A

MAJOR PROJECT REPORT ON

GENERATIVE ADVERSARIAL NETWORKS FOR RETINAL IMAGE ENHANCEMENT WITH PATHOLOGICAL INFORMATION

Submitted in partial fulfillment of the requirement for the award of degree of

BACHELOR OF TECHNOLOGY

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

SUBMITTED BY

ROUTHU. AKHILESWARA RAO	**218R1A04O8**
SHAIK. MD. RABBANI	**218R1A04O9**
SHAIK. RIYAZ PASHA	**218R1A04P0**
SRIGADHI. BHANU TEJA	**218R1A04P1**

Under the Esteemed Guidance of

Mrs. K. VANI
Associate Professor

DEPARTMENT OF ELECTRONICS & COMMUNICATIONENGINEERING

CMR ENGINEERING COLLEGE

UGC AUTONOMOUS

(Approved by AICTE, Affiliated to JNTU Hyderabad, Accredited by NBA)
Kandlakoya(V), Medchal(M), Telangana –501401

(2024-2025)

CMR ENGINEERING COLLEGE

UGC AUTONOMOUS

(Approved by AICTE, Affiliated to JNTU Hyderabad, Accreditedby NBA)Kandlakoya (V), Medchal Road, Hyderabad - 501 401

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

CERTIFICATE

This is to certify that Major project work entitled **"GENERATIVE ADVERSARIAL NETWORKS FOR RETINAL IMAGE ENHANCEMENT WITH PATHOLOGICAL INFORMATION"** is being Submitted by **R. AKHILESWARA RAO bearing Roll** No: **218R1A04O8, S.MD. RABBANI** bearing Roll No**: 218R1A04O9, SK. RIYAZ PASHA** bearing Roll No: **218R1A04P0, S. BHANU TEJA** bearing Roll No: **218R1A04P1** in B.Tech IV-II Semester, Electronics and Communication Engineering is a record bonafide work carried out by then during the academic year 2024-25.

INTERNAL GUIDE
Mrs. K. VANI
Associate Professor

HEAD OF THE DEPARTMENT
Dr. SUMAN MISHRA
Professor & HOD

EXTERNAL EXAMINER

ACKNOWLEDGEMENTS

We sincerely thank the management of our college **CMR Engineering College** for providing required facilities during our project work. We derive great pleasure in expressing our sincere gratitude to our Principal **Dr. A. S. Reddy** for his timely suggestions, which helped us to complete the project work successfully. It is the very auspicious moment we would like to express our gratitude to **Dr. SUMAN MISHRA,** Head of the Department, ECE for his consistent encouragement during the progress of this project.

We take it as a privilege to thank our major project coordinator **Dr. T. SATYANARAYANA**, Professor, Department of ECE for the ideas that led to complete the project work and we also thank him for his continuous guidance, support and unfailing patience, throughout the course of this work. We sincerely thank our project internal guide **Mrs. K. VANI,** Associate Professor of ECE for guidance and encouragement in carrying out this project work.

DECLARATION

We hereby declare that the Major project entitled **"GENERATIVE ADVERSARIAL NETWORKS FOR RETINAL IMAGE ENHANCEMENT WITH PATHOLOGICAL INFORMATION"** is the work done by us in campus at **CMR ENGINEERING COLLEGE,** Kandlakoya during the academic year 2024-2025 and is submitted as major project in partial fulfillment of the requirements for the award of degree of **BACHELOR OF TECHNOLOGY in ELECTRONICS AND COMMUNICATION ENGINEERING FRO JAWAHARLALNEHRU TECHNOLOGICAL UNIVERSITY, HYDERABAD.**

R. AKHILESWARA RAO	**218R1A04O8**
S. MD. RABBANI	**218R1A04O9**
SK. RIYAZ PASHA	**218R1A04P0**
S. BHANU TEJA	**218R1A04P1**

ABSTRACT

Age- related macular degeneration (AMD) is a disease of the central retina, which is one of the main reasons for vision loss of elderly people. To monitor the level of AMD, the doctors mainly use the retinal fundus images. However, the quality of retinal images can be affected during the imaging process. It leads to low contrast and blurry images. Those bad quality images cannot be used for analyzing and diagnosis. For that reason, there are many studies about image enhancement in order to improve the quality of retinal photography. However, previous methods could not guarantee to keep the disease information after the enhancement process. Therefore, we introduce a generative adversarial model for AMD retinal image enhancement with additional factors to preserve the disease information. By exploiting drusen segmentation masks, our proposed model can enhance retinal photography quality and keep the pathological information.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER-1
INTRODUCTION

1.1 OVER VIEW

Age-related macular degeneration (AMD) encompasses a variety of diseases and conditions. It is a group of optic nerve diseases, with 'characteristic' progressive structural changes leading to loss of visual function in a 'characteristic' way. AMD is the second leading cause of blindness worldwide. The retina is the innermost layer in the eye and the retinal nerve fibers transmit the visual signal from the photoreceptors in the eye to the brain via the bundle going out of the eye, known as the optic nerve. AMD leads to continuous and speedy damage of the retinal nerve fiber layer and hence can lead to permanent blindness. The progression of the nerve fiber layer loss can be effectively stopped by treatment consisting of medication or surgery to reduce the intraocular pressure. Hence the diagnosis of AMD at an earlier stage is very important for its treatment.

A major concern with AMD detection is that the disease has no particular set of physical causes or symptoms that doctors can recognize to detect the disease in an early stage. The main focus in AMD diagnosis is to detect changes in the visual functioning of the eye at early stages of the disease so that vision can be protected and preserved through medical treatment. It has been proved that the development of visual field defects is preceded by RNFL damage in Age-related macular degeneration (AMD).

Studies show that as much as 40% of retinal nerve fiber in the eye can be lost without the detection of characteristic visual defect in AMD patients. Hence it is believed that the detection of damage in nerve fiber layer can lead to an early detection of AMD. Several computer-assisted imaging technologies for detecting the structural changes in the retinal nerve fiber layer have been developed. The assessment of the ganglion cell structure is based on measuring the thickness of the retinal nerve fiber layer.

One such device, the Scanning Laser Polarimeter (GDx-VCC, Laser Diagnostic Technologies, Inc., San Diego, CA) is based on the principle of measuring the change in the polarization of light exiting the eye. The retinal nerve fiber layer consists of parallel structures of diameter smaller than the wavelength of light, which makes it a birefringent structure. A birefringent structure has the ability to change the polarization of polarized light

double passing it.[7] The amount of change in the state of polarization (retardation) can be The device provides a large array of points corresponding to retinal.

This is found to be proportional to the thickness of the nerve fiber layer. This retardation (degrees) information is converted to thickness (microns) through the conversion factor based on the histologic comparison with monkey eyes. A new version of the device (GDx-VCC) is designed to individually compensate for the effects of birefringent properties of other parts of the eye like the cornea. The device provides a large array of points corresponding to retinal never fiber layer thickness at each respective point across the back of the eye. The scans thus available are in the form of 128 X 256 images or gray-level thickness maps. The goal of this thesis is to analyze these scans and develop classification techniques for them.

Figure 1.1 Photograph of a Scanning Laser Polarimetry Device (Courtesy: Laser Diagnostics, San Diego, CA)

The first step in a classification problem is to extract useful features from the setof given data. The original scan is used to derive patterns or features that can separate the two classes. The feature extraction step is to obtain a feature vector, a set of different useful features, which reduce the dimension of the original data while keeping all the essential information contained in the data. The next step after obtaining distinguishable and reliable set of features is to make them statistically independent. A very effectiveway to achieve this end is to perform Principal component analysis on the feature set. This will help in reducing feature dimension by eliminating redundancy caused by interdependencies in the feature vector. The last step is the classification of the data set into the two classes. Fisher's Linear Discriminant Analysis (LDF) provides an easy and robust way.

CHAPTER-2
LITERATURE SURVEY

Computer vision and image processing techniques play an important role in all fields of medical science and are especially relevant to modern Ophthalmology. Medical imaging has revolutionized the field of medicine by providing cost-effective healthcare and efficient diagnosis in all major disease areas. Medical imaging allows scientists and physicians to understand potential life-saving information using less invasive techniques. Applications that can interpret an image are being developed, which in turn can aid a physician in detecting possible subtle abnormalities. The computer indicates places in the image that require extra attention from the physician because they could be abnormal. These technologies known as Computer Aided Diagnosis (CAD) systems show that CAD can be helpful to improve diagnostic accuracy of physicians and lighten the burden of increasing workload.

The influence and impact of digital images on modern society, science, technology and art are tremendous. Image processing has become such a critical component in contemporary science and technology that many tasks would not be attempted without it. Digital image processing is an interdisciplinary subject that draws from synergistic developments involving many disciplines and is used in medical imaging, microscopy, astronomy, computer vision, geology and many other fields The rapid and continuing progress in computerized medical imaging, the associated developments in methods of analysis and computer-aided diagnosis, have propelled medical imaging into one of the most important sub-fields in scientific imaging. Medical image analysis is an area of research that attracts intensive interests of scientists and physicians and covers image processing, pattern recognition and computer visualization. Medical image processing involves the study of digital images with the objective of providing computational tools which will assist the quantification and visualization of interesting pathology and anatomical structures. The progress achieved in this field in recent years has significantly improved the type of medical care that is available to the patients.

The application of digital imaging to ophthalmology has now provided the possibility of processing retinal images to assist clinical diagnosis and treatment. Automated diagnosis of retinal fundus images using digital image analysis offers huge potential benefits. Due to advances in computer technology, medical diagnosis can be benefited from

computers which will assist doctors to analyze medical data and images with improved accuracy. Designing and developing computer-aided diagnostic tools or systems for medical images is a fast growing area in recent years.Development of an automated system for analyzing the images of the retina will facilitate computer aided diagnosis of eye diseases. The interest towards automatic detection of glaucoma and diabetic retinopathy has been increasing along with the rapid development of digital imaging and computing power. However, the most important single event that attracted the wider attention of medical research community has been the decision to recognize digital imaging as an accepted modality to document eye fundus. This introductory chapter presents some background information on the anatomy of the eye, ocular diseases like glaucoma need for screening.

Glaucoma is a chronic disease which if not detected in early stages can lead to permanent blindness. The medical techniques used by ophthalmologists like HRT and OCT is costly and time consuming. Hence there is a need to develop automatic computer aided system which can detect glaucoma efficiently and in less time. Optic disk and optic cup are prime features which help in diagnosing glaucoma. Thus proper segmentation of optic disk and optic cup plays an important role in detecting the disorder.

In this paper an adaptive threshold-based method which is independent of image quality and invariant to noise is used to segment optic disk, optic cup, Neuroretinal rim and cup to disk ratio is calculated to screen glaucoma. Another ocular parameter, rim to disk ratio is also considered which in combination with CDR gives more reliability in determining glaucoma and makes the system more robust. Further an SVM classifier has been used to categorize the images as glaucomatic or non glaucomatic. The experimental results obtained are compared with those of ophthalmologist and are found to have high accuracy of 90%. Also in addition, the proposed method is faster having low computational cost.

R. Manjula Sri, Ch. Madhubabu, and KMM Rao: Lab VIEW based assessment of CDR for the detection of Glaucoma

In this paper, Image processing and analysis has great significance in the field of medicine, especially in non-invasive treatment and clinical study. Glaucoma is a group of diseases, which damage eye's optic nerve that leads to blindness. These are presented to aid benchmarking of new methods.

Jayanthi Sivaswamy, S.R.Krishnadas, Arunava Chakravarty, Gopal Datt Joshi, and Ujjwalt A Comprehensive Retinal Image Dataset for the Assessment of Glaucoma from the Optic Nerve Head Analysis

In this paper, Optic nerve head (ONH) segmentation problem is of interest for automated glaucoma assessment. Although various segmentation methods have been proposed in the recent past, it is difficult to evaluate and compare the p individual methods due to a lack of a benchmark dataset. The assessment involves segmentation of optic disk and cup region within the ONH. In this paper, we present a comprehensive dataset of retinal images of both normal and glaucomatous eyes with manual segmentations from multiple human experts. The dataset also provides expert opinion on an image representing a normal or glaucomatous eye and on the presence of notching in an image.

Several state of the art methods are assessed against this dataset using cup to disc diameter ratio (CDR), area and boundary-based evaluation measures. These are presented to aid benchmarking of new methods. A supervised, notch detection method based on the segmentation results is also proposed and its assessment results are included for benchmarking.

Eleesa Jacob, R.Venkatesh: A Method of Segmentation for Glaucoma Screening Using Superpixel Classification

In this paper, an optic disc and optic cup segmentation is used to identify the glaucoma disease in time. In optic disc and optic cup segmentation, super pixel classification for glaucoma screening is proposed. In optic disc segmentation, histograms and centre surround statistics are used to classify each super pixel as disc or non-disc. A self-assessment reliability score is computed to evaluate the quality of the automated optic disc segmentation. In optic cup segmentation, the location information is also included into the feature space for better performance in addition to the histograms and centre surround statistics. The segmented optic cup and optic disc is then used to compute the cup to disc ratio for glaucoma screening. From the cup to disc ratio. analysis is performed to identify whether the given image is glaucomatous or not. The segmentation can be analyzed using the MATLAB.

Sheeba O, Jithin George, Rajin P. K., Nisha Thomas, and Sherin George: Glaucoma Detection Using Artificial Neural Network

In this paper, the nerve that transmits visual images to the brain. Here the detection of glaucoma is done by image processing. The screening of patients for the development of the

glaucoma potentially reduces the risk of blindness in these patients by 50%.

Here neural network is trained to recognize the parameters for the detection of different stages of the disease. The neuron model has been developed using feed forward back propagation network. Here the program is developed using Matlab. The images acquired using medical imaging techniques are analysed in Matlab. Matlab provide variety of options for image processing that enable us to extract the required features and information from the images. The software can be used to detect the early stages of glaucoma

Li Xiong, Huiqi Li; Yan Zheng: Automatic detection of glaucoma in retinal images

In this paper, which is based on principle components analysis (PCA) and Bayes classifier. Firstly, optic disc center is located using the combination of thresholding and distance transformation. Eigenvector spaces of normal set and glaucoma set are obtained respectively using PCA. A test image is projected onto these two spaces and the distance between projection and each template is calculated. Finally. decision is made according to Bayes classifier. The success rate of optic disk localization is 95.3% and 89.9% for normal set and glaucoma set respectively. The glaucoma detection algorithm was tested by over three hundred retinal images and the success rate is 78%.

Dey, N, Roy, A.B.; Das, A.; Chaudhuri, S.S: Optical cup to disc ratio measurement for glaucoma diagnosis using Harris corner

In this paper, Glaucoma is physiologically described as the deterioration of optic nerve cells, and is characterized by alterations in the optic nerve head and visual field. The measurement of neuro-retinal optic cup-to-disc ratio (CDR) is an important index of Glaucoma, as the increased excavation of the optic cup occurs because of Glaucomatous neuropathy increasing the CDR. Currently, CDR evaluation is manually performed by ophthalmologists.

