

A

Project Report on

**“IMPLEMENTATION OF IMAGE ENHANCEMENT TECHNIQUES USING  
HARDWARE AND SOFTWARE ”**

*Submitted in partial fulfillment of the requirements for the award of the degree of*

**BACHELOR OF TECHNOLOGY**

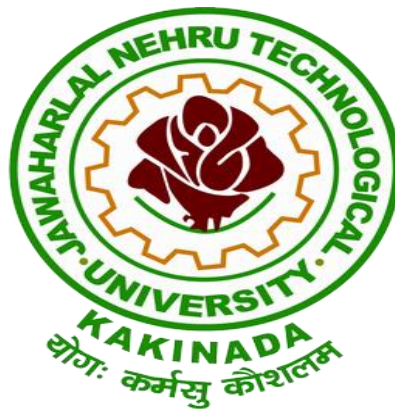
**IN**

**ELECTRONICS AND COMMUNICATION ENGINEERING**

**BY**

<b>P.BALA SAI MANASA</b>	<b>(16VV1A0440)</b>
<b>SK.SHAKEERA</b>	<b>(17VV1A0465)</b>
<b>K.VARSHINI</b>	<b>(16VV1A0419)</b>
<b>Y.HEMANTH</b>	<b>(16VV1A0456)</b>
<b>P.DORABABU</b>	<b>(16VV1A0444)</b>

Under the esteemed guidance of  
**DR.CH.SRINIVASA RAO.**  
Professor of ECE Department



**DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING**

**UNIVERSITY COLLEGE OF ENGINEERING VIZIANAGARAM  
JNTUK VIZIANAGARAM**

**2016-2020**

**UNIVERSITY COLLEGE OF ENGINEERING VIZIANAGARAM**  
**JNTUK VIZIANAGARAM**  
**DEPARTMENT OF**  
**ELECTRONICS AND COMMUNICATION ENGINEERING**



**CERTIFICATE**

This is to certify that the project report titled **“IMPLEMENTATION OF IMAGE ENHANCEMENT TECHNIQUES USING HARDWARE AND SOFTWARE ”**, being submitted by P.Bala Sai Manasa (16VV1A0440), Sk.Shakeera (17VV5A0465), K.Varshini (16VV1A0419), Y.Hemanth (16VV1A0456), P.Dorababu (16VV1A0444) in partial fulfillment of the requirements for the award of the Degree of Bachelor of Technology in **ELECTRONICS AND COMMUNICATION ENGINEERING**. It is a record of bonafide work carried out by them under the esteemed guidance and supervision of **Dr.Ch.Srinivasa Rao**.

**Project Guide**

**Dr.Ch.Srinivasa Rao,**

**Professor**

**Department of E.C.E.**

**Head of Department**

**Dr. K.C.B. RAO,**

**Professor & H.O.D,**

**Department of E.C.E.**

**External Examiner**

## DECLARATION

We, P.Bala Sai Manasa (16VV1A0440), Sk.Shakeera (17VV5A0465), K.Varshini (16VV1A0419), Y.Hemanth (16VV1A0456), P.Dorababu (16VV1A0444) declare that the project report titled “**IMPLEMENTATION OF IMAGE ENHANCEMENT TECHNIQUES USING HARDWARE AND SOFRWARE**”, submitted to University College of Engineering Vizianagaram, JNTUK in partial fulfillment of the requirements for the award of the degree of **B.Tech in Electronics and Communication Engineering** is a record of original and independent research work done by us during 2019-2020 under the supervision of **Dr.Ch.Srinivasa Rao**, Professor, **Department of E.C.E** and it has not formed the basis for the award of any Degree/Diploma/Associate Ship/Fellowship or other similar title to any candidate in any university.

DATE:

ROLL.NO	NAME	SIGNATURE
16VV1A0440	P. BALA SAI MANASA	
17VV5A0465	SK.SHAKEERA	
16VV1A0419	K.VARSHINI	
16VV1A0456	Y.HEMANTH	
16VV1A0444	P.DORABABU	

## ACKNOWLEDGEMENTS

We deeply express our hearty gratitude and thanks to our project guide **Dr.Ch.Srinivasa Rao**, Professor, Department of Electronics and Communication Engineering, for his guidance and cooperation for completing this project. We convey our heartfelt thanks to him for his inspiring assistance till the end of our project.

We have great pleasure in expressing our deep sense of gratitude to **Prof.K.C.B.RAO, HOD**, Department of Electronics and Communication Engineering, for his inspiring guidance, constant encouragement and support throughout our project work.

We express our deep sense of gratitude to the project review committee members, of Electronics and Communication Engineering Department, **Sri.R.Gurunadha, Dr.T.S.N.Murthy, Sri.A.Gangadhar, Sri.G.Appalanaidu, Smt.B.Nalini, Smt.M.Hema**. We are very much obliged to them for their help, affection and strain taken in reviewing the project at every stage of its progress.

We would like to thank **Prof.G.Swami Naidu**, Principal of **UNIVERSITY COLLEGE OF ENGINEERING VIZIANAGARAM JNTUK**, for the facilities provided in the college in carrying out the project successfully.

We thank the **Teaching and Non-teaching staff members** of our **E.C.E DEPARTMENT**, the college and administration who helped us directly or indirectly in carrying out this project successfully.

Yours Sincerely,

P.BALA SAI MANASA	16VV1A0440
SK.SHAKEERA	17VV5A0465
K.VARSHINI	16VV1A0419
Y.HEMANTH	16VV1A0456
P.DORABABU	16VV1A0444

## TABLE OF CONTENTS

<b>ABSTRACT .....</b>	<b>iii</b>
<b>LIST OF FIGURES .....</b>	<b>iv</b>
<b>LIST OF TABLES .....</b>	<b>vi</b>
<b>CHAPTER-1 .....</b>	<b>1</b>
<b>INTRODUCTION .....</b>	<b>2</b>
1.1.Introduction to Image Processing .....	2
1.1.1. Low-Level Processing.....	3
1.1.2. Mid-Level Processing .....	3
1.1.3. High-Level Processing.....	3
1.2.Introduction to Image Enhancement.....	4
1.3.Matlab.....	5-8
1.4.Xilinx-Vivado .....	9-12
1.5.Google colab Python.....	13-17
1.6.Motivation .....	18
1.7.Objectives .....	19
1.8.Project over view .....	19
<b>CHAPTER-2 .....</b>	<b>20</b>
<b>LITERATURE SURVEY .....</b>	<b>21</b>
2.1.Introduction .....	21
2.2.Different Image Enhancement Techniques .....	22-24
<b>CHAPTER-3 .....</b>	<b>25</b>
<b>IMAGE ENHANCEMENT TECHNIQUES .....</b>	<b>26</b>
3.1.Need for Image Enhancement .....	26
3.2.Different Image Enhancement Techniques .....	26
3.2.1.Image Negative .....	27
3.2.2.Log Transform.....	28-29
3.2.3.Gamma Correction.....	30-31
3.3.4.Contrast Stretching .....	32-33

3.3.5.Histogram Equalization.....	34-35
3.3.6.Median Filtering .....	36-37
3.3.Performance Measures used .....	38
3.4.Image Enhancement using MATLAB .....	39-42
3.5.Image Enhancement using XILINX-VIVADO .....	43
3.6.Image Enhancement using PYTHON.....	44
<b>CHAPTER-4 .....</b>	<b>45</b>
<b>RESULTS AND DISCUSSIONS.....</b>	<b>46-54</b>
<b>CHAPTER-5 .....</b>	<b>55</b>
<b>CONCLUSIONS AND FUTURE SCOPE.....</b>	<b>56</b>
5.1.Conclusion .....	56
5.2.Future scope.....	57
<b>REFERENCES .....</b>	<b>58-60</b>
<b>APPENDIX.....</b>	<b>61-63</b>

## **ABSTRACT**

Image enhancement is the processing of images to improve their appearance to human viewers or to enhance other image processing systems performance. Methods and objectives vary with the application. When images are enhanced for human viewers, as in television, the objective may be to improve perceptual aspects: image quality, intelligibility, or visual appearance. In other applications, such as object identification by machine, an image may be pre-processed to aid machine performance. Because, the objective of image enhancement is dependent on the application and the criteria for enhancement are often subjective or too complex to be easily converted to useful objective measures. Image enhancement algorithms tend to be simple, qualitative and without ambiguity.

