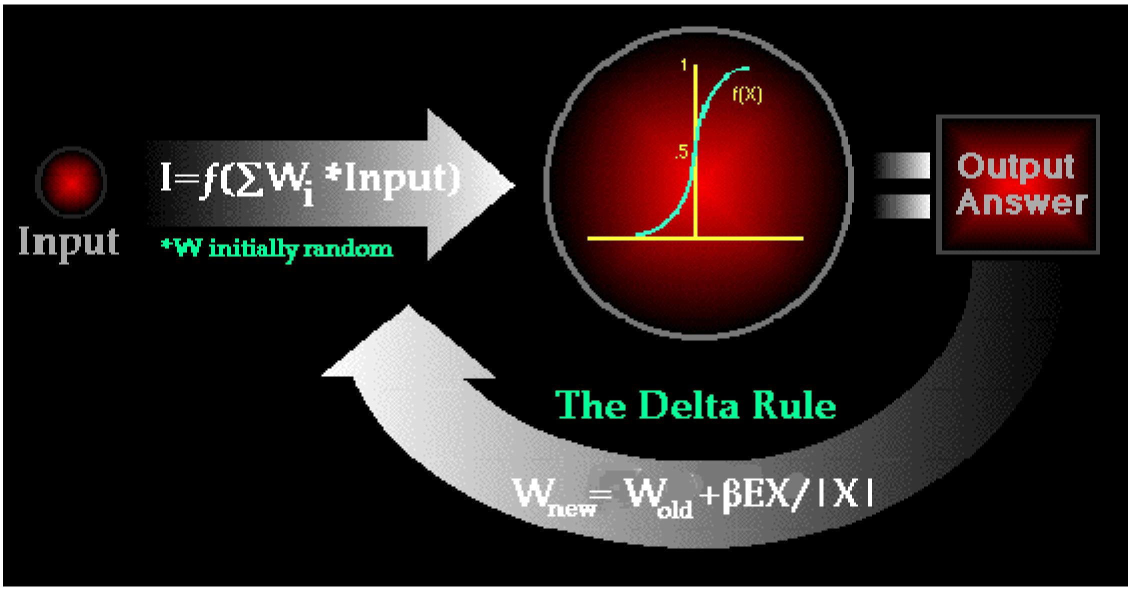

Figure 4.2 – A single node example

4.4) Levenberg Marquardt Algorithm

4.4.1) Problem Statement

The primary application of the Levenberg–Marquardt algorithm is in the least squares curve fitting problem: given a set of m empirical datum pairs of independent and dependent variables, (x_i, y_i), optimize the parameters $\beta$of the model curve $f(x,\boldsymbol{\beta})$ so that the sum of the squares of the deviations

$$S(\boldsymbol{\beta}) = \sum_{i=1}^{m} [y_i - f(x_i,\ \boldsymbol{\beta})]^2$$ becomes minimal.