CHAPTER-3
SOFTWARE INTRODUCTION

3.1. Introduction to MATLAB:

MATLAB is a high-performance language for technical computing. It integrates computation, visualization, and programming in an easy-to-use environment where problems and solutions are expressed in familiar mathematical notation. Typical uses include

- Math and computation
- Algorithm development
- Data acquisition
- Modeling, simulation, and prototyping
- Data analysis, exploration, and visualization
- Scientific and engineering graphics
- Application development, including graphical user interface building

MATLAB is an interactive system whose basic data element is an array that does not require dimensioning. This allows you to solve many technical computing problems, especially those with matrix and vector formulations, in a fraction of the time it would take to write a program in a scalar non interactive language such as C or FORTRAN.

The name MATLAB stands for matrix laboratory. MATLAB was originally written to provide easy access to matrix software developed by the LINPACK and EISPACK projects. Today, MATLAB engines incorporate the LAPACK and BLAS libraries, embedding the state of the art in software for matrix computation.

MATLAB has evolved over a period of years with input from many users. In university environments, it is the standard instructional tool for introductory and advanced courses in mathematics, engineering, and science. In industry, MATLAB is the tool of choice for high-productivity research, development, and analysis.

MATLAB features a family of add-on application-specific solutions called toolboxes. Very important to most uses of MATLAB, toolboxes allow you to learn and apply specialized technology. Toolboxes are comprehensive collections of MATLAB functions (M – files) that extend the MATLAB environment to solve particular classes of problems. Areas

in which toolboxes are available include signal processing, control systems, neural networks, fuzzy logic, wavelets, simulation, and many others.

3.2 The MATLAB system:

The MATLAB system consists of five main parts

- **Development Environment**:

This is the set of tools and facilities that help you use MATLAB functions and files. Many of these tools are graphical user interfaces. It includes the MATLAB desktop and command window, a command history, an editor and debugger, and browsers for viewing help, the workspace, files, and the search path.

- **The MATLAB Mathematical Function Library**:

This is a vast collection of computational algorithms ranging from elementary functions, like sum, sine, cosine, and complex arithmetic, to more sophisticated functions like matrix inverse, matrix Eigen values, Bessel functions, and fast Fourier transforms.

- **The MATLAB Language**:

This is a high-level matrix/array language with control flow statements, functions, data structures, input/output, and object-oriented programming features. It allows both "programming in the small" to rapidly create quick and dirty throw-away programs, and "programming in the large" to create large and complex application programs.

- **Graphics:**

MATLAB has extensive facilities for displaying vectors and matrices as graphs, as well as annotating and printing these graphs. It includes high-level functions for two-dimensional and three-dimensional data visualization, image processing, animation, and presentation graphics. It also includes low-level functions that allow you to fully customize the appearance of graphics as well as to build complete graphical user interfaces on your MATLAB applications. Areas in which toolboxes are available include signal processing, control systems. MATLAB features a family of add-on application-specific solutions called toolboxes.

- **The MATLAB Application Program Interface (API):**

MATLAB (Matrix Laboratory) is a high-level programming language and environment specifically designed for numerical computation, data analysis, and visualization. This is a library that allows you to write C and FORTRAN programs that interact with MATLAB. It includes facilities for calling routines from MATLAB (dynamic linking), calling MATLAB

as a computational engine, and for reading and writing MAT-files. Various toolboxes are there in MATLAB for computing recognition techniques, but we are using IMAGE PROCESSING toolbox.

3.3 GRAPHICAL USER INTERFACE (GUI):

MATLAB's Graphical User Interface Development Environment (GUIDE) provides a rich set of tools for incorporating graphical user interfaces (GUIs) in M-functions. Using GUIDE, the processes of laying out a GUI (i.e., its buttons, pop-up menus, etc.)and programming the operation of the GUI are divided conveniently into two easily managed and relatively independent tasks. The resulting graphical M-function is composed of two identically named (ignoring extensions) files:

- A file with extension .fig, called a FIG-file that contains a complete graphical description of all the function's GUI objects or elements and their spatial arrangement. A FIG-file contains binary data that does not need to be parsed when he associated GUI-based M-function is executed.
- A file with extension .m, called a GUI M-file, which contains the code that controls the GUI operation. This file includes functions that are called when the GUI is launched and exited, and callback functions that are executed when a user interacts with GUI objects for example, when a button is pushed.

To launch GUIDE from the MATLAB command window, type
guide filename
Where filename is the name of an existing FIG-file on the current path. If filename is omitted,
GUIDE opens a new (i.e., blank) window.

A graphical user interface (GUI) is a graphical display in one or more windows containing controls, called components that enable a user to perform interactive tasks. The user of the GUI does not have to create a script or type commands at the command line to accomplish the tasks. Unlike coding programs to accomplish tasks, the user of a GUI need not understand the details of how the tasks are performed.

GUI components can include menus, toolbars, push buttons, radio buttons, list boxes, and sliders just to name a few. GUIs created using MATLAB tools can also perform any type of computation, read and write data files, communicate with other GUIs, and display data as tables or as plots. The user of the GUI does not have to create a script or type

commands at the command line to accomplish the tasks. Unlike coding programs to accomplish tasks, the user of a GUI need not understand the details of how the tasks are performed.

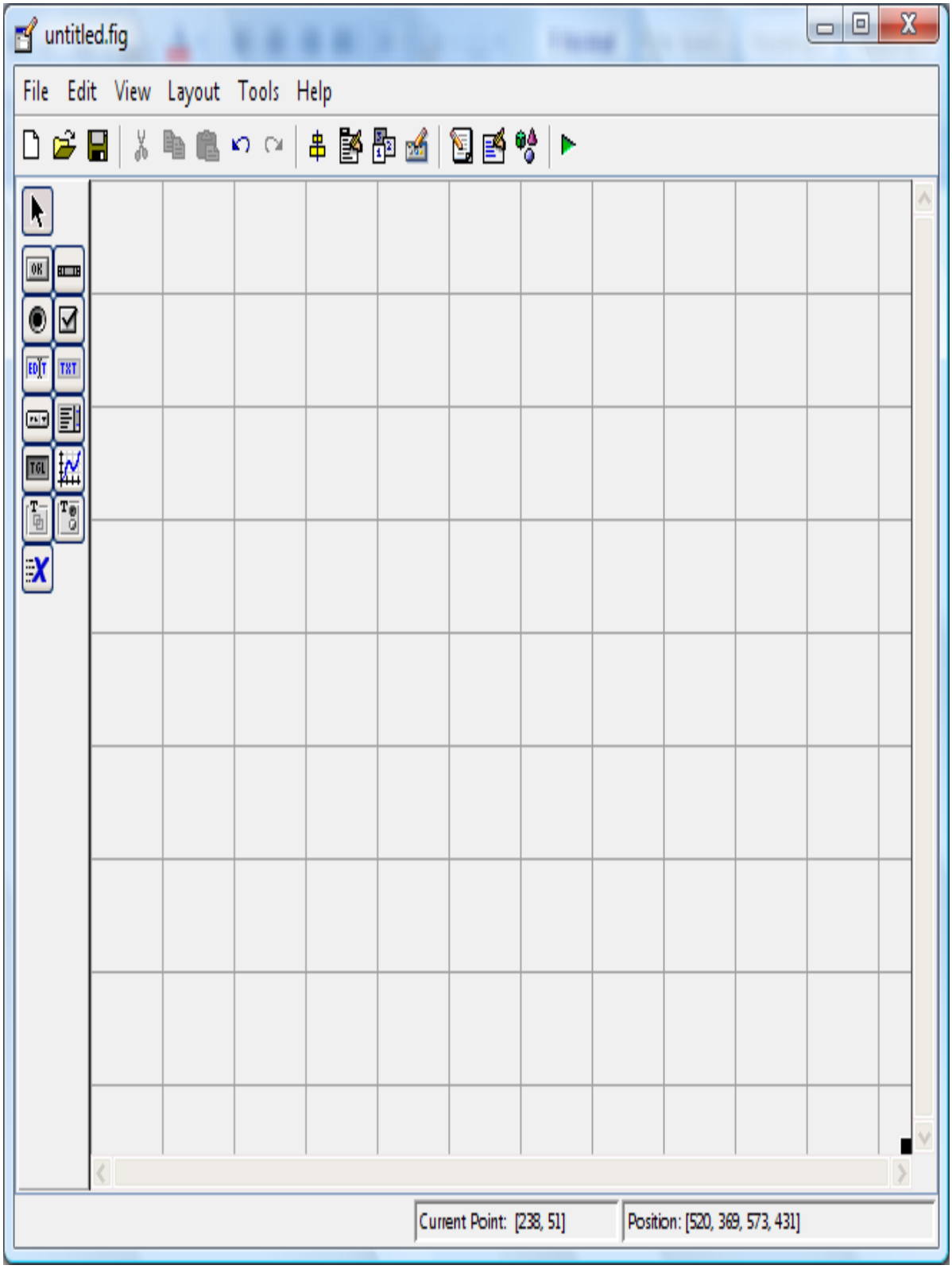

Fig 3.1 Graphical user interface

3.4 Software Description

3.4.1 Getting Started

If you are new to MATLAB, you should start by reading Manipulating Matrices. The most important things to learn are how to enter matrices, how to use the: (colon) operator, and how to invoke functions. After you master the basics, you should read the rest of the sections below and run the demos.

At the heart of MATLAB is a new language you must learn before you can fully exploit its power. You can learn the basics of MATLAB quickly, and mastery comes shortly after. You will be rewarded with high productivity, high-creativity computing power that will change the way you work.

3.4.2 Introduction - describes the components of the MATLAB system.

3.4.3 Development Environment - introduces the MATLAB development environment, including information about tools and the MATLAB desktop.

3.4.4 Manipulating Matrices - introduces how to use MATLAB to generate matrices and perform mathematical operations on matrices.

3.4.5 Graphics - introduces MATLAB graphic capabilities, including information about plotting data, annotating graphs, and working with images.

3.4.6 Programming with MATLAB - describes how to use the MATLAB language to create scripts and functions, and manipulate data structures, such as cell arrays and multidimensional arrays.

- **INTRODUCTION**

What Is MATLAB?

MATLAB is a high-performance language for technical computing. It integrates computation, visualization, and programming in an easy-to-use environment where problems and solutions are expressed in familiar mathematical notation. Typical uses include:

- Math and computation
- Algorithm development
- Modeling, simulation, and prototyping
- Data analysis, exploration, and visualization
- Scientific and engineering graphics
- Application development, including graphical user interface building

MATLAB is an interactive system whose basic data element is an array that does not Require dimensioning. MATLAB is an interactive system whose basic data element is an array that does not require dimensioning. This allows you to solve many technical computing problems, especially those with matrix and vector formulations, in a fraction of the time it would take to write a program in a scalar noninteractive language such as C or FORTRAN.

The name MATLAB stands for matrix laboratory. MATLAB was originally written to provide easy access to matrix software developed by the LINPACK and EISPACK projects. Today, MATLAB uses software developed by the LAPACK and ARPACK projects, which together represent the state-of-the-art in software for matrix computation.

MATLAB has evolved over a period of years with input from many users. In university environments, it is the standard instructional tool for introductory and advanced courses in mathematics, engineering, and science. In industry, MATLAB is the tool of choice for high-productivity research, development, and analysis.

MATLAB features a family of application-specific solutions called toolboxes. Very important to most users of MATLAB, toolboxes allow you to learn and apply specialized technology. Toolboxes are comprehensive collections of MATLAB functions (M-files) that extend the MATLAB environment to solve particular classes of problems. Areas in which toolboxes are available include signal processing, control systems, neural networks, fuzzy logic, wavelets, simulation, and many others.

- **The MATLAB System**

The MATLAB system consists of five main parts:

- **Development Environment**. This is the set of tools and facilities that help you use MATLAB functions and files. Many of these tools are graphical user interfaces. It includes the MATLAB desktop and Command Window, a command history, and browsers for viewing help, the workspace, files, and the search path.
- **The MATLAB Mathematical Function Library**. This is a vast collection of computational algorithms ranging from elementary functions like sum, sine, cosine, and complex arithmetic, to more sophisticated functions like matrix inverse, matrix eigenvalues, Bessel functions, and fast Fourier transforms.
- **The MATLAB Language**. This is a high-level matrix/array language with control flow statements, functions, data structures, input/output, and object-oriented programming features. It allows both "programming in the small" to rapidly create quick and dirty

throw-away programs, and "programming in the large" to create complete large and complex application programs.

- **Handle Graphics®**. This is the MATLAB graphics system. It includes high-level commands for two-dimensional and three-dimensional data visualization, image processing, animation, and presentation graphics. It also includes low-level commands that allow you to fully customize the appearance of graphics as well as to build complete graphical user interfaces on your MATLAB applications.
- **The MATLAB Application Program Interface (API).** This is a library that allows you to write C and FORTRAN programs that interact with MATLAB. It include facilities for calling routines from MATLAB (dynamic linking), calling MATLAB as a computational engine, and for reading and writing MAT-files.

3.5 DEVELOPMENT ENVIRONMENT

3.5.1 Introduction

This chapter provides a brief introduction to starting and quitting MATLAB, and the tools and functions that help you to work with MATLAB variables and files. For more information about the topics covered here, see the corresponding topics under Development Environment in the MATLAB documentation, which is available online as well as in print.

Starting and Quitting MATLAB

3.5.2 Starting MATLAB

On a Microsoft Windows platform, to start MATLAB, double-click the MATLAB shortcut icon on your Windows desktop. On a UNIX platform, to start MATLAB, type matlab at the operating system prompt. After starting MATLAB, the MATLAB desktop opens - see MATLAB Desktop.

You can change the directory in which MATLAB starts, define startup options including running a script upon startup, and reduce startup time in some situations.

3.5.3 Quitting MATLAB

To end your MATLAB session, select Exit MATLAB from the File menu in the desktop, or type quit in the Command Window. To execute specified functions each time MATLAB quits, such as saving the workspace, you can create and run a finish.m script.

3.5.4 MATLAB Desktop

When you start MATLAB, the MATLAB desktop appears, containing tools (graphical user interfaces) for managing files, variables, and applications associated with MATLAB. The first time MATLAB starts, the desktop appears as shown in the following illustration,

first time MATLAB starts, the desktop appears as shown in the following illustration, although your Launch Pad may contain different entries.

You can change the way your desktop looks by opening, closing, moving, and resizing the tools in it. You can also move tools outside of the desktop or return them back inside the desktop (docking). All the desktop tools provide common features such as context menus and keyboard shortcuts.

You can specify certain characteristics for the desktop tools by selecting Preferences from the File menu. For example, you can specify the font characteristics for Command Window text. For more information, click the Help button in the Preferences dialog box.

3.5.5 Desktop Tools

This section provides an introduction to MATLAB's desktop tools. You can also use MATLAB functions to perform most of the features found in the desktop tools. The tools are:

- Current Directory Browser
- Workspace Browser
- Array Editor
- Editor/Debugger
- Command Window
- Command History
- Launch Pad
- Help Browser

Command Window

Use the Command Window to enter variables and run functions and M-files.

Command History

Lines you enter in the Command Window are logged in the Command History window. In the Command History, you can view previously used functions, and copy and execute selected lines. To save the input and output from a MATLAB session to a file, use the diary function.

Running External Program

You can run external programs from the MATLAB Command Window. The exclamation point character! is a shell escape and indicates that the rest of the input line is a command to the operating system. This is useful for invoking utilities or running other programs without

quitting MATLAB. On Linux, for example, !emacs magic. m invokes an editor called emacs for a file named magik.m. When you quit the external program, the operating system returns control to MATLAB.