In this project, we are going to implement image enhancement techniques using both hardware and soft ware. The techniques include Median filtering, Image Negatives, Power law transform, Contrast Stretching etc. We also implement image enhancement using histogram equalization. We study all these techniques and implement them in both hardware and software. Implementation will be carried out on hardware board and software's used include MATLAB, XILINX and PYTHON.

## LIST OF FIGURES

Figure no	Description	Page no
1.1	Different levels of image processing	3
1.2	Vivado getting start page	11
3.1	Different enhancement Techniques	26
3.2	Example of image negative	27
3.3	Log transform for 8-bit image	28
3.4	Example of Log transform	29
3.5	Curve with different gamma values	30
3.6	Example for gamma correction	31
3.7	A typical transformation function	32
3.8	Example of contrast stretching	33
3.9	Example for histogram equalization	35
3.10	Example for Median filtering	38
3.11	Example for binary image	40
3.12	Example for intensity image	41
3.13	Example for RGB image	41
3.14	Xilinx vivado project navigator	43
4.1	Result of image negative	46
4.2	Result of log transform	46
4.3	Result of gamma correction	47
4.4	Result of contrast stretching	48
4.5	Result of histogram equalization	49
4.6	Result of median filtering with salt and pepper noise	50
4.7	Result of median filtering with Gaussian noise	50
4.8	Result of median filtering with	50



	Speckle noise	
4.9	Result of median filtering with	51
	Poisson noise	
4.10	Result of image negative	52
4.11	Result of image negative using python	52
4.12	Result of gamma correction	53
4.13	Result of histogram equalization	54

## **LIST OF TABLES**

<b>Table no</b>	<b>Description</b>	<b>Page no</b>
1.1	Comparison of different levels of image enhancement	3
4.1	Comparison of PSNR values of different Noises	51

# **CHAPTER 1**

# INTRODUCTION

## 1.1 INTRODUCTION TO IMAGE PROCESSING

Image Processing is a method to convert an image into digital form and perform some operations on it, in order to get an enhanced image or to extract some useful information from the image. It is a type of signal dispensation in which input is a image, like video frame or photograph and the output may be image or characteristics associated with that image. Usually Image Processing system includes treating images as two dimensional signals while applying already set in signal processing methods to them.

Image Processing basically includes the following three steps:

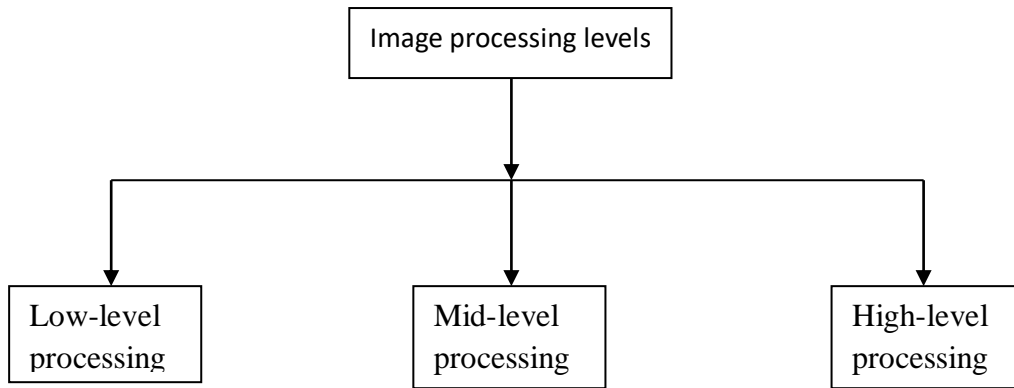
- Importing the image with optical scanner or by Digital photography.
- Analyzing and manipulating the image which includes data compression and image enhancement and spotting patterns that are not visible to human eyes like satellite photographs.
- Output is the last stage in which result can be altered image or the it depend on image analysis.

Image processing is processing of images using mathematical operations by using any form of signal processing for which the input is an image, such as a photograph or video frame; the output of image processing may be either an image or a set of characteristics or parameters related to the image. Most image-processing techniques involve treating the image as a two-dimensional signal and applying standard signal processing techniques to it. Image processing usually refers to digital image processing, but optical and analog image processing also are possible. This article is about general techniques that apply to all of them. The acquisition of images referred to as imaging.

### **Purpose of image processing:**

The purpose of image processing is divided into 5 groups:

- Visualization – Observe the objects that are not visible.
- Image sharpening and restoration – To create a better image.
- Image retrieval – Seek for the image of interest.
- Measurement of pattern – Measures various objects in an image.
- Image Recognition – Distinguish the objects in an image.



**Fig.1.1 Different levels of image processing**

### **1.1.1. LOW-LEVEL PROCESSING:**

Low-level image processing is mainly concerned with extracting descriptions from the images that are usually represented as images themselves. Low-level image processing involves primitive operation such as image preprocessing to reduce noise, contrast enhancement, image sharpening etc. In the low-level process, both input and output are images.

### **1.1.2. MID-LEVEL PROCESSING:**

Mid-level image processing is mainly concerned with extracting descriptions of the scene from the image descriptions extracted at low level. The output is usually in symbolic form, describing the position and shape of portions. It involves tasks such as image segmentation, description of images, object recognition, etc. In this inputs are generally images but its outputs are generally image attributes.

### **1.1.3. HIGH-LEVEL PROCESSING:**

High-level processing involves “making sense” from a group of recognized objects. This process is associated with computer vision. It is characterized by fact that its inputs generally are attributes extracted from image but outputs are images.

**Table 1.1 Comparison of different levels of image processing.**

TYPE	INPUT	OUTPUT	EXAMPLE
Low-level process	Image	Image	Noise removal, image sharpening
Mid-level process	Image	Attributes	Object recognition, Segmentation
High-level process	Attributes	Understanding	Scene understanding, autonomous navigation

## **1.2. INTRODUCTION TO IMAGE ENHANCEMENT:**

Image enhancement refers to accentuation, or sharpening of image features such as edges, boundaries, or contrast to make a graphic display more useful for display and analysis. The enhancement process does not increase the inherent information content in the data. But it does increase the dynamic range of the chosen features so that they can be detected easily. Image enhancement is used to improve the quality of an image for visual perception of human beings. It is also used for low level vision applications.

Image enhancement is the mechanism to process the input image to make it more appropriate and clearly visible for the required application. Image enhancement improves the information content of the image and alters the visual impact of the image on the observer. The objective of image enhancement is to modify attributes of an image to make it more suitable for a specific task. In the image enhancement process, one or more attributes of the image are modified and processed. The choice of attributes and the way they will be modified are specific to a given task. Enhancement is pre processing step in some computer vision applications to ease the vision task, for example to enhance the edges of an object to facilitate guidance of robotic gripper. Enhancement is also used as pre processing step in applications where human viewing of an image is required before further processing. Image enhancement is used for post processing generate a desirable image.

## **1.3. MATLAB**

MATLAB is a high-performance language for technical computing. 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, which together represent the state of art in software for matrix computation. 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. MATLAB feature a family of application specific solutions called toolboxes. Toolboxes are comprehensive collection of MATLAB functions (M-file) that extends the MATLAB environment to solve particular classes of problems. Areas in which toolboxes are available include signal processing, image processing, control system, neural networks, fuzzy logic and many others .This high performance numerical computation package with built in function for graphics and animation. It can perform symbolic algebra like mathematical, Maple and Matrix X. This allows solving 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 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.