4.4.2) Solution

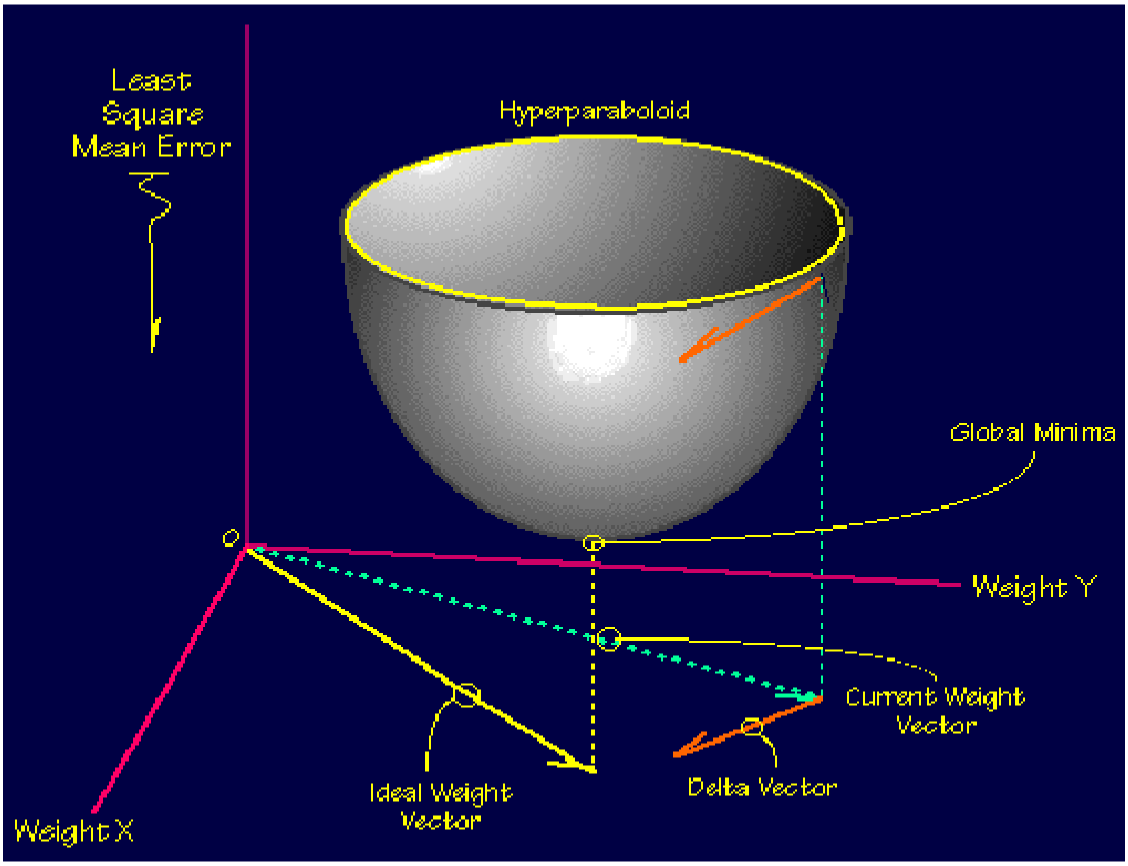

Figure 4.3 – Levenberg- Marquardt Algorithm – 3D representation

Like other numeric minimization algorithms, the Levenberg–Marquardt algorithm is an iterative procedure. To start a minimization, the user has to provide an initial guess for the parameter vector, β. In cases with only one minimum, an uninformed standard guess like $\beta^T = (1, 1... 1)$ will work fine; in cases with multiple minima, the algorithm converges to the global minimum only if the initial guess is already somewhat close to the final solution.[11][12]

In each iteration step, the parameter vector, β, is replaced by a new estimate, $\beta + \delta$. To determine δ, the functions $f(x_i, \boldsymbol{\beta} + \boldsymbol{\delta})$ are approximated by their linearization

$$f(x_i, \boldsymbol{\beta} + \boldsymbol{\delta}) \approx f(x_i, \boldsymbol{\beta}) + J_i \boldsymbol{\delta}$$

Where

$$J_i = \frac{\partial f(x_i, \boldsymbol{\beta})}{\partial \boldsymbol{\beta}}$$

Is the gradient (row-vector in this case) of f with respect to β.

At the minimum of the sum of squares, $S(\beta)$, the gradient of S with respect to β will be zero. The above first-order approximation of $f(x_i, \boldsymbol{\beta} + \boldsymbol{\delta})$ gives

$$S(\boldsymbol{\beta} + \boldsymbol{\delta}) \approx \sum_{i=1}^{m} (y_i - f(x_i, \boldsymbol{\beta}) - J_i \boldsymbol{\delta})^2.$$

Or in vector notation,

$$S(\boldsymbol{\beta} + \boldsymbol{\delta}) \approx \|\mathbf{y} - \mathbf{f}(\boldsymbol{\beta}) - \mathbf{J}\boldsymbol{\delta}\|^2.$$

Taking the derivative with respect to δ and setting the result to zero gives:

$$(\mathbf{J^T J})\boldsymbol{\delta} = \mathbf{J^T}[\mathbf{y} - \mathbf{f}(\boldsymbol{\beta})]$$

where $\mathbf{J}$ is the Jacobian matrix whose i^{th} row equals J_i, and where $\mathbf{f}$ and $\mathbf{y}$ are vectors with i^{th} component $f(x_i, \boldsymbol{\beta})$ and y_i, respectively. This is a set of linear equations which can be solved for δ.