Launch Pad

MATLAB's Launch Pad provides easy access to tools, demos, and documentation.

Help Browser

Use the Help browser to search and view documentation for all your Math Works products. The Help browser is a Web browser integrated into the MATLAB desktop that displays HTML documents.

To open the Help browser, click the help button in the toolbar, or type help browser in the Command Window. The Help browser consists of two panes, the Help Navigator, which you use to find information, and the display pane, where you view the information.

Help Navigator

Use to Help Navigator to find information. It includes:

Product filter - Set the filter to show documentation only for the products you specify.

Contents tab - View the titles and tables of contents of documentation for your products.

Index tab - Find specific index entries (selected keywords) in the MathWorks documentation for your products.

Search tab - Look for a specific phrase in the documentation. To get help for a specific function, set the Search type to Function Name.

Favorites tab - View a list of documents you previously designated as favorites.

Display Pane

After finding documentation using the Help Navigator, view it in the display pane. While viewing the documentation, you can:

Browse to other pages - Use the arrows at the tops and bottoms of the pages, or use the back and forward buttons in the toolbar.

Bookmark pages - Click the Add to Favorites button in the toolbar.

Print pages - Click the print button in the toolbar.

Find a term in the page - Type a term in the Find in page field in the toolbar and click Go. Other features available in the display pane are: copying information, evaluating a selection, and viewing Web pages. If you just need to view the contents of an M-file, you can display it in the Command Window by using the type function.

Current Directory Browser

MATLAB file operations use the current directory and the search path as reference points. Any file you want to run must either be in the current directory or on the search path.

Search Path

To determine how to execute functions you call, MATLAB uses a search path to find M-files and other MATLAB-related files, which are organized in directories on your file system. Any file you want to run in MATLAB must reside in the current directory or in a directory that is on the search path. By default, the files supplied with MATLAB and MathWorks toolboxes are included in the search path. If you just need to view the contents of an M-file, you can display it in the Command Window by using the type function.

Workspace Browser

The MATLAB workspace consists of the set of variables (named arrays) built up during a MATLAB session and stored in memory. You add variables to the workspace by using functions, running M-files, and loading saved workspaces.

To view the workspace and information about each variable, use the Workspace browser, or use the functions who and who's. To delete variables from the workspace, select the variable and select Delete from the Edit menu. Alternatively, use the clear function.

The workspace is not maintained after you end the MATLAB session. To save the workspace to a file that can be read during a later MATLAB session, select Save Workspace As from the File menu, or use the save function. This saves the workspace to a binary file called a MAT-file, which has a .mat extension. There are options for saving to different formats. To read in a MAT-file, select Import Data from the File menu, or use the load function.

Array Editor

Double-click on a variable in the Workspace browser to see it in the Array Editor. Use the Array Editor to view and edit a visual representation of one- or two-dimensional numeric arrays, strings, and cell arrays of strings that are in the workspace.

Editor/Debugger

Use the Editor/Debugger to create and debug M-files, which are programs you write to runMATLAB functions. The Editor/Debugger provides a graphical user interface for basic

text editing, as well as for M-file debugging. You can use any text editor to create M-files, such as Emacs, and can use preferences (accessible from the desktop File menu) to specify that editor as the default. If you use another editor, you can still use the MATLAB Editor/Debugger for debugging, or you can use debugging functions, such as dbstop, which sets a breakpoint. If you just need to view the contents of an M-file, you can display it in the Command Window by using the type function.

3.6 MANIPULATING MATRICES

3.6.1 Entering Matrices

The best way for you to get started with MATLAB is to learn how to handle matrices. Start MATLAB and follow along with each example.

You can enter matrices into MATLAB in several different ways:

- Enter an explicit list of elements.
- Load matrices from external data files.
- Generate matrices using built-in functions.
- Create matrices with your own functions in M-files.

Start by entering Dürer's matrix as a list of its elements. You have only to follow a few basic conventions:

- Separate the elements of a row with blanks or commas.
- Use a semicolon, ; , to indicate the end of each row.
- Surround the entire list of elements with square brackets, [ ].

To enter Dürer's matrix, simply type in the Command Window

```
A = [16 3 2 13; 5 10 11 8; 9 6 7 12; 4 15 14 1]
```

MATLAB displays the matrix you just entered.

```
A =
   16    3    2   13
    5   10   11    8
    9    6    7   12
    4   15   14    1
```

This exactly matches the numbers in the engraving. Once you have entered the matrix, it is automatically remembered in the MATLAB workspace. You can refer to it simply as A.

3.6.2 Expressions

Like most other programming languages, MATLAB provides mathematical expressions, but unlike most programming languages, these expressions involve entire matrices. The building blocks of expressions are:

- Variables
- Numbers
- Operators
- Functions

Variables

MATLAB does not require any type declarations or dimension statements. When MATLAB encounters a new variable name, it automatically creates the variable and allocates the appropriate amount of storage. If the variable already exists, MATLAB changes its contents and, if necessary, allocates new storage. For example,

num_students = 25

Creates a 1-by-1 matrix named num_students and stores the value 25 in its single element. Variable names consist of a letter, followed by any number of letters, digits, or underscores. MATLAB uses only the first 31 characters of a variable name. MATLAB is case sensitive; it distinguishes between uppercase and lowercase letters. A and a are not the same variable. To view the matrix assigned to any variable, simply enter the variable name.

Numbers

MATLAB uses conventional decimal notation, with an optional decimal point and leading plus or minus sign, for numbers. Scientific notation uses the letter e to specify a power-of-ten scale factor. Imaginary numbers use either i or j as a suffix. Some examples of legal numbers are

3	-99	0.0001
9.6397238	1.60210e-20	6.02252e23
1i	-3.14159j	3e5i

All numbers are stored internally using the long format specified by the IEEE floating-point standard. Floating-point numbers have a finite precision of roughly 16 significant decimal digits and a finite range of roughly 10-308 to 10+308.

3.6.3 Operators

Expressions use familiar arithmetic operators and precedence rules.

+	Addition
-	Subtraction
*	Multiplication
/	Division
\	Left division (described in "Matrices and Linear Algebra" in Using MATLAB)
^	Power
'	Complex conjugate transpose
( )	Specify evaluation order

Table 3.1 Arithmetic operators and precedence rules

3.6.4 Functions

MATLAB provides a large number of standard elementary mathematical functions, including abs, sqrt, exp, and sin. Taking the square root or logarithm of a negative number is not an error; the appropriate complex result is produced automatically. MATLAB also provides many more advanced mathematical functions, including Bessel and gamma functions. Most of these functions accept complex arguments. For a list of the elementary mathematical functions, type help elfun, For a list of more advanced mathematical and matrix functions, type help specfun help elmat.

Some of the functions, like sqrt and sin, are built-in. They are part of the MATLAB core so they are very efficient, but the computational details are not readily accessible. Other functions, like gamma and sinh, are implemented in M-files. You can see the code and even modify it if you want. Several special functions provide values of useful constants.

In terms of programming and development, MATLAB allows users to write scripts to automate tasks and perform repetitive operations. Users can also create custom functions to perform specific tasks. MATLAB supports object-oriented programming, enabling users to create classes and objects. Additionally, MATLAB provides tools for creating graphical

user interfaces, including buttons, sliders, and menus. MATLAB offers various specialized toolboxes that provide functions for specific tasks. For example, the Image Processing Toolbox provides functions for image processing and analysis, while the Signal Processing Toolbox offers functions for signal processing and analysis.

Pi	3.14159265...
i	Imaginary unit, √-1
I	Same as i
Eps	Floating-point relative precision, 2^{-52}
Realmin	Smallest floating-point number, 2^{-1022}
Realmax	Largest floating-point number, $(2-\varepsilon)2^{1023}$
Inf	Infinity
NaN	Not-a-number

Table 3.2 Special functions

3.7 GUI

A graphical user interface (GUI) is a user interface built with graphical objects, such as buttons, text fields, sliders, and menus. In general, these objects already have meanings to most computer users. For example, when you move a slider, a value changes; when you press an OK button, your settings are applied and the dialog box is dismissed. Of course, to leverage this built-in familiarity, you must be consistent in how you use the various GUI-building components.

Applications that provide GUIs are generally easier to learn and use since the person using the application does not need to know what commands are available or how they work. The action that results from a particular user action can be made clear by the design of the interface. MATLAB provides various GUI components, including buttons, sliders, menus,

lists, tables, and axes. Each component has its own set of properties and methods that you can use to customize its behavior.

The sections that follow describe how to create GUIs with MATLAB. This includes laying out the components, programming them to do specific things in response to user actions, and saving and launching the GUI; in other words, the mechanics of creating GUIs. This documentation does not attempt to cover the "art" of good user interface design, which is an entire field unto itself. Topics covered in this section include:

3.7.1 Creating GUIs with GUIDE

MATLAB implements GUIs as figure windows containing various styles of uicontrol objects. You must program each object to perform the intended action when activated by the user of the GUI. In addition, you must be able to save and launch your GUI. All of these tasks are simplified by GUIDE, MATLAB's graphical user interface development environment. callback routines (see Creating and Storing the Object Handle Structure for more information). The M-files provides a way to manage global data (see Managing GUI Data for more information).

3.7.2 GUI Development Environment

The process of implementing a GUI involves two basic tasks:

- Laying out the GUI components
- Programming the GUI components

GUIDE primarily is a set of layout tools. However, GUIDE also generates an M-file that contains code to handle the initialization and launching of the GUI. This M-file provides a framework for the implementation of the callbacks - the functions that execute when users activate components in the GUI.

The Implementation of a GUI

While it is possible to write an M-file that contains all the commands to lay out a GUI, it is easier to use GUIDE to lay out the components interactively and to generate two files that save and launch the GUI:

A FIG-file - contains a complete description of the GUI figure and all of its children (uicontrols and axes), as well as the values of all object properties.

An M-file - contains the functions that launch and control the GUI and the callbacks, which are defined as subfunctions. This M-file is referred to as the

application M-file in this documentation.

Note that the application M-file does not contain the code that lays out the uicontrols; this information is saved in the FIG-file.

The following diagram illustrates the parts of a GUI implementation.

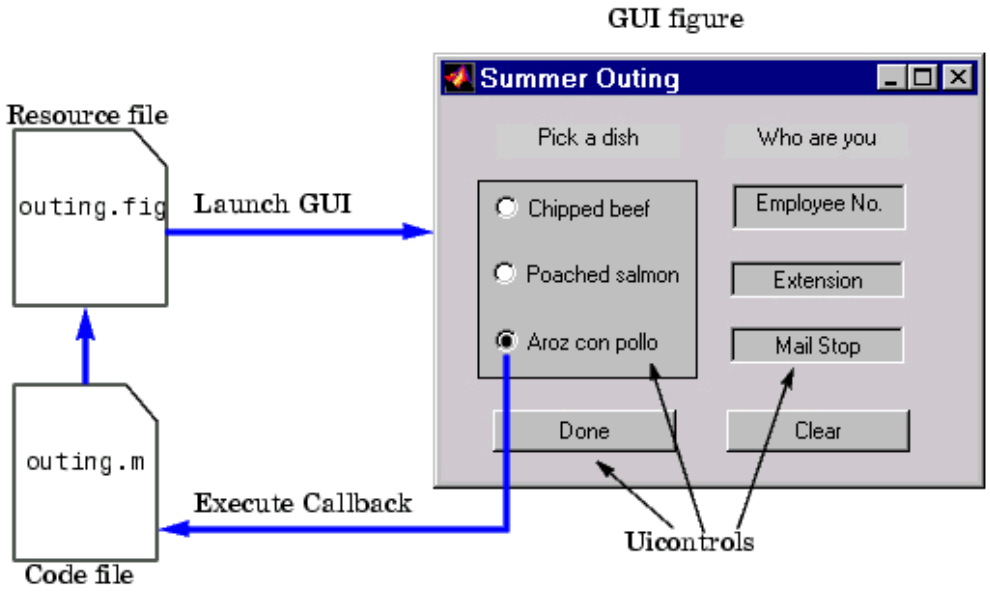

Fig 3.2 Graphical user blocks

3.7.3 Features of the GUIDE-Generated Application M-File

GUIDE simplifies the creation of GUI applications by automatically generating an M-file framework directly from your layout. You can then use this framework to code your application M-file. This approach provides a number of advantages:

The M-file contains code to implement a number of useful features (see Configuring Application Options for information on these features). The M-file adopts an effective approach to managing object handles and executing callback routines (see Creating and Storing the Object Handle Structure for more information). The M-files provides a way to manage global data (see Managing GUI Data for more information).

The automatically inserted subfunction prototypes for callbacks ensure compatibility with future releases. For more information, see Generating Callback Function Prototypes for information on syntax and arguments.

You can elect to have GUIDE generate only the FIG-file and write the application M-file yourself. Keep in mind that there are no uicontrol creation commands in the

application M-file; the layout information is contained in the FIG-file generated by the Layout Editor. The M-files provides a way to manage global data.

3.7.4 Beginning the Implementation Process

To begin implementing your GUI, proceed to the following sections:

Getting Started with GUIDE - the basics of using GUIDE.

Selecting GUIDE Application Options - set both FIG-file and M-file options.

Using the Layout Editor - begin laying out the GUI.

Understanding the Application M-File - discussion of programming techniques used in the application M-file.

Application Examples - a collection of examples that illustrate techniques which are useful for implementing GUIs.

Command-Line Accessibility

When MATLAB creates a graph, the figure and axes are included in the list of children of their respective parents and their handles are available through commands such as findobj, set, and get. If you issue another plotting command, the output is directed to the current figure and axes. GUIs are also created in figure windows. Generally, you do not want GUI figures to be available as targets for graphics output, since issuing a plotting command could direct the output to the GUI figure, resulting in the graph appearing in the middle of the GUI. In contrast, if you create a GUI that contains an axes and you want commands entered in the command window to display in this axes, you should enable command-line access.

3.7.5 User Interface Controls

The Layout Editor component palette contains the user interface controls that you can use in your GUI. These components are MATLAB uicontrol objects and are programmable via their Callback properties. This section provides information on these components.

- Push Buttons
- Sliders
- Toggle Buttons
- Frames
- Radio Buttons
- List boxes
- Checkboxes
- Popup Menus

- Edit Text
- Axes
- Static Text
- Figures

- **Push Buttons**

Push buttons generate an action when pressed (e.g., an OK button may close a dialog box and apply settings). When you click down on a push button, it appears depressed; when you release the mouse, the button's appearance returns to its nondepressed state; and its callback executes on the button up event.

Properties to Set

String - set this property to the character string you want displayed on the push button.

Tag - GUIDE uses the Tag property to name the callback subfunction in the application M-file. Set Tag to a descriptive name (e.g., close_button) before activating the GUI.

Programming the Callback

When the user clicks on the push button, its callback executes. Push buttons do not return a value or maintain a state.

- **Toggle Buttons**

Toggle buttons generate an action and indicate a binary state (e.g., on or off). When you click on a toggle button, it appears depressed and remains depressed when you release the mouse button, at which point the callback executes. A subsequent mouse click returns the toggle button to the nondepressed state and again executes its callback.

Programming the Callback

The callback routine needs to query the toggle button to determine what state it is in. MATLAB sets the Value property equal to the Max property when the toggle button is depressed (Max is 1 by default) and equal to the Min property when the toggle button is not depressed (Min is 0 by default).