### ➤ **Features of MATLAB**

- High-level language for technical computing. Development environment for managing code, files, and data.
- Interactive tools for iterative exploration, design, and problem solving.
- Mathematical functions for linear algebra, statistics, Fourier analysis, filtering, optimization, and numerical integration.
- 2-D and 3-D graphics functions for visualizing data.
- Tools for building custom graphical user interfaces.
- Functions for integrating MATLAB based algorithms with external applications and languages, such as C, C++, FORTRAN, Java TM, COM, and Microsoft Excel.

### ➤ **MATLAB System**

The MATLAB system consists of these main parts: Desktop Tools and Development Environment This part of MATLAB is the set of tools and facilities that help you use and become more productive with MATLAB functions and files. Many of these tools are graphical user interfaces. It includes: the MATLAB desktop and Command Window, an editor and debugger, a code analyzer, and browsers for viewing help, the workspace, and folders. Mathematical Function Library This library 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 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 programs you do not intend to reuse. You can also do “programming in the large” to create complex application programs intended for reuse. 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 wells as to build complete graphical user interfaces on your MATLAB applications.

### ➤ **External Interfaces**

The external interfaces library allows you to write C/C++ and FORTRAN programs that interact with MATLAB. It includes facilities for calling routines from MATLAB (dynamic linking), for calling MATLAB as a computational engine, and for reading and writing MAT files.

## ➤ **Development Environment**

This section explains about the developing environment of MATLAB like how to start & quit MATLAB and about MATLAB tools etc.

### ➤ **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.

### ➤ **Start and Quit**

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. 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.

### ➤ **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, 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.

### ➤ **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 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.

### ➤ **Workspace Browser**

The MATLAB workspace consists of the set of variables (named arrays) built up during a MATLAB session and stored in memory. Variables can be added 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.

### ➤ **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 Math Works 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. 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 Math Works toolboxes are included in the search path.

### ➤ 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, if no.of 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' is not the same variable. To view the matrix assigned to any variable, simply enter the variable name.

## 1.4. XILINX-VIVADO:

The Vivado Design Suite is designed to improve productivity. This tool suite is architected to increase the overall productivity for designing, integrating, and implementing systems using the Xilinx Ultra Scale and 7 series devices, Zynq Ultra Scale MPSoC device, and Zynq-7000 All Programmable (AP) SoC. Xilinx devices are now much larger and come with a variety of new technology, including stacked silicon interconnect (SSI) technology, up to 28 gigabyte (GB) high speed I/O interfaces, hardened microprocessors and peripherals, analog mixed signal, and more. These larger and more complex devices create multidimensional design challenges, when handled incorrectly, that can prevent the achievement of faster time-to-market and increased productivity. With the Vivado Design Suite, you can accelerate design implementation with place and route tools that analytically optimize for multiple and concurrent design metrics, such as timing, congestion, total wire

length, utilization and power. The Vivado Design Suite provides you with design analysis capabilities at each design stage. This allows for design and tool setting modifications earlier in the design processes where they have less overall schedule impact, thus reducing design iterations and accelerating productivity.

The Vivado Design Suite replaces the existing Xilinx ISE Design Suite of tools. It replaces all of the ISE Design Suite point tools, such as Project Navigator, Xilinx Synthesis Technology (XST), implementation, CORE Generator tool, Timing Constraints Editor, ISE Simulator (ISim), Chip Scope Analyzer, Xilinx Power Analyzer, FPGA Editor, Plan Ahead design tool, and Smart Xplorer. All of these capabilities are now built directly into the Vivado Design Suite and leverage a shared scalable data model. Built on the shared scalable data model of the Vivado Design Suite, the entire design process can be executed in memory without having to write or translate any intermediate file formats, which accelerates run times, debug, and implementation while reducing memory requirements.

All of the Vivado Design Suite tools are written with a native tool command language (Tcl) interface. All of the commands and options available in the Vivado Integrated Design Environment (IDE), which is the graphical user interface (GUI) for the Vivado Design Suite, are accessible through Tcl. The Vivado Design Suite also provides powerful access to the design data for reporting and configuration as well as the tool commands and options.

You can interact with the Vivado Design Suite using :

- GUI-based commands in the Vivado IDE.
- Tcl commands entered in the Tcl Console in the Vivado IDE, in the Vivado Design Suite Tcl shell outside the Vivado IDE, or saved to a Tcl script file that is run either in the Vivado IDE or in the Vivado Design Suite Tcl shell.
- A mix of GUI-based and Tcl commands.

## ➤ **Migrating Designs to the Vivado Design Suite:**

### **Migration Considerations**

When migrating, consider the following:

- **IP:** You can migrate existing ISE Design Suite projects and IP to Vivado Design Suite projects and IP. The Vivado Design Suite can use ISE Design Suite IP during implementation. However, updating to the latest Vivado Design Suite native IP is highly recommended to leverage the latest IP updates and to use proper constraints. The Vivado Design Suite is tested and validated with native Vivado Design Suite IP only.
- **Source files:** You can add ISE Design Suite source files from an existing ISE Design Suite project to a new project in the Vivado Design Suite.

- **Run results:** Run results are not migrated. However, new run results are generated after implementing the design in the Vivado tools.
- **Constraints:** User constraint format (UCF) files used for the design must be converted to Xilinx design constraints (XDC) format for use with Vivado Design Suite. For information on migrating UCF constraints to XDC.

### ➤ **Launching the Vivado Design Suite**

You can launch the Vivado Design Suite and run the tools using different methods depending on your preference. For example, you can choose a Tcl script-based compilation style method in which you manage sources and the design process yourself, also known as *Non-Project Mode*. Alternatively, you can use a project-based method to automatically manage your design process and design data using projects and project states, also known as *Project Mode*. Either of these methods can be run using a Tcl scripted batch mode or run interactively in the Vivado IDE.

### ➤ **Working with Tcl**

If you prefer to work directly with Tcl, you can interact with your design using Tcl commands using either of the following methods:

- Enter individual Tcl commands in the Vivado Design Suite Tcl shell outside of the Vivado IDE.
- Enter individual Tcl commands in the Tcl Console at the bottom of the Vivado IDE.
- Run Tcl scripts from the Vivado Design Suite Tcl shell.
- Run Tcl scripts from the Vivado IDE.

### ➤ **Launching the Vivado Design Suite Tcl Shell:**

Use the following command to invoke the Vivado Design Suite Tcl shell either at the Linux command prompt or within a Windows Command Prompt window:

```
vivado -mode tcl.
```

On Windows, you can also select **Start > All Programs > Xilinx Design Tools > Vivado 2017.x > Vivado 2017.x Tcl Shell**.

### ➤ **Working with the Vivado IDE:**

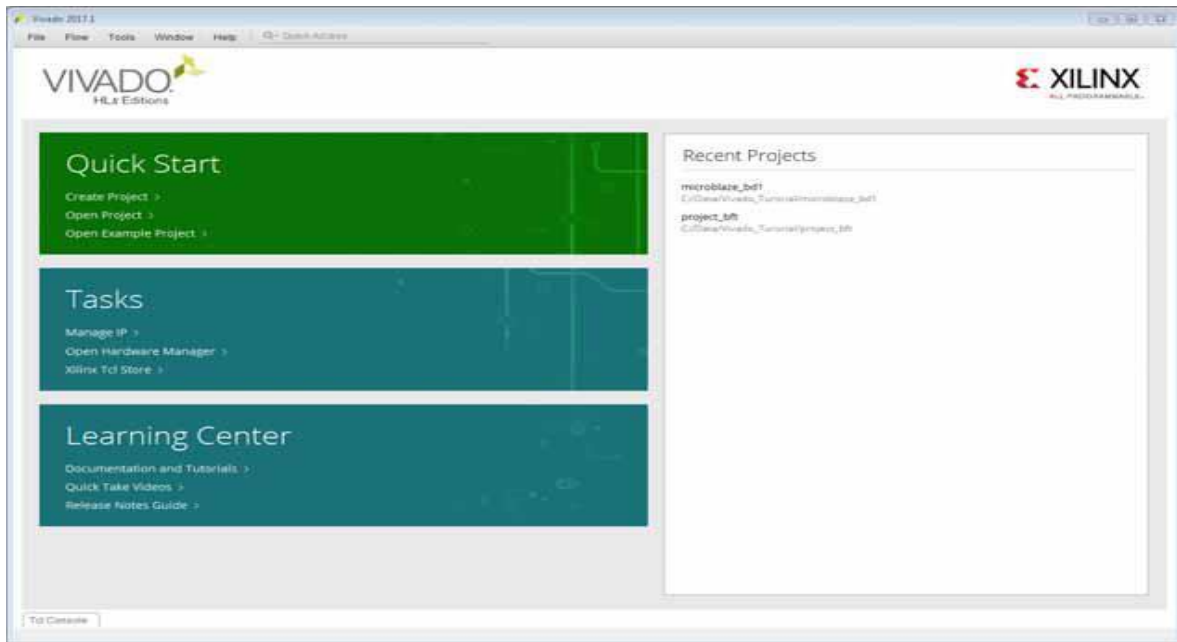
If you prefer to work in a GUI, you can launch the Vivado IDE from Windows or Linux.