Levenberg's contribution is to replace this equation by a "damped version",

$$(\mathbf{J^T J} + \lambda \mathbf{I})\boldsymbol{\delta} = \mathbf{J^T}[\mathbf{y} - \mathbf{f}(\boldsymbol{\beta})]$$

Where I is the identity matrix, giving as the increment, δ, to the estimated parameter vector, β.

The (non-negative) damping factor, λ, is adjusted at each iteration. If reduction of S is rapid, a smaller value can be used, bringing the algorithm closer to the Gauss–Newton algorithm, whereas if an iteration gives insufficient reduction in the residual, λ can be increased, giving a step closer to the gradient descent direction. Note that the gradient of S with respect to δ equals $-2(\mathbf{J}^T[\mathbf{y} - \mathbf{f}(\boldsymbol{\beta})])^T$. Therefore, for large values of λ, the step will be taken approximately in the direction of the gradient. If either the length of the calculated step, δ, or the reduction of sum of squares from the latest parameter vector, $\beta + \delta$, fall below predefined limits, iteration stops and the last parameter vector, β, is considered to be the solution.

Levenberg's algorithm has the disadvantage that if the value of damping factor, λ, is large, inverting $J^TJ + \lambda I$ is not used at all. Marquardt provided the insight that we can scale each component of the gradient according to the curvature so that there is larger movement along the directions where the gradient is smaller. This avoids slow convergence in the direction of small gradient. Therefore, Marquardt replaced the identity matrix, I, with the diagonal matrix consisting of the diagonal elements of J^TJ, resulting in the Levenberg–Marquardt algorithm:

$$(\mathbf{J^T J} + \lambda\, \mathbf{diag}(\mathbf{J^T J}))\boldsymbol{\delta} = \mathbf{J^T}[\mathbf{y} - \mathbf{f}(\boldsymbol{\beta})].$$

4.5) MATLAB implementation of neural network modelling

MATLAB provides a very high level graphical user interface (GUI) for neural network modelling. The steps involved are as follows

4.5a) nnstart – Starts the GUI for neural network

4.5b) Select the “Fitting Tool” for our purpose of function approximation – nftool
4.5c) Load the data for inputs and outputs (Already calculated) from the workspace which define our function approximation problem

[Input: Thickness of cornea of pre-surgery (damaged) eye (10)
Output: Thickness of cornea of post-surgery eye (10)]

4.5d) Set aside data for validation and testing
Training – 6 samples and Validation and Testing – 2 samples each

4.5e) Define the network architecture – Number of hidden neurons (1-10)

4.5f) Train the network to the fit the inputs and targets – trainlm – using Levenberg-Marquardt back-propagation algorithm

4.5e) Deploy the solved neural network to Simulink which is used for simulation (predicting) the thickness of cornea of post-surgery eye given any input which represents the thickness of cornea of pre-surgery eye.

4.6) The following table compares the thickness of actual post-surgery thickness and predicted post-surgery thickness

Patient No.	Actual thickness Post Surgery (μm)	Predicted Thickness Post Surgery (μm)	Percentage Change (%)
1	461.22	477.27	+3.48
2	487.99	531.68	+8.95
3	494.16	528.79	+7.01
4	547.70	527.35	-3.71
5	560.05	490.81	-12.34
6	485.93	521.52	+7.32
7	490.05	512.30	+4.54
8	537.40	521.52	-2.95
9	535.40	520.02	-2.87
10	520.93	528.37	+1.43

Prediction Error (%) = **+1.09** (Increase from the Actual Thickness)

5) Further Theoretical Improvements

While developing the neural network model one of recurring chances of improvement was to predict the number of hidden neurons which as of now have been obtained by Trial & Error method.

A theoretical study of "*A Two Phase Method for Determining the Number of Neurons in the Hidden Layer of a 3-Layer neural Network*" – Kazuhiro Shin-ike, MNCT, Kyoto was undertaken which proceeds with a 2-Phase approach to solve the problem of determination of number of hidden layer neurons.

Phase-I: Candidates for the number of neurons is determined by using the back-propagation method.

Phase-II: Optimal number of neurons is determined by considering the generalization capacity.