From the GUIDE Application M-File

The following code illustrates how to program the callback in the GUIDE application M-file.

```
function varargout = togglebutton1_Callback(h,eventdata,handles,varargin)
button_state = get(h,'Value');
```

```
if button_state == get(h,'Max')
% toggle button is pressed
elseif button_state == get(h,'Min')
% toggle button is not pressed
End
```

Adding an Image to a Push Button or Toggle Button

Assign the CData property an m-by-n-by-3 array of RGB values that define a truecolor image. For example, the array a defines 16-by-128 truecolor image using random values between 0 and 1 (generated by rand).

```
a(:,:,1) = rand(16,128);
a(:,:,2) = rand(16,128);
a(:,:,3) = rand(16,128);
set(h,'CData',a)
```

- **Radio Buttons**

Radio buttons are similar to checkboxes, but are intended to be mutually exclusive within a group of related radio buttons (i.e., only one button is in a selected state at any given time). To activate a radio button, click the mouse button on the object. The display indicates the state of the button.

Implementing Mutually Exclusive Behavior

Radio buttons have two states - selected and not selected. You can query and set the state of a radio button through its Value property:

Value = Max, button is selected.

Value = Min, button is not selected.

To make radio buttons mutually exclusive within a group, the callback for each radio button must set the Value property to 0 on all other radio buttons in the group. MATLAB sets the Value property to 1 on the radio button clicked by the user.

The following subfunction, when added to the application M-file, can be called by each radio button callback. The argument is an array containing the handles of all other radio buttons in the group that must be deselected.

```
function mutual_exclude(off)
set(off,'Value',0)
```

Obtaining the Radio Button Handles.

The handles of the radio buttons are available from the handles structure, which contains the handles of all components in the GUI. This structure is an input argument to all radio button callbacks.

The following code shows the call to mutual exclude being made from the first radio button's callback in a group of four radio buttons.

```
function varargout = radiobutton1_Callback(h,eventdata,handles,varargin)
off = [handles.radiobutton2,handles.radiobutton3,handles.radiobutton4];
mutual_exclude(off)
% Continue with callback
   .
   .
```

After setting the radio buttons to the appropriate state, the callback can continue with its implementation-specific tasks.

- **Checkboxes**

Check boxes generate an action when clicked and indicate their state as checked or not checked. Check boxes are useful when providing the user with a number of independent of

choices that set a mode (e.g., display a toolbar or generate callback function prototypes).

The Value property indicates the state of the check box by taking on the value of the Max or Min property (1 and 0 respectively by default):

Value = Max, box is checked.

Value = Min, box is not checked.

You can determine the current state of a check box from within its callback by querying the state of its Value property, as illustrated in the following example:

```
function checkbox1_Callback(h,eventdata,handles,varargin)
if (get(h,'Value') == get(h,'Max'))
% then checkbox is checked-take approriate action
else
% checkbox is not checked-take approriate action
End
```

When a check box is clicked, it generates an action and indicates its state as checked or not checked. Check boxes are useful when providing users with independent choices.

- **Edit Text**

Edit text controls are fields that enable users to enter or modify text strings. Use edit text when you want text as input. The String property contains the text entered by the user.
To obtain the string typed by the user, get the String property in the callback.

```
function edittext1Callback(h,eventdata, handles,varargin)
user_string = get(h,'string');
% proceed with callback...
```

Obtaining Numeric Data from an Edit Test Component

MATLAB returns the value of the edit text String property as a character string. If you want users to enter numeric values, you must convert the characters to numbers. You can do this using the str2double command, which converts strings to doubles. If the user enters non-numeric characters, str2double returns NaN.

You can use the following code in the edit text callback. It gets the value of the String property and converts it to a double. It then checks if the converted value is NaN, indicating the user entered a non-numeric character (isnan) and displays an error dialog (errordlg).

```
function edittext1_Callback(h,eventdata,handles,varargin)
user_entry = str2double(get(h,'string'));
if isnan(user_entry)
errordlg('You must enter a numeric value','Bad Input','modal')
end
% proceed with callback...
```

Triggering Callback Execution

On UNIX systems, clicking on the menubar of the figure window causes the edit text callback to execute. However, on Microsoft Windows systems, if an editable text box has focus, clicking on the menubar does not cause the editable text callback routine to execute. This behavior is consistent with the respective platform conventions. Clicking on other components in the GUI execute the callback.

- **Static Text**

Static text controls displays lines of text. Static text is typically used to label other controls, provide directions to the user, or indicate values associated with a slider. Users cannot change static text interactively and there is no way to invoke the callback routine associated with it.

- **Frames**

Frames are boxes that enclose regions of a figure window. Frames can make a user interface easier to understand by visually grouping related controls. Frames have no callback routines associated with them and only uicontrols can appear within frames (axes cannot).

Placing Components on Top of Frames

Frames are opaque. If you add a frame after adding components that you want to be positioned within the frame, you need to bring forward those components. Use the Bring to Front and Send to Back operations in the Layout menu for this purpose.

- **List Boxes**

List boxes display a list of items and enable users to select one or more items. The String property contains the list of strings displayed in the list box. The first item in the list has an index of 1.

The Value property contains the index into the list of strings that correspond to the selected item. If the user selects multiple items, then Value is a vector of indices. By default, the first item in the list is highlighted when the list box is first displayed. If you do not want any item highlighted, then set the Value property to empty, []. The ListboxTop property defines which string in the list displays as the top most item when the list box is not large enough to display all list entries. ListboxTop is an index into the array of strings defined by the String property and must have a value between 1 and the number of strings. Noninteger values are fixed to the next lowest integer.

Single or Multiple Selection

The values of the Min and Max properties determine whether users can make single or multiple selections:

If Max - Min > 1, then list boxes allow multiple item selection.

If Max - Min <= 1, then list boxes do not allow multiple item selection.

Selection Type

Listboxes differentiate between single and double clicks on an item and set the figure SelectionType property to normal or open accordingly. See Triggering Callback Execution for information on how to program multiple selection. When using check boxes in GUI design, there are several best practices to keep in mind. First, use check boxes for independent options that don't affect each other.

Triggering Callback Execution

MATLAB evaluates the list box's callback after the mouse button is released or a keypress event (including arrow keys) that changes the Value property (i.e., any time the user clicks on an item, but not when clicking on the list box scrollbar). This means the callback is executed after the first click of a double-click on a single item or when the user is making multiple selections.

In these situations, you need to add another component, such as a Done button (push button) and program its callback routine to query the list box Value property (and possibly the figure SelectionType property) instead of creating a callback for the list box. If you are using the automatically generated application M-file option, you need to either:

Set the list box Callback property to the empty string (') and remove the callback subfunction from the application M-file. Leave the callback subfunction stub in the application M-file so that no code executes when users click on list box items.

The first choice is best if you are sure you will not use the list box callback and you want to minimize the size and efficiency of the application M-file. However, if you think you may want to define a callback for the list box at some time, it is simpler to leave the callback stub in the M-file.

Popup Menus

Popup menus open to display a list of choices when users press the arrow. The string property contains the list of string displayed in the popup menu. The Value property contains the index into the list of strings that correspond to the selected item. When not open, a popup menu displays the current choice, which is determined by the index contained in the Value property. The first item in the list has an index of 1. Popup menus are useful when you want to provide users with a number of mutually exclusive choices, but do not want to take up the amount of space that a series of radio buttons requires.

Programming the Popup Menu

You can program the popup menu callback to work by checking only the index of the item selected (contained in the Value property) or you can obtain the actual string contained in the selected item. This callback checks the index of the selected item and uses a switch statement to take action based on the value. If the contents of the popup menu is fixed, then you can use this approach.

```
function varargout = popupmenu1_Callback(h,eventdata,handles,varargin)
val = get(h,'Value');
switch val
case 1
% The user selected the first item
case 2
% The user selected the second item
% etc.
```

This callback obtains the actual string selected in the popup menu. It uses the value to index into the list of strings. This approach may be useful if your program dynamically loads the contents of the popup menu based on user action and you need to obtain the selected string. Note that it is necessary to convert the value returned by the String property from a cell array to a string.

```
function varargout = popupmenu1_Callback(h,eventdata,handles,varargin)
val = get(h,'Value');
string_list = get(h,'String');
selected_string = string_list{val}; % convert from cell array to string
% etc.
```

Enabling or Disabling Controls

You can control whether a control responds to mouse button clicks by setting the Enable property. Controls have three states:

on - The control is operational

off - The control is disabled and its label (set by the string property) is grayed out.

inactive - The control is disabled, but its label is not grayed out.

When a control is disabled, clicking on it with the left mouse button does not execute its callback routine. However, the left-click causes two other callback routines to execute: First the figure WindowButtonDownFcn callback executes. Then the control's ButtonDownFcn callback executes. A right mouse button click on a disabled control posts a context menu, if one is defined for that control.

See the Enable property description for more details. Disabling a GUI component means making it inactive and unavailable for user interaction. When a component is

disabled, it cannot be clicked, edited, or selected. Enabling and disabling are two important concepts in MATLAB GUI that allow you to control the behavior of GUI components.

- **Axes**

Axes enable your GUI to display graphics (e.g., graphs and images). Like all graphics objects, axes have properties that you can set to control many aspects of its behavior and appearance. See Axes Properties for general information on axes objects.

Axes Callbacks

Axes are not uicontrol objects, but can be programmed to execute a callback when users click a mouse button in the axes. Use the axes ButtonDownFcn property to define the callback.

Plotting to Axes in GUIs

GUIs that contain axes should ensure the Command-line accessibility option in the Application Options dialog is set to Callback (the default). This enables you to issue plotting commands from callbacks without explicitly specifying the target axes.

GUIs with Multiple Axes

If a GUI has multiple axes, you should explicitly specify which axes you want to target when you issue plotting commands. You can do this using the axes command and the handles structure. For example,

axes(handles.axes1)

makes the axes whose Tag property is axes1 the current axes, and therefore the target for plotting commands. You can switch the current axes whenever you want to target a different axes. See GUI with Multiple Axes for and example that uses two axes. This enables you to issue plotting commands from callbacks without explicitly specifying the target axes. A GUI can contain various components, such as buttons, sliders, and axes. This enables you to issue plotting commands from callbacks without explicitly specifying the target axes.

- **Figure**

Figures are the windows that contain the GUI you design with the Layout Editor. See the description of figure properties for information on what figure characteristics you can control.

3.8 INTRODUCTION TO IMAGE PROCESSING

3.8.1 IMAGE:

An image is a two-dimensional picture, which has a similar appearance to some subject usually a physical object or a person. Image is a two-dimensional, such as a photograph, screen display, and as well as a three-dimensional, such as a statue. They may be captured by optical devices—such as cameras, mirrors, lenses, telescopes, microscopes, etc. and natural objects and phenomena, such as the human eye or water surfaces.

The word image is also used in the broader sense of any two-dimensional figure such as a map, a graph, a pie chart, or an abstract painting. In this wider sense, images can also be rendered manually, such as by drawing, painting, carving, rendered automatically by printing or computer graphics technology, or developed by a combination of methods, especially in a pseudo-photograph.

Fig 3.3 General image

An image is a rectangular grid of pixels. It has a definite height and a definite width counted in pixels. Each pixel is square and has a fixed size on a given display. However different computer monitors may use different sized pixels. The pixels that constitute an image are ordered as a grid (columns and rows); each pixel consists of numbers representing magnitudes of brightness and color.

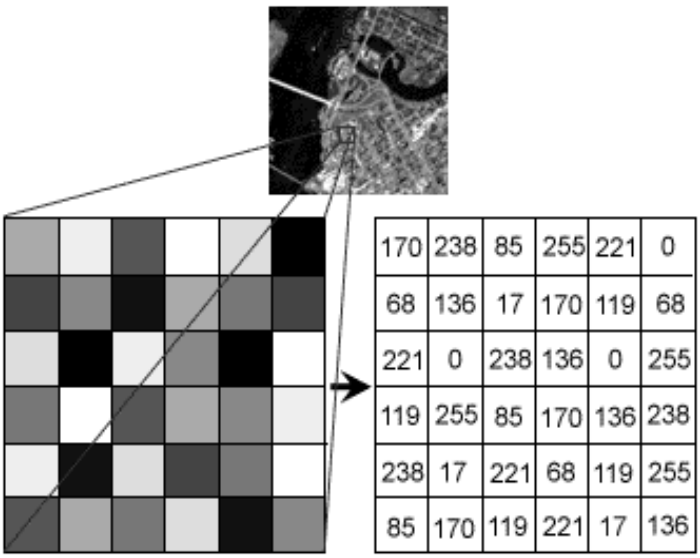

Fig 3.4 Image pixel

Each pixel has a color. The color is a 32-bit integer. The first eight bits determine the redness of the pixel, the next eight bits the greenness, the next eight bits the blueness, and the remaining eight bits the transparency of the pixel.

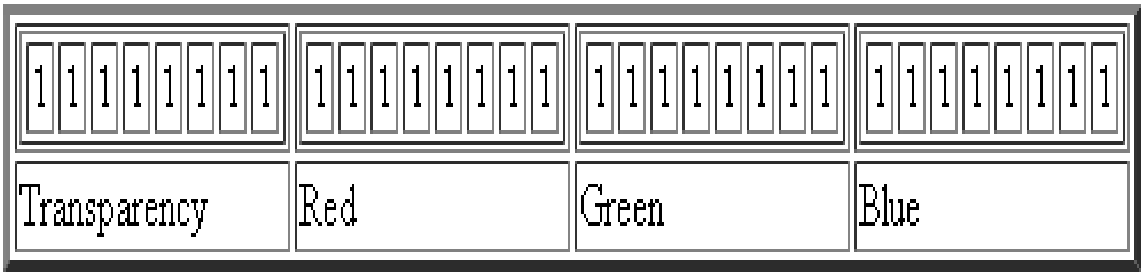

Fig 3.5 Transparency image

3.8.2 IMAGE FILE SIZES:

Image file size is expressed as the number of bytes that increases with the number of pixels composing an image, and the color depth of the pixels. The greater the number of rows and columns, the greater the image resolution, and the larger the file. Also, each pixel of an image increases in size when its color depth increases, an 8-bit pixel (1 byte) stores 256 colors, a 24-bit pixel (3 bytes) stores 16 million colors, the latter known as true color.

Image compression uses algorithms to decrease the size of a file. High resolution cameras produce large image files, ranging from hundreds of kilobytes to megabytes, per the camera's resolution and the image-storage format capacity. High resolution digital cameras record 12 megapixel (1MP = 1,000,000 pixels / 1 million) images, or more, in true

color. For example, an image recorded by a 12 MP camera; since each pixel uses 3 bytes to record true color, the uncompressed image would occupy 36,000,000 bytes of memory, a great amount of digital storage for one image, given that cameras must record and store many images to be practical. Faced with large file sizes, both within the camera and a storage disc, image file formats were developed to store such large images.

3.8.3 IMAGE FILE FORMATS:

Image file formats are standardized means of organizing and storing images. This entry is about digital image formats used to store photographic and other images. Image files are composed of either pixel or vector (geometric) data that are rasterized to pixels when displayed (with few exceptions) in a vector graphic display. Including proprietary types, there are hundreds of image file types. The PNG, JPEG, and GIF formats are most often used to display images on the Internet.