Launching the Vivado IDE on Windows

Select **Start > All Programs > Xilinx Design Tools > Vivado 2017.x > Vivado 2017.x**.

## Using the Vivado IDE

When you launch the Vivado IDE, the Getting Started page displays and provides you with different options to help you begin working with the Vivado Design Suite.



**Fig.1.2 Vivado Getting started page**

### ➤ **STARTING WITH A PROJECT:**

You can create or open a project, and add source files to define your design. The Quick Start section of the Getting Started Page provides links for easy access to the following steps:

- Create a project using the New Project wizard.
- Open existing projects.
- Open example projects provided by Xilinx.

### ➤ **Opening the hardware manager:**

You can open the Vivado Design Suite Hardware Manager to program your design into a device. The Vivado logic analyzer and Vivado serial I/O analyzer features of the tool enable you to debug your design. For example, you can add ILA, VIO, Memory IP, and JTAG-to-AXI cores to your design for debugging in the Vivado logic analyzer, or use the IBERT example design from the Xilinx IP catalog to test and configure the GTs in your design with the Vivado serial I/O analyzer.

➤ **Accessing the Tcl Store:**

The Xilinx Tcl Store is an open source repository of Tcl code designed primarily for use in FPGA designs with the Vivado Design Suite. The Tcl Store provides access to multiple scripts and utilities contributed from different sources, which solve various issues and improve productivity. You can install Tcl scripts and also contribute Tcl scripts to share your expertise with others.

➤ **Documentation Navigator**

You can view the Xilinx® tool and hardware documentation in the Xilinx Documentation Navigator (DocNav) or on the Xilinx website. DocNav is integrated with the Vivado Design Suite. It provides an environment to access and manage the entire set of Xilinx documentation for hardware and software products, training, and support materials.

To open the Documentation Navigator:

- In the Vivado IDE, select any documentation link on the Getting Started page or in the Help menu.
- On Windows, select **Start > All Programs > Xilinx Design Tools > DocNav**.
- At the Linux command prompt, enter: docnav

➤ **Features of the Documentation Navigator include:**

- **Catalog:** Displays all available Xilinx software and hardware documents, Quick Take videos, Design Advisories, and Application Notes.
- **Filters:** Allows you to view documentation by specific document types, specific devices, or other relevant categories.
- **Search:** Enables you to find documentation based on the specified search terms. The search capability works for documentation both in the local repository and on the Xilinx website.
- **Design Hubs:** Provides quick access to documentation, training, and information for specific design tasks.
- **Ultra Fast Design Methodology Checklist:** Perform the Checklist on your design to ensure Xilinx recommended design practices are followed for the best user experience and design performance.
- **Quick Download:** Documentation Navigator manages downloading Xilinx documentation to your local desktop.
- **Web Search Results:** Allows you to search for documents on [Xilinx Support](#).

- **Send Feedback:** Allows you to send feedback on a Design Hub to Xilinx.
- **Documentation Update:** Documentation Navigator monitors and indicates when documents are updated on the Xilinx website.

#### ➤ **Vivado quick help:**

The Vivado Quick Help system is available from within the Vivado IDE by clicking on the **?** button in dialog boxes, windows, and wizards. A browser -like window opens with an overview of the feature, and the various inputs or settings that drive it. The Vivado Quick Help system also provides references to user guides, Quick Take videos, and other documentation for a specific feature.

The Quick Help browser window includes a search function for locating text within a specific help file. The browser has back and forward buttons for viewing the history of Quick Help windows viewed while working in the Vivado IDE.

#### ➤ **Documentation Suite:**

- **Vivado Design Suite User Guides:** These guides are categorized by design task for easy navigation to the information you need. User guides contain detailed information about running specific commands and performing specific design tasks within the Vivado Design Suite.
- **Reference Guides:** These guides provide reference information for topics, such as Tcl commands, constraints, and device libraries.
- **Methodology Guides:** These guides provide high-level guidance for performing specific design tasks, such as design migrating and large design guidance.

## **1.5 GOOGLE COLAB-PYTHON:**

#### ➤ **Introduction:**

Google is quite aggressive in AI research. Over many years, Google developed AI framework called Tensor Flow and a development tool called Collaboratory. Today Tensor Flow is open-sourced and since 2017, Google made Collaboratory free for public use. Collaboratory is now known as Google Colab or simply Colab. Another attractive feature that Google offers to the developers is the use of GPU. Colab supports GPU and it is totally free. The reasons for making it free for public could be to make its software a standard in the academics for teaching machine learning and data science. It may also have a long term perspective of building a customer base for Google Cloud APIs which are sold per-use basis.

Irrespective of the reasons, the introduction of Colab has eased the learning and development of machine learning applications.

Most importantly, it does not require a setup and the notebooks that you create can be simultaneously edited by team members - just the way you edit documents in Google Docs. Colab supports many popular machine learning libraries which can be easily loaded.

➤ **System:**

As a programmer the following functions has been performed using Google Colab.

- Write and execute code in Python.
- Document the code that supports mathematical equations.
- Create/Upload/Share notebooks.
- Import/Save notebooks from/to Google Drive.
- Import/Publish notebooks from GitHub.
- Import external datasets e.g. from Kaggle.
- Integrate PyTorch, TensorFlow, Keras, OpenCV.
- Free Cloud service with free GPU.

As Colab implicitly uses Google Drive for storing your notebooks, ensure that you are logged in to your Google Drive account before proceeding further. We need to follow following steps in order to execute program.

Step 1: Open the following URL in your browser: <https://colab.research.google.com>

Step 2: Click on the NEW PYTHON 3 NOTEBOOK link at the bottom of the screen.

There is a code window in which you would enter your Python code.

➤ **Setting Notebook Name**

To rename the notebook, click on this name and type in the desired name. We will call this notebook as My First Colab Notebook. So type in this name in the edit box and hit ENTER. The notebook will acquire the name that you have given now.

➤ **Entering Code:**

You will now enter a trivial Python code in the code window and execute it.

➤ **Execution code:**

To execute the code, click on the arrow on the left side of the code window. After a while, you will see the output underneath the code window.

➤ **Adding code cell:**

To add more code to your notebook, select the following menu options



➤ **Insert / Code Cell**

Alternatively, just hover the mouse at the bottom center of the Code cell. When the CODE and TEXT buttons appear, click on the CODE to add a new cell. A new code cell will be added underneath the current cell. To run the entire code in your notebook without an interruption, execute the following menu options:

Runtime / Reset and run all.

➤ **Changing cell order:**

When your notebook contains a large number of code cells, you may come across situations where you would like to change the order of execution of these cells. You can do so by selecting the cell that you want to move and clicking the UP CELL or DOWN. You may click the buttons multiple times to move the cell for more than a single position.

➤ **Deleting Cell:**

During the development of your project, you may have introduced a few now-unwanted cells in your notebook. You can remove such cells from your project easily with a single click. Click on the vertical-dotted icon at the top right corner of your code cell. Click on the Delete cell option and the current cell will be deleted.