The experiment implemented was to predict the accuracy of Ex-OR circuit where, 352 samples of inputs (2 voltages values as inputs) and output (Output voltage)

The result of the Hit & Trial method by choosing the number of hidden layers randomly was

No. of Neurons	Training Iteration
5	3059
10	1402
15	5434
20	2414
25	2221
30	2522

The application of Two-phase method

Step-1: Apply Hit & Trail for 2 groups of data for training in the 1st phase

No. of Neurons	No. of correct answers	Correctness Rate (%)
5	146	83
10	159	90.3
15	159	90.3
20	164	93.2
25	160	91
30	159	90.3

Selection criteria for the next phase was correctness rate greater than 90%

Step-II : We use new data which were not used to train the neural network and perform the simulations on it.

No. of Neurons	No. of correct answers
10	68
15	65
20	65
25	74
30	59

Finally, we select the number of hidden neurons which yields the maximum number of correct answers (25 in our case).

CONCLUSION

This report discusses the development of an image processing algorithm to determine the thickness of cornea of eyes of a patient in stage of pre-surgery (Damaged) & post-surgery (Reconstructed) and that of a donor (Normal).Further, a neural network based model is developed to predict the thickness of cornea post-surgery from the thickness of cornea obtained in the pre-surgery stage. Since the neural network is a learning based approach for function approximation, more instances of training data will improve the ability of the model to predict results accurately. Since the training data in this experiment comprises of images of cornea of patients in the pre-surgery stage, more such images will lead to improved results.

REFERENCES

[1] Anatomy and Visual Process. http://www.tedmontgomery.com/the_eye/

[2] National Eye Institute, USA https://nei.nih.gov/health/cornealdisease

[3] Feynman Lectures on Physics – Volume I

[4] Block Engineering, USA www.blockeng.com/technology/ftirtechnology.html

[5] Digital Image Processing Using MATLAB, 2e, Gonzalez and Woods

[6] Thesis – Luis, Burgeon and Toropov, Chapter-2, University of Bradford, UK.

[7] Genetic Programming www.genetic-programming.org

[8] Cilimkovic., M "Neural Networks and Back Propagation Algorithm", Institute of Technology Blanchard town, Dublin, Ireland

[9] Cybenko., G. (1989) "Approximations by superpositions of sigmoidal functions", *Mathematics of Control, Signals, and Systems*, 2 (4), 303-314

[10] University of Wisconsin: Department of Computer Sciences

http://pages.cs.wisc.edu/~bolo/shipyard/neural/local.html

[11] Kenneth Levenberg (1944). "A Method for the Solution of Certain Non-Linear Problems in Least Squares". *Quarterly of Applied Mathematics* **2**: 164–168.

[12] Donald Marquardt (1963). "An Algorithm for Least-Squares Estimation of Nonlinear Parameters". *SIAM Journal on Applied Mathematics* **11** (2): 431–441.

APPENDIX A

1. MATLAB code for the Edge Detection (Modifications of the code were used for certain images to provide better results in that specific case).

```
%To find the edges of the images.
%Author:Abishek Krishnan
clc;
close all;
clear all;
%Read all the images from a folder
warning('off', 'Images:initSize:adjustingMag'); %To turn off the warning the image is
to big to fit in this window
d = dir('*.jpg');
files = {d.name};
number_of_images = length(d);
%To segments for morphological operations
se1 = strel('line',3,0);
se2 = strel ('line',3,90);
se1e = strel('line',2,0);
se2e = strel ('line',2,90);
for k=1:number_of_images
    I{k} = imread(files{k});
    level = graythresh(I{k});
    level1 = graythresh(I{1});

    I_binary = im2bw(I{k},level);
    I_binary1 = im2bw(I{1},level1);

    I_noise_remove_salt_and_pepper = medfilt2(I_binary,[3 3]);
    I_noise_remove_salt_and_pepper1 = medfilt2(I_binary1,[3 3]);

    I_noise_removed = wiener2(I_noise_remove_salt_and_pepper,[3 3]);
    I_noise_removed1 = wiener2(I_noise_remove_salt_and_pepper1,[3 3]);

    I_binary_complement = imcomplement(I_noise_removed);
    I_binary_complement1 = imcomplement(I_noise_removed1);

    I_sobel = edge(I_binary_complement,'sobel');
    I_sobel1 = edge(I_binary_complement1,'sobel');

    I_filled_holes = imfill(I_sobel,'holes');
    I_filled_holes1 = imfill(I_sobel1,'holes');
```

```
    I_extra_sobel = edge(I_filled_holes,'sobel');
    I_extra_sobel1 = edge(I_filled_holes1,'sobel');

    I_dilated = imdilate(I_extra_sobel,[se1e se2e]);
    I_dilated1 = imdilate(I_extra_sobel1,[se1e se2e]);

    Result{k} = I_dilated; % or I_dilated1 depending on the input image
end
```