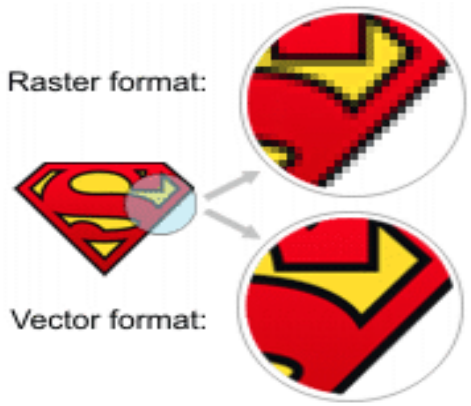

Fig 3.6 Resolution image

In addition to straight image formats, Metafile formats are portable formats which can include both raster and vector information. The metafile format is an intermediate format. Most Windows applications open metafiles and then save them in their own native format.

RASTER FORMATS:

These formats store images as bitmaps (also known as pixmaps).

- **JPEG/JFIF:**

JPEG (Joint Photographic Experts Group) is a compression method. JPEG compressed images are usually stored in the JFIF (JPEG File Interchange Format) file format. JPEG compression is lossy compression. Nearly every digital camera can save images in the JPEG/JFIF format, which supports 8 bits per color (red, green, blue) for a 24-bit total, producing relatively small files. Photographic images may be better stored in a lossless non-

JPEG format if they will be re-edited, or if small "artifacts" are unacceptable. The JPEG/JFIF format also is used as the image compression algorithm in many Adobe PDF files.

- **EXIF:**

The EXIF (Exchangeable image file format) format is a file standard similar to the JFIF format with TIFF extensions. It is incorporated in the JPEG writing software used in most cameras. Its purpose is to record and to standardize the exchange of images with image metadata between digital cameras and editing and viewing software. The metadata are recorded for individual images and include such things as camera settings, time and date, shutter speed, exposure, image size, compression, name of camera, color information, etc. When images are viewed or edited by image editing software, all of this image information can be displayed.

- **TIFF:**

The TIFF (Tagged Image File Format) format is a flexible format that normally saves 8 bits or 16 bits per color (red, green, blue) for 24-bit and 48-bit totals, respectively, usually using either the TIFF or TIF filename extension. TIFFs are lossy and lossless. Some offer relatively good lossless compression for bi-level (black & white) images. Some digital cameras can save in TIFF format, using the LZW compression algorithm for lossless storage. TIFF image format is not widely supported by web browsers. TIFF remains widely accepted as a photograph file standard in the printing business. TIFF can handle device-specific color spaces, such as the CMYK defined by a particular set of printing press inks.

- **PNG:**

The PNG (Portable Network Graphics) file format was created as the free, open-source successor to the GIF. The PNG file format supports true color (16 million colors) while the GIF supports only 256 colors. The PNG file excels when the image has large, uniformly colored areas. The lossless PNG format is best suited for editing pictures, and the lossy formats, like JPG, are best for the final distribution of photographic images, because JPG files are smaller than PNG files.

PNG, an extensible file format for the lossless, portable, well-compressed storage of raster images. PNG provides a patent-free replacement for GIF and can also replace many common uses of TIFF. Indexed-color, grayscale, and true color images are supported, plus an optional alpha channel. PNG is designed to work well in online viewing applications, such as the World Wide Web. PNG is robust, providing both full file integrity checking and simple detection of common transmission errors.

- **GIF:**

GIF (Graphics Interchange Format) is limited to an 8-bit palette, or 256 colors. This makes the GIF format suitable for storing graphics with relatively few colors such as simple diagrams, shapes, logos and cartoon style images. The GIF format supports animation and is still widely used to provide image animation effects. It also uses a lossless compression that is more effective when large areas have a single color, and ineffective for detailed images or dithered images.

- **BMP:**

The BMP file format (Windows bitmap) handles graphics files within the Microsoft Windows OS. Typically, BMP files are uncompressed, hence they are large. The advantage is their simplicity and wide acceptance in Windows programs.

VECTOR FORMATS:

As opposed to the raster image formats above (where the data describes the characteristics of each individual pixel), vector image formats contain a geometric description which can be rendered smoothly at any desired display size.

At some point, all vector graphics must be rasterized in order to be displayed on digital monitors. However, vector images can be displayed with analog CRT technology such as that used in some electronic test equipment, medical monitors, radar displays, laser shows and early video games. Plotters are printers that use vector data rather than pixel data to draw graphics.

- **CGM:**

CGM (Computer Graphics Metafile) is a file format for 2D vector graphics, raster graphics, and text. All graphical elements can be specified in a textual source file that can be compiled into a binary file or one of two text representations. CGM provides a means of graphics data interchange for computer representation of 2D graphical information independent from any particular application, system, platform, or device.

- **SVG:**

SVG (Scalable Vector Graphics) is an open standard created and developed by the World Wide Web Consortium to address the need for a versatile, scriptable and all purpose vector format for the web and otherwise. The SVG format does not have a compression scheme of its own, but due to the textual nature of XML, an SVG graphic can be compressed using a program such as gzip.

3.8.4 IMAGE PROCESSING:

Digital image processing, the manipulation of images by computer, is relatively recent development in terms of man's ancient fascination with visual stimuli. In its short history, it has been applied to practically every type of images with varying degree of success. The inherent subjective appeal of pictorial displays attracts perhaps a disproportionate amount of attention from the scientists and also from the layman.

Digital image processing like other glamour fields, suffers from myths, mis-connect ions, mis-understandings and mis-information. It is vast umbrella under which fall diverse aspect of optics, electronics, mathematics, photography graphics and computer technology. It is truly multidisciplinary endeavor ploughed with imprecise jargon.

Several factor combine to indicate a lively future for digital image processing. A major factor is the declining cost of computer equipment. Several new technological trends promise to further promote digital image processing. These include parallel processing mode practical by low cost microprocessors, and the use of charge coupled devices (CCDs) for digitizing, storage during processing and display and large low cost of image storage arrays.

FUNDAMENTAL STEPS IN DIGITAL IMAGE PROCESSING:

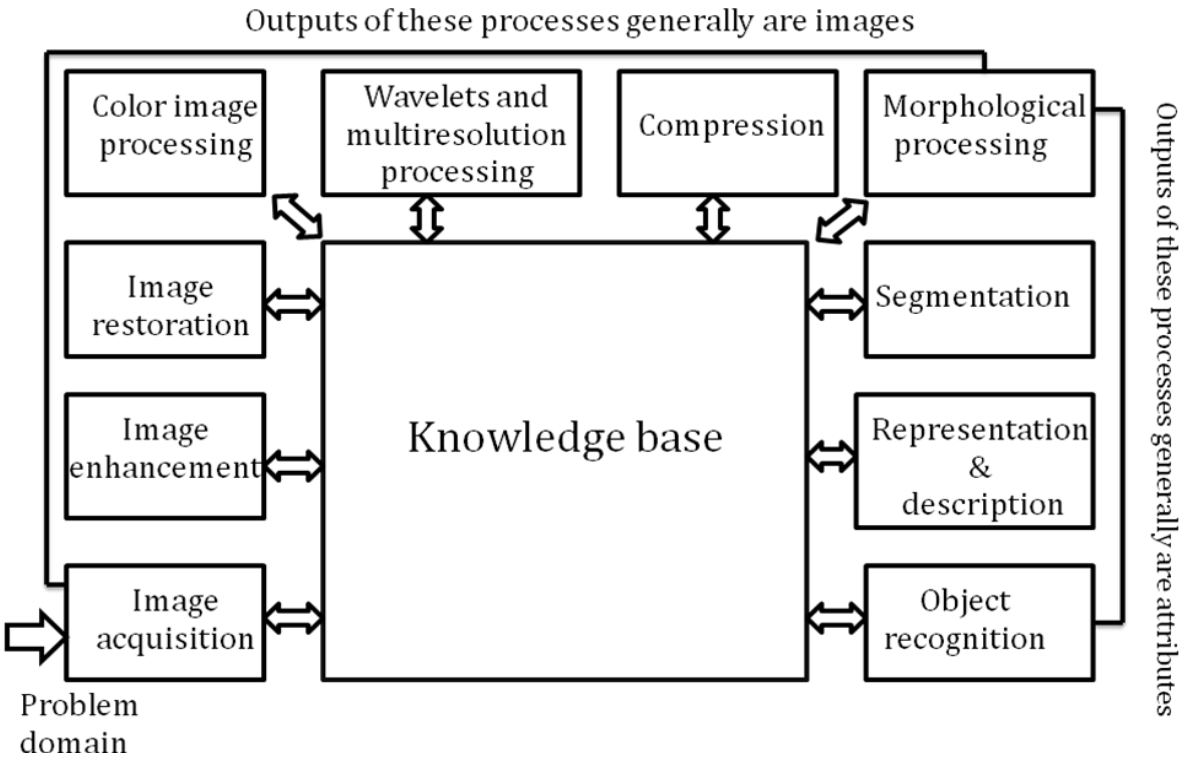

Fig 3.7 Fundamental steps in Digital Image Processing

- **Image Acquisition:**

Image Acquisition is to acquire a digital image. To do so requires an image sensor and the capability to digitize the signal produced by the sensor. The sensor could be monochrome or color TV camera that produces an entire image of the problem domain every 1/30 sec. the image sensor could also be line scan camera that produces a single image line at a time. In this case, the objects motion past the line.

Fig 3.7.1 Acquisation of Image

Scanner produces a two-dimensional image. If the output of the camera or other imaging sensor is not in digital form, an analog to digital converter digitizes it. The nature of the sensor and the image it produces are determined by the application.

- **Image Enhancement:**

Image enhancement is among the simplest and most appealing areas of digital image processing. Basically, the idea behind enhancement techniques is to bring out detail that is obscured, or simply to highlight certain features of interesting an image. A familiar example of enhancement is when we increase the contrast of an image because “it looks better.” It is important to keep in mind that enhancement is a very subjective area of image processing.

Fig 3.7.2 Enhancement of image

- **Image restoration:**

Image restoration is an area that also deals with improving the appearance of an image. However, unlike enhancement, which is subjective, image restoration is objective, in the sense that restoration techniques tend to be based on mathematical or probabilistic models of image degradation. Enhancement, on the other hand, is based on human subjective preferences regarding what constitutes a "good" enhancement result. For example, contrast stretching is considered an enhancement technique because it is based primarily on the pleasing aspects it might present to the viewer, where as removal of image blur by applying a deblurring function is considered a restoration technique.

Fig 3.7.3 Restoration of Image

- **Color image processing:**

The use of color in image processing is motivated by two principal factors. First, color is a powerful descriptor that often simplifies object identification and extraction from a scene. Second, humans can discern thousands of color shades and intensities, compared to about only two dozen shades of gray. This second factor is particularly important in manual image analysis.

Fig 3.7.4 color image

- **Wavelets and multiresolution processing:**

Wavelets are the formation for representing images in various degrees of resolution. Although the Fourier transform has been the mainstay of transform based image processing since the late1950's, a more recent transformation, called the wavelet transform, and is now making it even easier to compress, transmit, and analyze many images. Unlike the Fourier transform, whose basis functions are sinusoids, wavelet transforms are based on small values, called Wavelets, of varying frequency and limited duration.

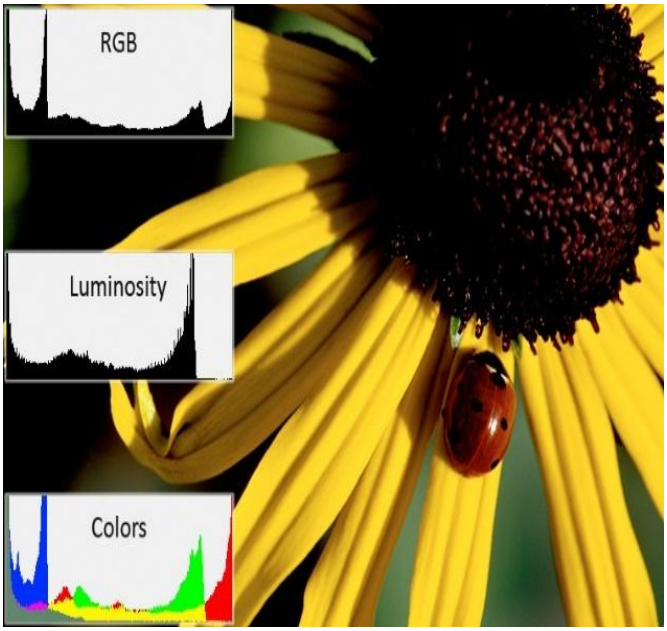

Fig 3.7.5 Wavelets

Wavelets were first shown to be the foundation of a powerful new approach to signal processing and analysis called Multiresolution theory. Multiresolution theory incorporates and unifies techniques from a variety of disciplines, including sub band coding from signal processing, quadrature mirror filtering from digital speech recognition, and pyramidal image processing.

- **Compression:**

Compression, as the name implies, deals with techniques for reducing the storage required saving an image, or the bandwidth required for transmitting it. Although storage technology has improved significantly over the past decade, the same cannot be said for transmission capacity. This is true particularly in uses of the Internet, which are characterized by significant pictorial content. Image compression is familiar to most users of computers in the form of image file extensions, such as the jpg file extension used in the JPEG (Joint Photographic Experts Group) image compression standard.

- **Morphological processing:**

Morphological processing deals with tools for extracting image components that are useful in the representation and description of shape. The language of mathematical morphology is set theory. As such, morphology offers a unified and powerful approach to numerous image processing problems. Sets in mathematical morphology represent objects in an image. For example, the set of all black pixels in a binary image is a complete morphological description of the image. In binary images, the sets in question are members of the 2-D integer space Z^2, where each element of a set is a 2-D vector whose coordinates are the (x,y) coordinates of a black(or white) pixel in the image.

Fig 3.7.6 Binary Image

Gray-scale digital images can be represented as sets whose components are in Z^3. In this case, two components of each element of the set refer to the coordinates of a pixel, and the third corresponds to its discrete gray-level value.

- **Segmentation:**

Segmentation procedures partition an image into its constituent parts or objects. In general, autonomous segmentation is one of the most difficult tasks in digital image processing. A rugged segmentation procedure brings the process a long way toward successful solution of imaging problems that require objects to be identified individually.

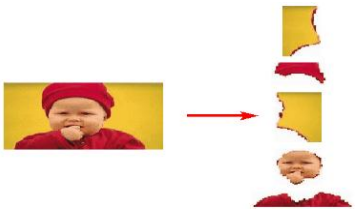

Fig 3.7.7 Segmentation of Image

On the other hand, weak or erratic segmentation algorithms almost always guarantee eventual failure. In general, the more accurate the segmentation, the more likely recognition is to succeed.

- **Representation and description:**

Representation and description almost always follow the output of a segmentation stage, which usually is raw pixel data, constituting either the boundary of a region (i.e., the set of pixels separating one image region from another) or all the points in the region itself. In either case, converting the data to a form suitable for computer processing is necessary. The first decision that must be made is whether the data should be represented as a boundary or as a complete region. Boundary representation is appropriate when the focus is on external shape characteristics, such as corners and inflections.

Regional representation is appropriate when the focus is on internal properties, such as texture or skeletal shape. In some applications, these representations complement each other. Choosing a representation is only part of the solution for transforming raw data into a form suitable for subsequent computer processing.