As the code cell supports full Python syntax, we may use Python comments in the code window to describe code. However, many a time we need more than a simple text based comments to illustrate the ML algorithms. ML heavily uses mathematics and to explain those terms and equations to your readers you need an editor that supports LaTeX - a language for mathematical representations. Colab provides Text Cells for this purpose. Text Cells are formatted using markdown.

➤ **Saving to Google Drive :**

Colab allows you to save your work to your Google Drive. The action will create a copy of your notebook and save it to your drive.

➤ **Saving to GitHub:**

Save your work to your GitHub repository. File / Save a copy in GitHub.

➤ **Sharing notebook:**

To share the notebook that you have created with other co-developers, you may share copy that you have made in your Google Drive. To publish the notebook to general audience, you may share it from your GitHub repository. There is one more way to share your work and that is by clicking on the SHARE link at the top right hand corner of your Colab notebook. This will open the share box

We may enter the email IDs of people with whom you would like to share the current document. We can set the kind of access by selecting from the three options.

- Specified group of people.
- Colleagues in your organization.
- Anyone with the link.
- .All public on the web.

### ➤ **System Aliases**

You will see the list in the output window as shown below:

```
bzcmp@ less* systemd-ask-password*
bzdiff@ lessecho* systemd-escape*
bzegrep@ lessfile@ systemd-hwdb*
bzexe* lesskey* systemd-inhibit*
bzfgrep@ lesspipe* systemd-machine-id-setup*
bzgrep* ln* systemd-notify*
bzip2* login* systemd-sysusers*
bzip2recover* loginctl* systemd-tmpfiles*
bzless@ ls* systemd-tty-ask-password-agent*
bzmore* lsblk* tar*
cat* lsmod@ tempfile*
chgrp* mkdir* touch*
chmod* mknod* true*
chown* mktemp* udevadm*
cp* more* ulockmgr_server*
dash* mount* umount*
date* mountpoint* uname*
dd* mv* uncompress*
df* networkctl* vdir*
dir* nisdomainname@ wdctl*
dmesg* pidof@ which*
dnsdomainname@ ps* ypdomainname@
domainname@ pwd* zcat*
echo* rbash@ zcmp*
egrep* readlink* zdiff*
false* rm* zegrep*
fgrep* rmdir* zfgrep*
findmnt* run-parts* zforce*
fusermount* sed* zgrep*
grep* sh@ zless*
gunzip* sh.distrib@ zmore*
gzexe* sleep* znew*
gzip* stty*
hostname* su*
```

Execute any of these commands.

➤ **Mounting drive:**

First, you need to mount your Google Drive in Colab. Select the following menu options: Tools / Command palette

➤ **Listing Drive Contents**

You can list the contents of the drive using the ls command as follows:

```
!ls "/content/drive/My Drive/Colab Notebooks"
```

➤ **Running Python Code**

Now, let us say that you want to run a Python file called hello.py stored in your GoogleDrive. Type the following command in the Code cell:

```
!python3 "/content/drive/My Drive/Colab Notebooks/hello.py"
```

➤ **Graphical output:**

Note that the graphical output is shown in the output section of the Code cell. Likewise, you will be able to create and display several types of charts throughout your program code.

➤ **ML-libraries:**

- **Keras**

Keras, written in Python, runs on top of TensorFlow, CNTK, or Theano. It enables easy and fast prototyping of neural network applications. It supports both convolutional networks (CNN) and recurrent networks, and also their combinations. It seamlessly supports GPU.

To install Keras, use the following command:

```
!pip install -q keras
```

- **Open CV**

OpenCV is an open source computer vision library for developing machine learning applications. It has more than 2500 optimized algorithms which support several applications such as recognizing faces, identifying objects, tracking moving objects, stitching images, and so on. Giants like Google, Yahoo, Microsoft, Intel, IBM, Sony, Honda, Toyota use this library. This is highly suited for developing real-time vision applications.

To install OpenCV use the following command:

```
!apt-get -qq install -y libsm6 libxext6 && pip install -q -U opencv-python
```

- **XGBoost**

XGBoost is a distributed gradient boosting library that runs on major distributed environments such as Hadoop. It is highly efficient, flexible and portable. It implements ML algorithms under the Gradient Boosting framework. To install XGBoost, use the following command:

```
!pip install -q xgboost==0.4a30
```

- **GraphViz**

Graphviz is an open source software for graph visualizations. It is used for visualization in networking, bioinformatics, database design, and for that matter in many domains where a visual interface of the data is desired.

To install GraphViz, use the following command:

```
!apt-get -qq install -y graphviz && pip install -q pydot
```

By this time, you have learned to create Jupyter notebooks containing popular machine learning libraries. You are now ready to develop your machine learning models. This requires high processing power. Colab provides free GPU for your notebooks.

## 1.6. MOTIVATION

Natural images suffer from problems like pick up noise, relative motion of the camera and sensor noise. In order to remove these problems and to make the images suitable for human vision we use Image Enhancement. Image Enhancement can be done by many techniques. These are mainly Spatial domain techniques and Frequency domain techniques. Spatial domain refers to the image plane itself and is based on direct manipulation of pixels in an image. These techniques are based on gray level transformation function. Where as in frequency domain techniques, the image is first transferred into frequency domain i.e. Fourier transform of image is computed then the enhancement operations are performed on it and then inverse is performed to get resultant image.

In this we perform different special domain techniques in order to enhance the image. The different special domain techniques are Image Negative, Log Transform, Gamma correction, Contrast stretching, Histogram equalization and Median filtering. These techniques improve the image quality and also remove the noise present in the images. Median filtering mainly used for the removal of different noises present in the image. Contrast Stretching improves the contrast of the image by stretching the pixel values. Histogram is the measure that is used for the measure of contrast of the image where as histogram equalization improves the contrast of the image by effectively spreading out the most frequent intensity values. So the main intention here is to improve the quality of the image by removing noise or by some other techniques and make the images more suitable for human vision.

## **1.7. OBJECTIVES**

The objectives of this work are mainly to improve the image quality by performing different image enhancement techniques using different software's. They are as follows:

- To verify different image enhancement techniques by Matlab.
- To verify image enhancement techniques using Xilinx-vivado.
- To verify image enhancement techniques by using Python.
- To compare performance measure using MSE and PSNR.

## **1.8. PROJECT OVER VIEW**

Image processing, different levels of image processing are reviewed in this chapter. Various important aspects that motivated us and objectives of the proposed work are presented in this chapter.

Literature survey i.e. a brief introduction about image enhancement is discussed in chapter 2. Need for image enhancement Different image enhancement techniques, performance measures used are presented in chapter 3.

Results obtained and their detailed discussion is presented in chapter 4. Conclusions and future scope is given in chapter 5

## **CHAPTER 2**

# LITARATURE SURVAY

## 2.1. INTRODUCTION

Image enhancement involves the process of processing the image such a way that the resulting image is better than the original one. Another important goal of image enhancement is to process an image which is more suitable for image processing applications. It also used to bring out specific features of an image or to highlight certain characteristics of image. Image enhancement have many applications they are vision graphics, Remote sensing, Satellite imaging, Medical imaging etc.

## 2.2. IMAGE ENHNACEMENT TECHNIQUES

Image enhancement techniques can be broadly classified into two categories. They are

- Spatial domain techniques.
- Frequency domain techniques.

Spatial domain techniques are based on modifying the pixel values of an image. These methods directly operate on the pixel values. Frequency domain techniques are based on alteration of the Fourier transform on an image. There are various types of spatial domain techniques available and they are Image negative, logarithmic transform, Power law transform, Contrast stretching, Histogram equalization and median filtering. Some frequency domain techniques are Low pass filters, high pass filters, Pseudo color processing etc.

Image negative is produced by subtracting each pixel from the maximum intensity value. For gray scale image, light areas appear dark and vice versa. For color image, colors are replaced by their complementary colors. Thus, red areas appear cyan, green appears magenta, and blues appears yellow, and vice versa.