2. MATLAB code to measure the thickness of cornea.

```
%To measure the thickness of Cornea.
%Author:ABishek Krishnan
clc;
close all;
clear all;
warning('off', 'Images:initSize:adjustingMag'); %To turn off the warning the image is to big to fit in this window

I  = imread('ODa_after_extraction.jpg');
level = graythresh(I);
I_bin = im2bw(I,level);
[rows columns] = size(I_bin);
ad = zeros(rows,columns);
al = logical(ad);
alr = logical(ad);

count = 1;
rcount = 1;
se1 = strel('line',2,0);
se2 = strel ('line',2,90);

for c = 1:columns;
    for r = 1:rows;
        if(I_bin(r,c)==1)
            al(r,c) = 1;
            xset(count) = c;
            yset(count) = r;
            count= count +1;
            break;
        end
```

```
    end
end

for cr = 1:columns;
    for rr = rows:-1:1;
        if(al(rr,cr)~=1 && I_bin(rr,cr) == 1)
            alr(rr,cr) = 1;
            rcount = rcount+1;
            break;
        end;
    end;
end;
I_before_dilation_top = mat2gray(al);
I_extracted = imdilate(I_before_dilation_top,[se1 se2]);
I_before_dilation_bottom = mat2gray(alr);
I_bottom_extracted = imdilate(I_before_dilation_bottom,[se1 se2]);
imshow(I_extracted)
imshow(I_bottom_extracted)
I_final = I_extracted + I_bottom_extracted;
imshow(I_final)

%Polyfit
[r1, c1] = find(I_extracted);
figure;
plot(c1,r1,'.');
hold on;
f1 = fit(c1, r1, 'poly2');
plot((min(c1):max(c1)),f1(min(c1):max(c1)), 'red', 'LineWidth', 1);

[r2, c2] = find(I_bottom_extracted);
figure;
plot(c2,r2,'.');
hold on;
f2 = fit(c2, r2, 'poly2');
plot((min(c2):max(c2)),f2(min(c2):max(c2)), 'red', 'LineWidth', 1);


%Change of origin
x = (min(c1):max(c1))';
y = round(f1(x));
I1 = zeros(size(I_extracted));
I1(y +((x-1)*size(I1,1))) = 1;
figure
```

```
imshow(I1)

x1 = (min(c2):max(c2))';
y1 = round(f2(x));
I2 = zeros(size(I_bottom_extracted));
I2(y1 +((x1-1)*size(I2,1))) = 1;
figure
imshow(I2);

I3 = I1+I2;
imshow(I3)

%Trying to dilate the final image and apply a morph to it
I_f = imdilate(I3,[se1 se2]);
imshow(I_f)

%On this apply the up down algo and populate the array
[rf cf] = size(I_f);
adf = zeros(rf,cf);
alf = logical(adf);
alrf = logical(adf);
countf = 1;
rcountf = 1;

for c = 1:cf;
    for r = 1:rf;
        if(I_f(r,c)==1)
            alf(r,c) = 1;
            top(countf,:) = [c,r];
            countf= countf +1;
            break;
        end

    end
end

for cr = 1:cf;
    for rr = rf:-1:1;
        if(alf(rr,cr)~=1 && I_f(rr,cr) == 1)
            alrf(rr,cr) = 1;
            bottom(rcountf,:) = [cr,rr];
            rcountf = rcountf+1;
            break;
```

```
        end;
    end;
end;

count = countf-1;

%Subtracting the distance
for i = 1:count
    sub(i) = abs(top(i,2)-bottom(i,2));
end

%Plot the graph of difference in length

x = 1:count;
y = sub(x);
p = polyfit(x,y,2);
plot(x,y);
```