A method must also be specified for describing the data so that features of interest are highlighted. Description, also called feature selection, deals with extracting attributes that result in some quantitative information of interest or are basic for differentiating one class of objects from another.

- **Object recognition:**

The last stage involves recognition and interpretation. Recognition is the process that assigns a label to an object based on the information provided by its descriptors. Interpretation involves assigning meaning to an ensemble of recognized objects.

- **Knowledgebase:**

Knowledge about a problem domain is coded into image processing system in the form of a knowledge database. This knowledge may be as simple as detailing regions of an image when the information of interests is known to be located, thus limiting the search that has to be conducted in seeking that information. The knowledge base also can be quite complex, such as an inter related to list of all major possible defects in a materials inspection problem or an image data base containing high resolution satellite images of a region in connection with change deletion application.

In addition to guiding the operation of each processing module, the knowledge base also controls the interaction between modules. The system must be endowed with the knowledge to recognize the significance of the location of the string with respect to other components

of an address field. This knowledge glides not only the operation of each module, but it also aids in feedback operations between modules through the knowledge base. We implemented preprocessing techniques using MATLAB.

COMPONENTS OF AN IMAGE PROCESSING SYSTEM:

As recently as the mid-1980s, numerous models of image processing systems being sold throughout the world were rather substantial peripheral devices that attached to equally substantial host computers. Late in the 1980s and early in the 1990s, the market shifted to image processing hardware in the form of single boards designed to be compatible with industry standard buses and to fit into engineering workstation cabinets and personal computers. In addition to lowering costs, this market shift also served as a catalyst for a significant number of new companies whose specialty is the development of software written specifically for image processing.

Although large-scale image processing systems still are being sold for massive imaging applications, such as processing of satellite images, the trend continues toward miniaturizing and blending of general-purpose small computers with specialized image processing hardware. Figure 1.24 shows the basic components comprising a typical general-purpose system used for digital image processing. The function of each component is discussed in the following paragraphs, starting with image sensing.

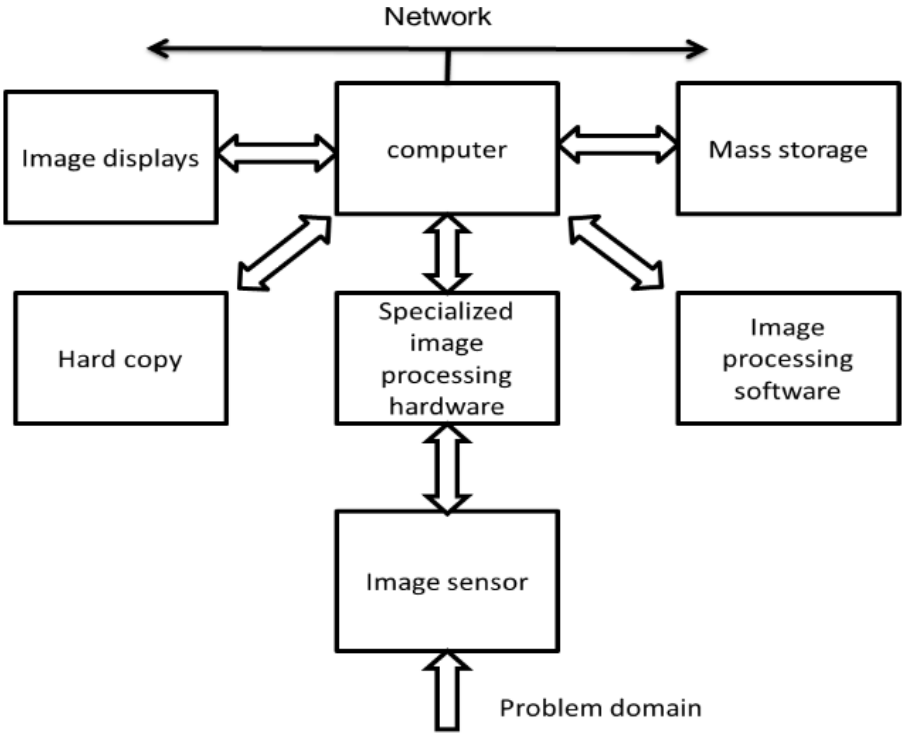

Fig 3.8 Components of An Image Processing System

- **Image sensors:**

With reference to sensing, two elements are required to acquire digital images. The first is a physical device that is sensitive to the energy radiated by the object we wish to image. The second, called a digitizer, is a device for converting the output of the physical sensing device into digital form. For instance, in a digital video camera, the sensors produce an electrical output proportional to light intensity. The digitizer converts these outputs to digital data.

- **Specialized image processing hardware:**

Specialized image processing hardware usually consists of the digitizer just mentioned, plus hardware that performs other primitive operations, such as an arithmetic logic unit (ALU), which performs arithmetic and logical operations in parallel on entire images. One example of how an ALU is used is in averaging images as quickly as they are digitized, for the purpose of noise reduction. This type of hardware sometimes is called a front-end subsystem, and its most distinguishing characteristic is speed. In other words, this unit performs functions that require fast data throughputs (e.g., digitizing and averaging video images at 30 frames) that the typical main computer cannot handle.

- **Computer:**

The computer in an image processing system is a general-purpose computer and can range from a PC to a supercomputer. In dedicated applications, sometimes specially designed computers are used to achieve a required level of performance, but our interest here is on general-purpose image processing systems. In these systems, almost any well-equipped PC-type machine is suitable for offline image processing tasks.

- **Image processing software:**

Software for image processing consists of specialized modules that perform specific tasks. A well-designed package also includes the capability for the user to write code that, as a minimum, utilizes the specialized modules. More sophisticated software packages allow the integration of those modules and general-purpose software commands from at least one computer language.

- **Mass storage:**

Mass storage capability is a must in image processing applications. An image of size 1024*1024 pixels, in which the intensity of each pixel is an 8-bit quantity, requires one megabyte of storage space if the image is not compressed. When dealing with thousands, or even millions, of images, providing adequate storage in an image processing system can be a challenge. Digital storage for image processing applications fall into three principal

categories: (1) short-term storage for use during processing, (2) on-line storage for relatively fast recall, and (3) archival storage, characterized by infrequent access. Storage is measuredin bytes (eight bits), Kbytes (one thousand bytes), Mbytes (one million bytes), Gbytes (meaning giga, or one billion, bytes), and Tbytes (meaning tera, or one trillion, bytes).

One method of providing short-term storage is computer memory. Another is by specialized boards, called frame buffers that store one or more images and can be accessed rapidly, usually at video rates. The latter method allows virtually instantaneous image zoom, as well as scroll (vertical shifts) and pan (horizontal shifts). Frame buffers usually are housed in the specialized image processing hardware unit shown in Fig. 1.24. Online storage generally takes the form of magnetic disks or optical-media storage. The key factor characterizing on-line storage is frequent access to the stored data. Finally, archival storage is characterized by massive storage requirements but infrequent need for access. Magnetic tapes and optical disks housed in "jukeboxes" are the usual media for archival applications.

- **Image displays:**

Image displays in use today are mainly color (preferably flat screen) TV monitors. Monitors are driven by the outputs of image and graphics display cards that are an integral part of the computer system. Seldom are there requirements for image display applications that cannot be met by display cards available commercially as part of the computer system. In some cases, it is necessary to have stereo displays, and these are implemented in the form of headgear containing two small displays embedded in goggles worn by the user.

- **Hard Copy:**

Hardcopy devices for recording images include laser printers, film cameras, heat-sensitive devices, inkjet units, and digital units, such as optical and CD-ROM disks. Film provides the highest possible resolution, but paper is the obvious medium of choice for written material. For presentations, images are displayed on film transparencies or in a digital medium if image projection equipment is used. The latter approach is gaining acceptance as the standard for image presentations.

- **Network:**

Networking is almost a default function in any computer system in use today. Because of the large amount of data inherent in image processing applications, the key consideration in image transmission is bandwidth. In dedicated networks, this typically is not a problem, but communications with remote sites via the Internet are not always as efficient.

CHAPTER 4
EXISTING SYSTEM

The Scanning laser polarimetry devices provide a number of software-generated parameters, the main ones being the software generated parameters as 'the number' and the 'NFI' (nerve fiber indicator). Reports have shown that GDx software generated parameters have limited ability for AMD detection. Several other approaches for the analysis of the scan data available through these devices have been presented to improve upon these results. Locally based techniques like relative surface height, and sectoral-based analysis have been reported with better results than GDx parameters. One approach proposed a linear discriminant function combining GDx parameters and found better result than the 'number'.[16] Another approach proposed discriminant analysis of 30^0 sectoral data from the scanning laser polarimetry. A lot of proposed techniques use the data obtained from a circular ring band around the center of the optic disc in the scan data. The inner radius of the ring is taken to be 1.75 times the disc diameter.

A double hump pattern has been reported to exist in the one-dimensional data thus obtained. Most of the algorithms analyze the changes in the double hump pattern using different techniques. In some more robust approaches the global shape analysis of the double hump curve has been suggested and found to better in identifying AMD patients. These techniques include Fourier analysis suggested with different number of sampling as well as different combinations of the Fourier analysis components.[19-22] Wavelet-Fourier analysis of the one-dimensional data has also been proposed and shown to have better identification power as compared with the 'number'.

The techniques used currently for the analysis of the scan data from the Scanning laser polarimetry devices have chiefly depended on the ring data around the optic disc center. The scan is divided into four sectors known as the Temporal, Superior, Nasal, and Inferior and hence the data obtained from the ring around the optic disc is called the TSNIT graph.

Although these data provide considerable amount of information for AMD recognition, the information in the rest of the scanned image must not be completely neglected. In order to further improve the performance of the classifier, Fourier analysis of the data obtained using the entire region of the scan as well as analysis of the two dimensional Fourier transforms of the scan is proposed.

This thesis presents a unique approach to the analysis of data from scanning laser polarimetry devices by using the entire data in the retinal scan data as an image to generate useful feature vectors. Techniques like taking different angular projections (Radon Transform) of the image (scan data), 2D Fourier Transform and correlation of the scan images with observed pattern images are proposed.

Chapter two provides a brief description and discussion of the disease in question. The third chapter explains concisely the physics of the scanning laser polarimetry device used to obtain the data and gives a brief history of the analysis methods previously used for the scan data. The first phase of the thesis will mainly focus on generation and selection of the useful features through experimentation on available data. The fourth chapter discusses the features that were finally chosen to formulate the classification system. After the selection of final set of features, the feature optimization and final classification steps are explained and discussed in chapter five.

Once the recognition system is realized, the ROC curve analysis is used for the performance evaluation of the proposed analysis algorithm and comparison with one of the previous algorithms using global analysis of the one dimensional ring data are presented in chapter six. The conclusions are discussed in chapter seven. The appendix has parts of the code attached for obtaining the discussed features, their PCA analysis, LDF classification as well as the ROC analysis.

Age-related macular degeneration (AMD)

The term ‘Age-related macular degeneration (AMD)’ refers to a large number of optic nerve diseases, which is associated with loss of visual activity and can lead to total, irreversible blindness if left untreated. AMD retinal image optic neuropathy is the second leading cause of blindness worldwide.

The optic nerve is a cylindrical structure responsible for carrying the visual information out of the eye towards the brain. Neural fibers, the primary component of the optic nerve, are composed of about 1.2-1.5 million ganglion cell axons.

These axons originating in the ganglion cell layer of the retina, the innermost layer of the eye, form the retinal nerve fiber layer (RNFL). These axons collect the visual information and carry it outside the eye via the optic nerve. The nerve head is the distal portion of the optic nerve. The retinal nerves converge upon the nerve head from all points

of the fundus. The portion of the optic nerve head that is clinically visible by an ophthalmoscope is known as the optic disc. The optic nerve head is slightly vertically oval and it is also the site of entry for the retinal vessels. The shape and size of the optic disc is important in evaluation for AMD diagnosis.

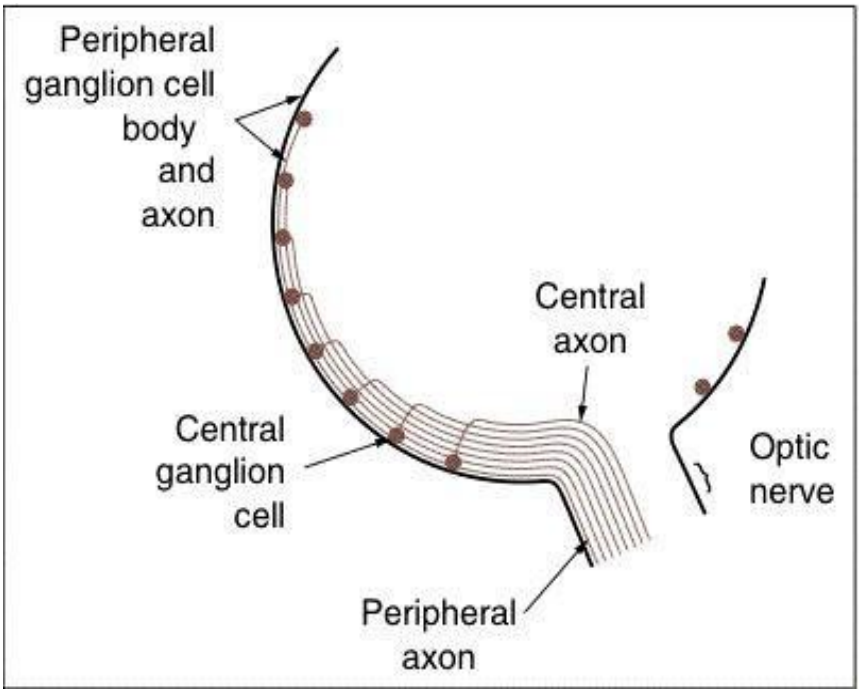

Figure 4.1 Anatomy of the eye (Courtesy: Handbook of Age-related macular degeneration (AMD) (Azuara-Blanco Augusto)

Intraocular Pressure (IOP) is a result of complex interplay of components of the aqueous humor dynamics in the eye. It describes the pressure in the eye due to production and flow of optic fluid known as aqueous humor. A 'normal' IOP (that does not lead to AMD retinal image damage) is usually defined as 15.5 +_ 2.5 mmHg while 'AMD retinal image' IOP is generally described as above 20.5 mmHg. However, AMD patients are known to have IOP within the normal range and raised IOP can be found in non-AMD retinal image eyes. Intraocular pressure is subject to a certain daily variation as well as variation during the same day[24]. Normal eyes show less diurnal variation in the IOP than AMD retinal image eyes. The elevated IOP when beyond that compatible with normal ocular function leads to irreversible damage to the nerve fibers in the retina, thus causing visual impairment. Intraocular pressure has a central role in the treatment of all forms of AMD today. It has been considered the main risk factor for AMD, and almost every treatment for AMD patient is aimed at reducing the IOP. Although raised IOP is considered a big risk factor for AMD, alone it is insufficient for the diagnosis of most forms of AMD. It has been associated with only 50% sensitivity

and 90% specificity.[6] However it still is the primary criterion for making diagnosis for patients with normal optic nerve heads and normal visual fields as well as in cases of congenital and secondary AMD. The most widely used and accepted gold standard for measuring IOP is Goldmann tonometry. Goldmann determined that when an area of 3.06 mm in a human eye is flattened with 520 μm corneal thickness, then resistance of cornea balances with the surface tension and hence could be ignored. This is the main principle on which the tonometer is based.