Log transformation means replacing each pixel value with its logarithm. For lower amplitude of input image the range of gray levels are expanded. For higher amplitude of input image the range of gray levels are compressed.

Gamma correction is also called as power law transform. Like log transformation, power law curves with  $\gamma < 1$  map a narrow range of dark input values into a wider range of output values, with the opposite being true for higher input values. Similarly, for  $\gamma > 1$ , we get the opposite result.

Contrast stretching as the name suggests is an image enhancement technique that tries to improve the contrast by stretching the intensity values of an image to fill the entire dynamic range. The transformation function used is always linear and monotonically increasing.

Histogram equalization is a method in image processing of contrast adjustment using the image's histogram. This method usually increases the global contrast of many images, especially when the usable data of the image is represented by close contrast values. Through this adjustment, the intensities can be better distributed on the histogram. This allows for areas of lower local contrast to gain a higher contrast. Histogram equalization accomplishes this by effectively spreading out the most frequent intensity values.

The Median Filter is a non-linear digital filtering technique, often used to remove noise from an image or signal. Such noise reduction is a typical pre-processing step to improve the results of later processing (for example, edge detection on an image). Median filtering is very widely used in digital image processing because, under certain conditions, it preserves edges while removing noise.

For enhancement of remote sensing images (Lee et al. (2013)) [1] on the basis of adaptive intensity transfer function and dominant brightness level analysis has been used. It divide the input image into four wavelet sub bands and split the LL sub band into low-, middle-, and high-intensity layers by analyzing the log-average luminance of the resultant layer. After that apply adaptive intensity transfer function and then implement contrast enhancement technique then combine the decomposed image by using image fusion method after that at last use inverse discrete wavelet transform method. Then the contrast enhanced image has ready as a result. By using the Discrete Wavelet transform and singular value Decomposition, Discrete Cosine Transform the Histogram equalization, Contrast Enhancement, Bi-histogram equalization; (Veena et al. (2013)) [2] has projected a new method for contrast enhancement based on the dominant Brightness and Adaptive transformation. Without changing original image quality the proposed techniques has an appropriate for enhancement of low contrast satellite image.

Histogram equalization (Srivastava et al. (2013)) [3] has one of the best method that is very effective method to process the digital contrast enhancement but has not been suitable for every image. Sometimes it shows not good outcomes. To overcome this problem it provides a new method to improve the image result. In this interact with histogram that reflects improved outcomes as compare to conservative one. On the basis of Absolute mean brightness error and peak Signal to Noise Ratio values. It has an appropriate for real time applications.

The major limitation of contrast enhancement algorithm (Cheng and Zhang (2012)) [4] has Over-Enhancement which could stimulate the loss of edges, alter the main texture, damage the fine details, and create the image appearance unnatural. It has no efficient reason for Over-enhancement until now. It provides a new technique for the recognition of Over-enhancement. The outcomes have shown that the projected technique can establish the Over-



enhancement areas perfectly and efficiently and give a quantitative method to assess the Over-enhancement levels fine.

(Ghimire and Lee (2011)) [5] Work has been focused on nonlinear colour image enhancement techniques. The purpose of image enhancement has to get better some features of an image to construct it visually good one. It shows the image enhancement has applied only on the V(luminance value) component of the HSV colour image and H & S component need no modification for enhancement because these components has not been changed. The V element enhanced in two steps. In first step the V element by using non linear transfer function has divided into smaller overlapped blocks and for every pixel within the block the luminance enhancement has accepted out. In the next step the contrast enhancement method has applies on it. At last the H and S element image and V element has converted back to RGB Image. The result shows that enhancement has one of the best methods for the image enhancement.

(Roomi and Prabhu et al. (2011)) [6] Has provided that for better visualization of low contrast images contrast enhancement method has been used. Histogram equalization used for Contrast enhancement. Histogram equalization has not suitable for consumer electronics product straightforwardly. It provides a new method of histogram equalization that tries to found foreground and background pixels of an image and apply bi-histogram equalization on them. Its outcomes shows that this algorithm preserves the original image as compare to other techniques.

Chauhan and Bhadoria (2011) [7] Histogram equalization has predictable technique for contrast enhancement. Histogram equalization has some limitations. Histogram equalization recovers the disparity of an image by altering the intensity level of the pixel based on the intensity of the original image. To overcome these problems apply brightness preserving weight clustering histogram equalization that protect image brightness and enhance visual effects of an image efficiently as compare to histogram equalization technique.

Josephus and Remya S (2011) [11] proved that for local content emphasis that the adaptive histogram equalization has the best and efficient algorithm. But sometimes has a problem of amplification and introduction of the speckle noise due to it information lost. To overcome this problem the multilayered contrast limited adaptive histogram equalization with frost filter that focused on application to medical images. In this on contrast limited adaptive histogram equalization the combination of frost and median filter both has been used. For the removal of speckle noise in images the technique of frost filter has been done. The work has been done on medical images such as mammogram, knee, and brain images.

Demirel et al. (2010) [8] provided a novel satellite image contrast enhancement method based on the discrete wavelet transform and singular value decomposition has been projected. In this method by using discrete wavelet transform divide the input image into the four frequencies subbands and estimates the singular value matrix of low-low subband image and then restructure improved by applying inverse discrete wavelet transform. The illustration results on the finishing image quality show the advantage of the projected technique over the

predictable and the state-of-the-art method. The different techniques used for example discrete wavelet transform, Image equalization and satellite image contrast enhancement. Compare the techniques with general histogram equalization and local histogram equalization.

Ke et al.(2010) [9] This provide there are so many types of image enhancement techniques that makes the image results better that associate to the person visual system. It includes the two techniques bilateral tone Adjustment and Saliency Weighted Contrast Enhancement both combined in image enhancement framework. The main scenes that are contained in mid-tone regions enhanced by bilateral tone adjustment in most of the curve-based global contrast enhancement techniques. The saliency-weighted Contrast enhancement integrates the notion of image saliency into an easy filter-based contrast enhancement technique. It performs extra enhancement in regions that persons give larger concentration to. By using the luminance component in this saliency weighted contrast enhancement achieves extra performance. It proved that to achieve higher contrast enhancement with slight sound and huge image quality.

Murahira et al. (2010) [10] proved for improving the disparity in digital images histogram equalization is one of the general technique. On the other hand, it will cause a consequence on the brightness saturation or shadow in several identical areas. To overcome these things mean preserving bi-histogram equalization technique has been developed. New histogram equalization with variable enhancement degree and bi-histogram equalization with variable degree has developed. By only one parameter the degree of every of these techniques has controlled. Every type of images is enhanced effectively.

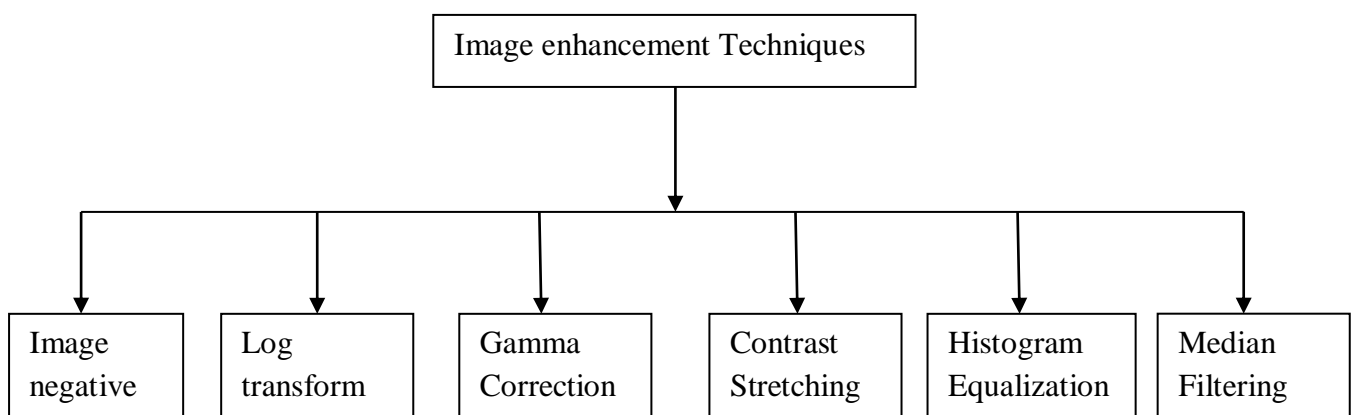
## **CHAPTER 3**

# IMAGE ENHANCEMENT TECHNIQUES

## 3.1 NEED FOR IMAGE ENHANCEMENT

- The main objective of image enhancement is to process a given image so that the result is more suitable than the original image for a specific application.
- It accentuates or sharpens image features such as edges, boundaries, or contrast to make a graphic display more helpful for display and analysis.
- To provide better input for other automated image processing techniques i.e. for Frequency domain methods; this operates in Fourier Transform of an image.
- Image enhancement is used to remove the noise from an image and also improves its quality.
- Image enhancement techniques help in improving the visibility of any portion or feature of the image suppressing the information in other portions or features.

## 3.2 DIFFERENT IMAGE ENHANCEMENT TECHNIQUES



**Fig.3.1 Different Image Enhancement Techniques**

### 3.2.1. IMAGE NEGATIVE:

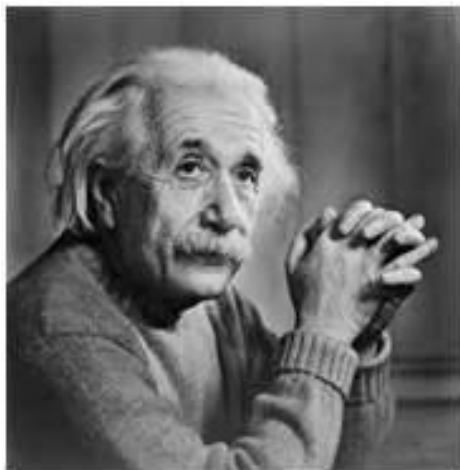
Image negative is produced by subtracting each pixel from the maximum intensity value. e.g. for an 8-bit image, the max intensity value is  $2^8 - 1 = 255$ , thus each pixel is subtracted from 255 to produce the output image.

Thus, the transformation function used in image negative is

$$S = T(r) = L - 1 - r.$$

Where  $L-1$  is the max intensity value and  $s$ , and  $r$  are the output and input pixel values respectively.

This expression results in reversing of the grey level intensities of the image there by produces a negative like image i.e. for gray scale image, light areas appear dark and vice versa. For color image, colors are replaced by their complementary colors. Thus, red areas appear cyan, green appears magenta, and blues appears yellow, and vice versa.



(a)



(b)

**Fig.3.2 Example for image negative, (a) Original image (b) Image negative**

### 3.2.1 LOG TRANSFORMATON:

Log transformation means replacing each pixel value with its logarithm. The general form of log transformation function is

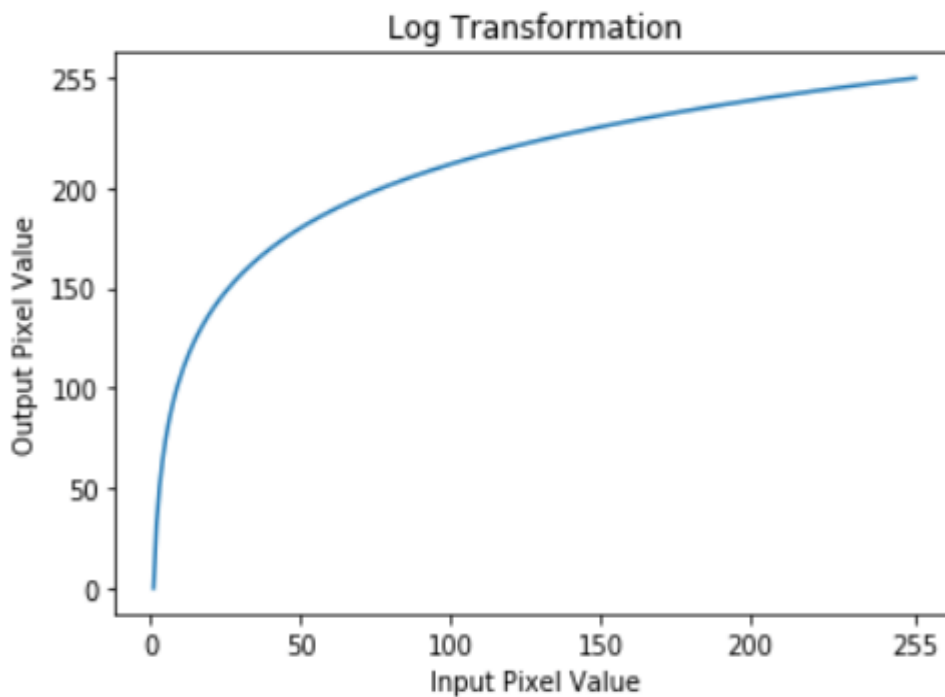
$$s = T(r) = c \cdot \log(1+r)$$

Where, 's' and 'r' are the output and input pixel values and c is the scaling constant represented by the following expression (for 8-bit)

$$c = 255/(\log(1 + \text{max\_input\_pixel\_value}))$$

The value of c is chosen such that we get the maximum output value corresponding to the bit size used. e.g. for 8 bit image, c is chosen such that we get max value equal to 255.

For an 8-bit image, log transformation looks like this



**Fig 3.3 Log transformation for 8-bit image**

Clearly, the low intensity values in the input image are mapped to a wider range of output levels. The opposite is true for the higher values.

➤ **PROPERTIES:**

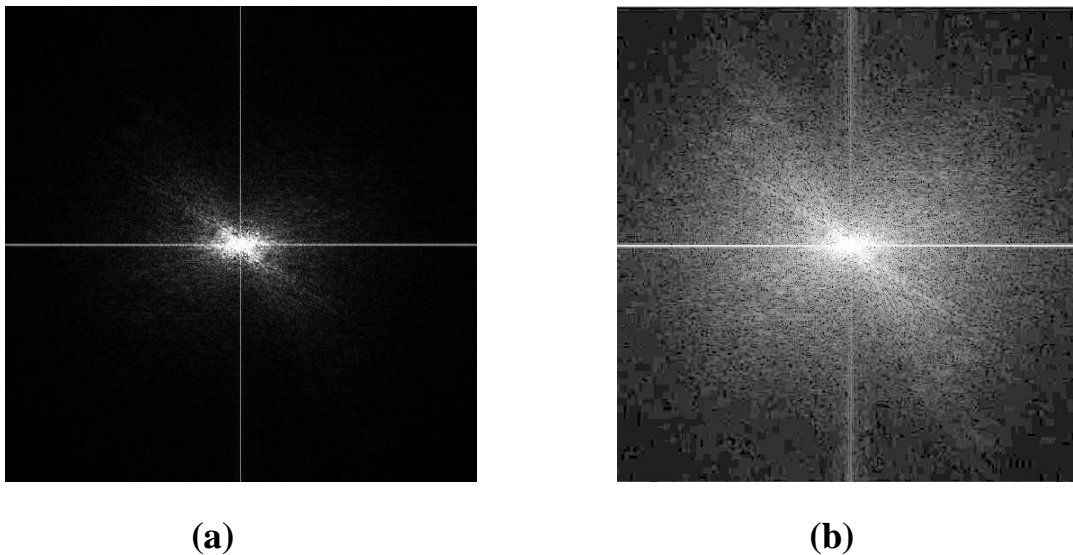
- For lower amplitude of input image the range of gray levels are expanded.
- For higher amplitude of input image the range of gray levels are compressed.

➤ **Applications:**

- Expands the dark pixels in the image while compressing the brighter pixels.
- Compresses the dynamic range (display of Fourier transform).