AMD can cause damage to the optic nerve in a variety of ways. It has been proved that irrespective of the type of damage, the development of visual field defects is always preceded by optic nerve damage in AMD.[8] The appearance of the optic disc is a very important characteristic to determine AMD retinal image damage. The shape of the optic disc in a normal eye is round or horizontally oval. The region in the retina around the optic disc has been divided into four areas, the horizontal sector towards the nose is called the Nasal region, the other horizontal sector being the Temporal, the vertical sector above the disc is known as Superior while the sector below is called Inferior (Figure 2.2). The neural rim around the optic disc is widest in the inferior quadrant, followed by superior, nasal and temporal.

There are various patterns of optic disc changes in AMD, and the detection of change is the diagnosis of AMD. The concentric enlargement of the optic cup, notching, and other similar patterns of AMD retinal image damage are the most commonly found. The optic disc to optic cup ratio is therefore usually taken into consideration while evaluation. However the asymmetries of cup/disc can have other diseases as a cause and are therefore not as reliable. Other features taken into account are the size and shape of

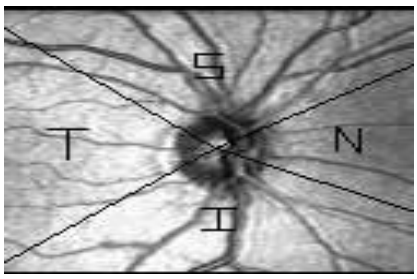

Figure 4.2 Photograph of the optic nerve with optic disc in the center and the areas around it divided into the four sectors – Temporal, Superior, Nasal, and Inferior

The rim around the optic disc and the presence of optic disc hemorrhage. To examine the shape of the optic disc for AMD retinal image changes, ophthalmoscope is generally used. The direct examination of the optic disc through an ophthalmoscope is called

ophthalmoscopy and it can provide useful information for diagnosis of AMD. It is generally performed in a dark room with dilated pupil. Although doctors can detect a lot of features through this technique, it does not yield a permanent record and has interobserver and intraobserver variabilities. Its sensitivity and specificity has been reported to be only 59% and 73%, respectively.

A careful examination and detection of change in the optic nerve and the nerve fiber layer is the key to early diagnosis of AMD. There are several instruments currently available for imaging of optic nerve and the nerve fiber layer, such as red-free photography, the Topcon ImageNet system, the confocal scanning laser ophthalmoscope, the retinal nerve fiber layer analyzer, and the optical coherence tomograph. The main disadvantages of these techniques however, are the lack of adequate amount of research and high cost.

Another very widely used test for AMD is the visual field test. The most common way to measure how well the optic nerve functions is the assessment of the eye's ability to detect the brightness of small points of light both centrally and peripherally. This type of examination is called visual-field testing or perimetry. Although high sensitivity and specificity numbers have been reported for this test,[5] it depends a lot on the conditions of the test, momentary or immediate state of the patient and the design of the test. Hence they need to be augmented by another screening technique for confirmation.

CHAPTER 5
PROPOSED SYSTEM

5.1 Retinal Nerve Fiber Layer and Scanning Laser Polarimetry

The retinal nerve fiber layer is composed of about 1.2-1.5 million ganglion cell axons originating in the retina. The axons are distributed in a characteristic pattern. The axons originating in the region nasal to the optic disc as well as in the macular area run directly toward the optic nerve head, while the axons originating in the temporal section run towards the superior or inferior poles of the optic disc before converging to the nerve head. These fibers are known to be most susceptible to early AMD retinal image damage.

The peripheral ganglion cell axons travel to the optic nerve head in the peripheral position while the central axons take a more superficial path and follow the innermost part within the optic nerve head. Due to the characteristic pattern of the nerve fiber layer axons, the thickness of the nerve fiber layer on the vertical poles of the optic disc is much higher than in the nasal and temporal optic disc poles.

The importance of the detection of RNFL damage as an early sign of Age-related macular degeneration (AMD) has been confirmed by numerous studies. Hoyt and Newman first described it in 1987[25,26]. Histological studies show that as much as 40% of retinal nerve fiber in the eye can be lost without the detection of characteristic visual defect in AMD patients[6]. The findings of Sommer and colleagues showed that RNFL damage could precede visual field loss by up to 5 years[27]. Hence it is believed that the detection of damage in nerve fiber layer can lead to an early detection of AMD.

RNFL defects related to Age-related macular degeneration (AMD) can be either diffused or localized. Localized defects generally include slit-like or groove-like defects in the RNFL. When these slit like defects extend to the disc margin or the wedge shaped defects are seen as notches in the neuroretinal rim in inferior or superior regions, it is judged as a sign of AMD retinal image abnormality.

Although localized defects are easier to detect, diffuse RNFL loss is more common and difficult to diagnose. The second order retinal vessels, which are normally well concealed by the retinal nerve fiber layer, start to be seen in this kind of defects. The progressive loss of RNFL thickness in the superior and inferior poles is a sign of AMD retinal image damage.

There are several different techniques for the qualitative examination and quantitative measurement of nerve fiber damage caused by AMD. The qualitative techniques include examination of the retina through a dilated pupil using an ophthalmoscope or by using a red-free camera or using high-resolution black and white photographs. These are all, however, limited by the pupil size and media optics and tend to have high intra- and interobserver variability. To reduce these difficulties and provide more quantitative measurements of the nerve fiber layer, different devices have been developed. Several instruments have been developed that focus on imaging of the fundus (a mirror like structure just behind the retina which acts as a light amplifier) and analyzing the topography of the retinal surface. These methods attempt to quantify the three-dimensional size and shape of the optic disc, which is considered to represent the bulk of the retinal nerve fibers (Figure 3.1). Stereoscopic fundus photography uses photographs of fundus under different angles to obtain topographic information of the disc.

Confocal scanning laser ophthalmoscopy tries to obtain optical section images of the retina by scanning a laser beam across the eye fundus in two dimensions and provides video images on a monitor. These methods include instruments like the Topcon Imagenet, the Rodenstock Optic Nerve Analyzer (Rodenstock Instruments, Munich, Germany) and Heidelberg Retina Tomograph (Heidelberg Engineering, Heidelberg, Germany). Although these methods are a reasonable indicator of the condition of the optic disc, the analysis of the topography of the fundus is an indirect measure of the nerve fiber layer and is only suggestive. Furthermore, the ultimate resolution of these methods is limited by the properties of the ocular media. Hence these kinds of imaging systems are not suitable for accurate measurement of the retinal nerve fiber layer thickness.

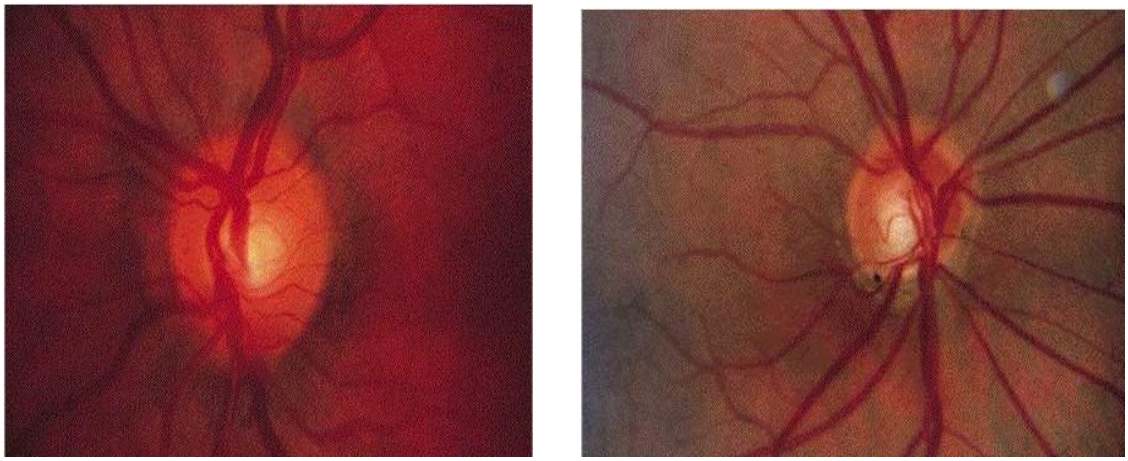

Figure 5.1 Example of optic disc photography *a)* normal disc *b)* notching in optic disc (Courtesy: Handbook of Age-related macular degeneration (AMD) (Azuara-Blanco Augusto)

Scanning Laser Polarimetry (SLP) is a technique of providing a more quantitative measure of the thickness of the RNFL. The method is based on the principle of using imaging polarimetry to detect the birefringence of the retinal nerve fiber layer[7,9] (Figure 3.2). This technique utilizes the polarization properties of the retinal nerve fiber layer. The nerve fiber layer and other regions of the retina have been known to have polarization properties or birefringent properties. Form birefringence occurs when a medium consists of parallel cylindrical structure with diameters smaller than the wavelength of light passing through it. A birefringent structure has the ability to change the polarization of a polarized light double passing it[29].

The ganglion axons that constitute the nerve fiber layer are essentially cylindrical rod-like structures that are parallel to the retinal surface and have extremely small diameters. When a light beam, perpendicular to its surface, is impinged on the retina, the reflected light is split into two rays that travel at different velocities. This is due to the birefringence of the retinal nerve fibers, which being parallel to the retina, are essentially perpendicular to the scanning laser beam. This delay in the two emerging rays or the phase shift between the two is known as the retardation. The amount of retardation is found to be dependent on the thickness of the RNFL due to its properties and it can be measured using a polarimeter. This retardation (degrees) information is converted to thickness (microns) through the conversion factor based on the histological comparison with monkey eyes.

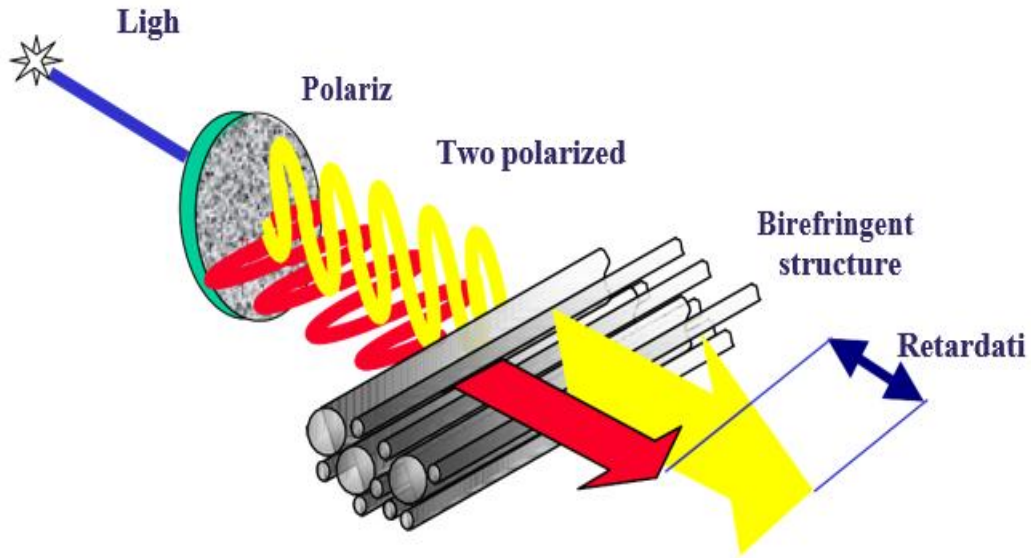

Figure 5.2 Scanning Laser Polarimetry Device - Principle (Courtesy: Laser Diagnostics, San Diego, CA)

The Scanning Laser Polarimeter uses this principle to scan the thickness of the retinal nerve fiber by employing a low power near infrared laser beam to illuminate the human

retina. The device using this technology is currently available through Laser Diagnostic Technologies, Inc, San Diego, CA. The device scans the retina with laser beam and measures the retardation at 65,536 discrete points within the retinal area of 15° by 15° in less than 1 second. The software application displays this retardation information on a computer screen as a color-mapped image of the retina (Figure 3.3). A grayscale image of the thickness map of the same retina is also showed in figure 3.4 for reference. The software also provides other software generated information and parameters extracted from the thickness map. The company has generated a normative database using the thickness maps obtained from variety of patients as well as normal eyes. This is then used to compare the parameters obtained from current patient to provide the probability of the patient having AMD based on those parameters. These computer generated parameters include summary measures based on the calculation circle.

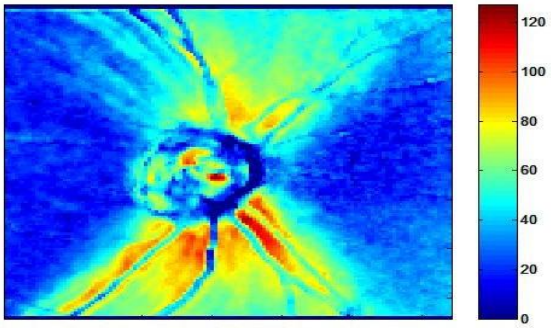

Figure 5.3 Color Coded RNFL thickness map of the 65356 points obtained by the SLP (with the color scale on the right).

Among the number of software-generated parameters provided by the company, the main ones are 'the number' and the 'NFI' (nerve fiber indicator). The parameter "number" is obtained through a neural network, which is fed with around 100-200 features from the scanned image. The NFI is obtained through a support vector machines recognizer and is available in the newer versions of the device. The latest version of the device implements the correction for the birefringent properties of the parts of the eye other than the nerve fiber layer and is called GDx VCC. Currently doctors use the above- mentioned factors from the device along with other tests and gauge.

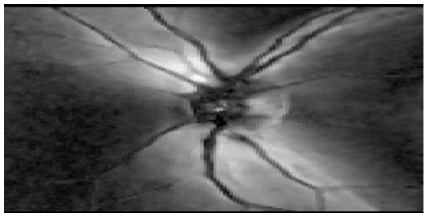

Figure 5.4 Grayscale representation of the RNFL thickness map image

CHAPTER 6

RESULT

6.1 RESULT

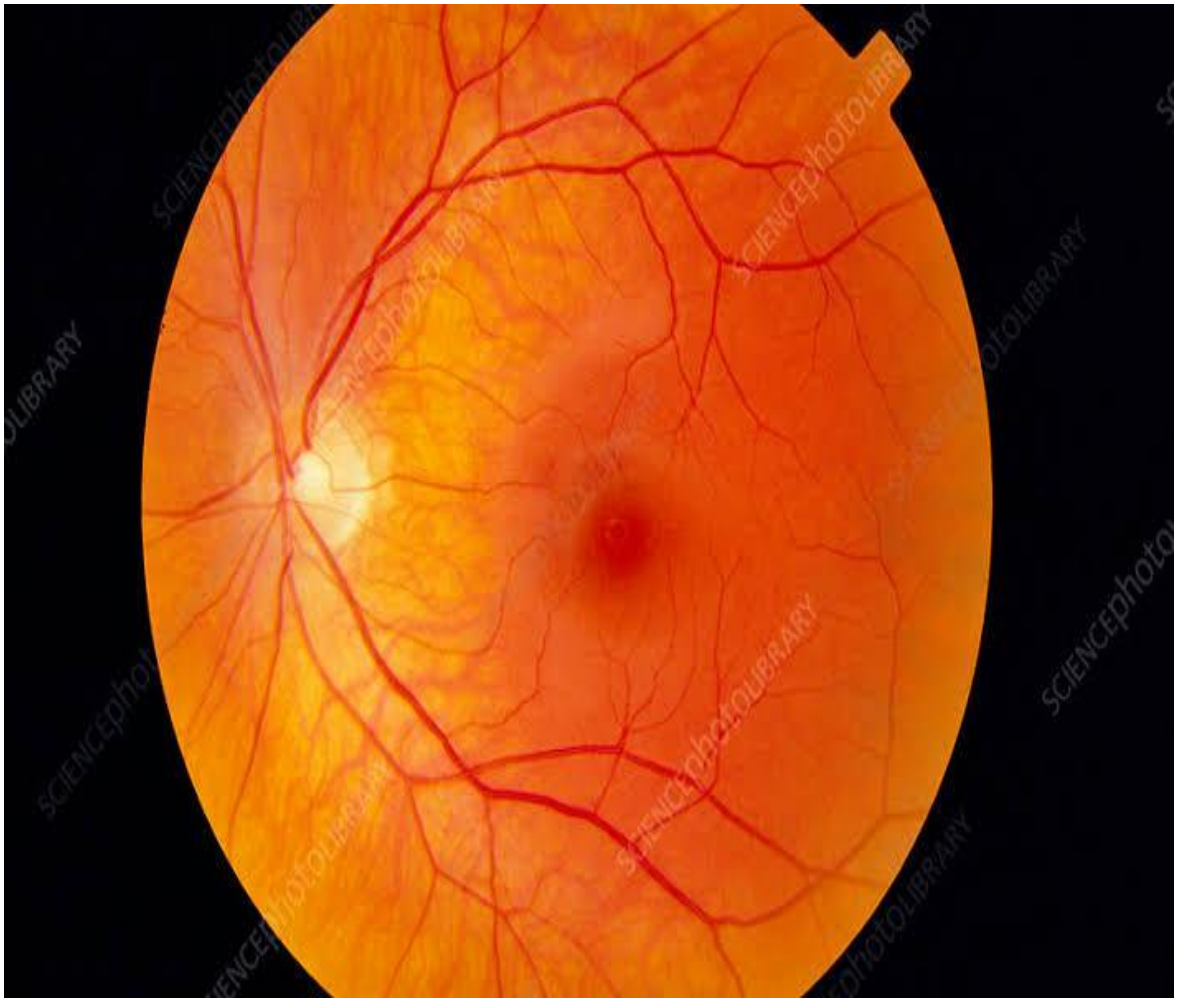

FIG: 6.1 Healthy retina

Fig:6.2 Age-related macular degeneration

Our present sample consists of SLP scans of 92 AMD retinal image eyes and 92 normal eyes, obtained from 47 Age-related macular degeneration (AMD) patients and 45 healthy people. Dr. Michael Sinai of Laser Diagnostics made the data set and its diagnosis (classification based on clinical findings) available to us for research purposes. This set was divided randomly and uniformly into two subsets. One subset was used as a trainee set for +the classifier while the other was used as a test set. After the classification results were obtained, the ROC analysis for performance evaluation.

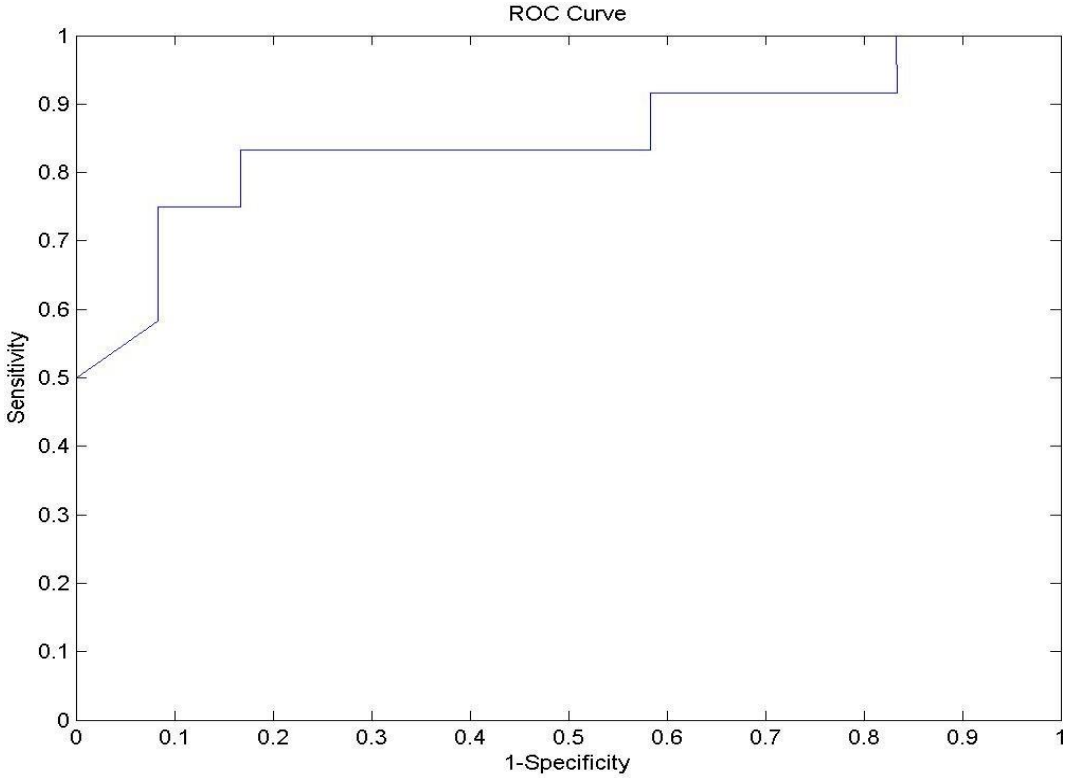

FIG: 6.3 ROC Curve

Method	Sensitivity	Specificity	Area under ROC curve
FT of 90^0 projection	75.12%	99.15%	0.7789
Wavelet analysis with FT of 90^0 projection	58.7%	89%.	0.8235
Ring data from 2D FT	74.92%	91.5%	0.7813
Correlation coefficient with pattern image	67.5%	82%	0.8287
Combined Feature set	84.78%	91.3%	0.8433
FFTA – based on previously performed analysis	76.09	91.3	0.8588

Table 6.1 Performance evaluation of the different feature sets based on the ROC analysis

6.2 APPLICATIONS

 Detection and diagnosis of Age-related Macular Degeneration (AMD): Utilizing computer-assisted imaging technologies to detect structural changes in the retinal nerve fiber layer for early AMD diagnosis.

 Glaucoma screening and diagnosis: Developing automated systems for glaucoma detection using image processing and analysis of retinal images.

 Retinal image analysis: Processing retinal images to assist clinical diagnosis and treatment, with a focus on automated diagnosis of retinal fundus images.

6.3 ADVANTAGES

- It discusses techniques that "aid a physician in detecting possible subtle abnormalities."
- The application of digital imaging to ophthalmology offers the possibility of processing retinal images to assist clinical diagnosis and treatment.
- The development of an automated system for analyzing the images of the retina will facilitate computer-aided diagnosis of eye diseases.
- The automated system can detect glaucoma efficiently and in less time.

CHAPTER 7
CONCLUSION & FEATURE SCOPE

7.1 CONCLUSION

Current techniques for analyzing data from Scanning Laser Polarimetry devices primarily focus on the ring data around the optic disc center. These methods, while informative, may not utilize all available information within the scan. To improve the performance of classifiers, it is proposed to use the entire region of the scan to generate more comprehensive and effective feature vectors.

7.2 FEATURE OPTIMIZATION

The main objective of this step in the classification process is to reduce the dimension of the feature vector without losing the variability of the feature set. Principal Component analysis (PCA) identifies linear combinations of the feature set so that most of the variability (information) of the original feature set is contained within that combination[11,12,37]. It essentially transforms a feature vector with correlated variables into a smaller sized feature vector with uncorrelated variables.

These uncorrelated variables are called the principal components. The first principal component is the projection of the data points (points in the feature set/ feature vector) in the direction of the line giving the best orthogonal regression fit to the data points. Since the best fit to this type should pass through the mean, the data points are centered on the mean by subtracting the mean from the data points. The first principal component is hence the projection of the data points into the direction with maximal variance of the projected points. The first principal component corresponds to the maximum variability of the original feature set and the second component corresponds to the second highest variability of the set and so forth, there are *p* principal components (p is the feature vector size).

The first step in the PCA is to find the sample covariance matrix *S,* for the combined samples of both classes. Then the eigenvalues and corresponding eigenvectors of this matrix are calculated. The combination of the feature points that has the maximum variability is obtained in the direction of the first principal component and this direction is that of the eigenvector corresponding to the highest eigenvalues.

REFERENCES

[1] J. Kim, M. Kim, H. Kang, and K. H. Lee, “U-gat-it: Unsupervised generative attentional networks with adaptive layer-instance normalization for image-to-image translation,” in International Conference on Learning Representations, 2020.

[2] H. Fu, B. Wang, J. Shen, S. Cui, Y. Xu, J. Liu, and L. Shao, “Evaluation of retinal image quality assessment networks in different color-spaces,” Medical Image Computing and Computer Assisted Intervention MICCAI 2019, p. 4856, 2019.

[3] H. Zhao, B. Yang, L. Cao, and H. Li, Data-Driven Enhancement of Blurry Retinal Images via Generative Adversarial Networks, 10 2019, pp. 75–83.

[4] M. Zhou, K. Jin, S. Wang, J. Ye, and D. Qian, “Color retinal image enhancement based on luminosity and contrast adjustment,” IEEE Transactions on Biomedical Engineering, vol. 65, no. 3, pp. 521–527, 2018.

[5] D. B. Russakoff, A. Lamin, J. Oakley, A. Dubis, and S. Sivaprasad, “Deep learning for prediction of amd progression: A pilot study.” Investigative ophthalmology and visual science, vol. 60 2, pp. 712–722, 2019.

[6] G. Bhupendra and M. Tiwari, “Color retinal image enhancement using luminosity and quantile based contrast enhancement,” Multidimensional Systems and Signal Processing, 01 2019.

[7] M. Zhou, K. Jin, S. Wang, J. Ye, and D. Qian, “Color retinal image enhancement based on luminosity and contrast adjustment,” IEEE Transactions on Biomedical Engineering, vol. 65, no. 3, pp. 521–527, 2018.

[8] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, “Unpaired image-toimage translation using cycle-consistent adversarial networks,” 2017 IEEE International Conference on Computer Vision (ICCV), Oct 2017.

[9] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, “Inception-v4, inception-resnet and the impact of residual connections on learning,” in AAAI, 2017.

[10] P. Dai, H. Sheng, J. Zhang, L. Li, J. Wu, and M. Fan, “Retinal fundus image enhancement using the normalized convolution and noise removing,” International Journal of Biomedical Imaging, vol. 2016, pp. 1–12, 01 2016.

[11] X. Huang, M.-Y. Liu, S. Belongie, and J. Kautz, “Multimodal unsupervised image-to-image translation,” Lecture Notes in Computer Science, p. 179196, 2018.

[12] Y. Peng, S. Dharssi, Q. Chen, T. Keenan, E. Agrn, W. Wong, E. Chew, and Z. lu, “Deepseenet: A deep learning model for automated classification of patient-based age-

related macular degeneration severity from color fundus photographs," Ophthalmology, vol. 126, 11 2018.
[13] Tjon-Fo, MJ Sang, HG Lemij, The sensitivity and specificity of nerve fiber layer measurements in Age-related macular degeneration (AMD) as determined with scanning laser polarimetry, Am J Ophthalmology, 1997; 123: 62–69.
[14] RN Weinreb, L Zangwill, CC Berry, et al., Detection of Age-related macular degeneration (AMD) with scanning laser polarimetry, Arch Ophthalmology, 1999; 117:1298-1304
[15] JR Trible, RD Schultz, JC Robinson, et al. Accuracy of scanning laser polarimetry in the diagnosis of Age-related macular degeneration (AMD), Archives of Ophthalmology, 1993; 117:1298-304

APPENDIX

```
Clc
clear all
close all
[input, pathname] = uigetfile('*.jpg');
if isequal(input,0)
   disp('User selected Cancel')
else
   disp(['User selected ', fullfile(pathname, input)])
end
inpimg=imread(input);
imshow(inpimg)
title('input image ')
%For disk image
red=inpimg(:,:,1);
green=inpimg(:,:,2);
%figure,imshow(green)
%title('green max image ')
black=inpimg(:,:,3);
nefilt=green>130;
binaryImage=nefilt;
% Get rid of stuff touching the border
binaryImage = imclearborder(binaryImage);
fill=imfill(binaryImage,'holes');
 flatten1=strel('disk',6);
diskimg=imdilate(fill,flatten1);
 %figure,imshow(diskimg)
%title('disk image ')
%For the Cup Image
nefilt=green>140;
binaryImage=nefilt;
% Get rid of stuff touching the border
binaryImage = imclearborder(binaryImage);
```

```
cup=imfill(binaryImage,'holes');
flatten2=strel('disk',2);
dilate=imdilate(cup,flatten2);
cupimg=dilate;
%figure,imshow(cup)
%title('cup image ')
% Extract only the two largest blobs.
BW=binaryImage ;
CC = bwconncomp(BW);
numPixels = cellfun(@numel,CC.PixelIdxList);
[biggest,idx] = max(numPixels);
BW(CC.PixelIdxList{idx}) = 0;
filteredForeground=BW;
figure, imshow(BW);
binaryImage = imfill(binaryImage, 'holes');
% % Display the binary image.
dis(:,:,1)=immultiply(binaryImage,inpimg(:,:,1));
dis(:,:,2)=immultiply(binaryImage,inpimg(:,:,2));
dis(:,:,3)=immultiply(binaryImage,inpimg(:,:,3));
a = diskimg;
stats = regionprops(double(a),'Centroid',...
    'MajorAxisLength','MinorAxisLength');
centers = stats.Centroid;
diameters = mean([stats.MajorAxisLength stats.MinorAxisLength],2);
radii = diameters/2;
% Plot the circles.
 figure,imshow(inpimg)
hold on
viscircles(centers,radii);
hold off
figure
subplot(3,3,1)
imshow(inpimg)
```

```
title('input image ')
subplot(3,3,2)
imshow(diskimg,[])
title('disk segment image ')
subplot(3,3,3)
imshow(inpimg)
hold on
viscircles(centers,radii);
hold off
title('Disc boundary')
subplot(3,3,4)
imshow(dilate,[])
title('cup image ')
subplot(3,3,5)
imshow(inpimg)
hold on
viscircles(centers,radii/2);
hold off
title('cup boundary')
c1=bwarea(diskimg);
c2=bwarea(dilate);
cdr=c2./(c1)
rim=(1-dilate)-(1-diskimg);
RDR=bwarea(rim)./(c2);
nn=sprintf('The CDR is  %2f ',cdr)
msgbox(nn)
pause(2)
nn1=sprintf('The RDR is  %2f ',RDR/2)
msgbox(nn1)
pause(2)
if cdr<0.45
   msgbox('NO AMD')
elseif cdr <0.6 & cdr>0.45
```

```
    msgbox('AMD DETECTED:Medium risk')
else cdr>0.6
    msgbox('AMD DETECTED:High risk')
end
```