Dynamic range refers to the ratio of max and min intensity values. When the dynamic range of the image is greater than that of displaying device (like in Fourier transform), the lower values are suppressed. To overcome this issue, we use log transform. Log transformation first compresses the dynamic range and then up scales the image to a dynamic range of the display device.

Thus, a logarithmic transform is appropriate when we want to enhance the low pixel values at the expense of loss of information in the high pixel values.



**Fig 3.4 Example for Log Transform ,(a) original image,(b) image after log transform**

### 3.2.3 GAMMA CORRECTION:

Gamma correction is also called as power law transform. The general form of Power law (Gamma) transformation function is

$$s = c * r^\gamma$$

Where, 's' and 'r' are the output and input pixel values, respectively and 'c' and  $\gamma$  are the positive constants. Like log transformation, power law curves with  $\gamma < 1$  map a narrow range of dark input values into a wider range of output values, with the opposite being true for higher input values. Similarly, for  $\gamma > 1$ , we get the opposite result which is shown in the figure below

This is also known as gamma correction, gamma encoding or gamma compression.

The below curves are generated for r values normalized from 0 to 1. Then multiplied by the scaling constant c corresponding to the bit size used.

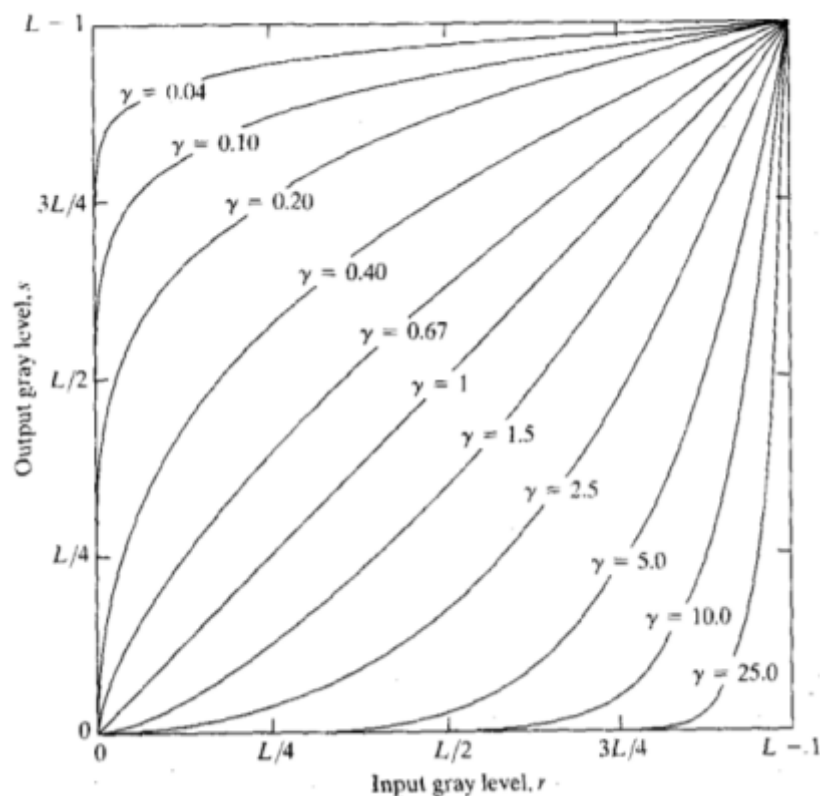


Fig3.5 Curve with different gamma values

This type of transformation is used for enhancing images for different type of display devices. The gamma of different display devices is different. For example Gamma of CRT lies in between of 1.8 to 2.5, that means the image displayed on CRT is dark.





(a)



(b)

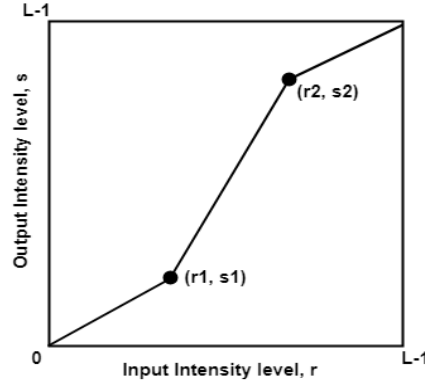


(c)

**Fig3.6 Example for gamma correction,(a) Original image, (b) image with  $\gamma < 1$ , (c) image with  $\gamma > 1$**

### **3.2.4.CONTRAST STRETCHING:**

Contrast stretching as the name suggests is an image enhancement technique that tries to improve the contrast by stretching the intensity values of an image to fill the entire dynamic range. The transformation function used is always linear and monotonically increasing.



**Fig: 3.7 shows a typical transformation function used for Contrast Stretching**

By changing the location of points  $(r_1, s_1)$  and  $(r_2, s_2)$ , we can control the shape of the transformation function. For example,

When  $r_1 = s_1$  and  $r_2 = s_2$ , transformation becomes a **Linear function**.

When  $r_1 = r_2$ ,  $s_1 = 0$  and  $s_2 = L-1$ , transformation becomes a **Thresholding function**.

When  $(r_1, s_1) = (r_{\min}, 0)$  and  $(r_2, s_2) = (r_{\max}, L-1)$ , this is known as **Min-Max Stretching**.

When  $(r_1, s_1) = (r_{\min} + c, 0)$  and  $(r_2, s_2) = (r_{\max} - c, L-1)$ , this is known as **Percentile Stretching**.

In **Min-Max Stretching**, the lower and upper values of the input image are made to span the full dynamic range. In other words, Lower value of the input image is mapped to 0 and the upper value is mapped to 255. All other intermediate values are reassigned new intensity values according to the following formulae

$$X_{\text{new}} = \frac{X_{\text{input}} - X_{\min}}{X_{\max} - X_{\min}} \times 255$$

Sometimes, when Min-Max is performed, the tail ends of the histogram becomes long resulting in no improvement in the image quality. So, it is better to clip a certain percentage like 1%, 2% of the data from the tail ends of the input image histogram. This is known as **Percentile Stretching**. The formulae is same as Min-Max but now the  $X_{\max}$  and  $X_{\min}$  are the clipped values.

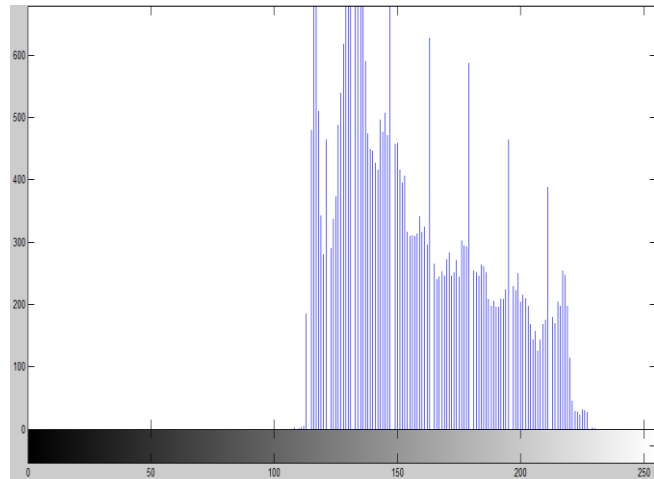
In order to perform contrast stretching operation on an image, the first step is to determine the limits over which image intensity values will be extended. These lower and upper limits will be called  $a$  and  $b$ , respectively (for standard 8-bit grayscale pictures, these limits are usually 0 and 255). Next, the histogram of the original image is examined to determine the value limits (lower =  $c$ , upper =  $d$ ) in the unmodified picture. If the original range covers the full possible set of values, straightforward contrast stretching will achieve nothing, but even then sometimes most of the image data is contained within a restricted

range; this restricted range can be stretched linearly, with original values which lie outside the range being set to the appropriate limit of the extended output range. Then for each pixel, the original value  $r$  is mapped to output value  $s$  using the function:

$$s = (r - c) \left( \frac{b - a}{d - c} \right) + a$$



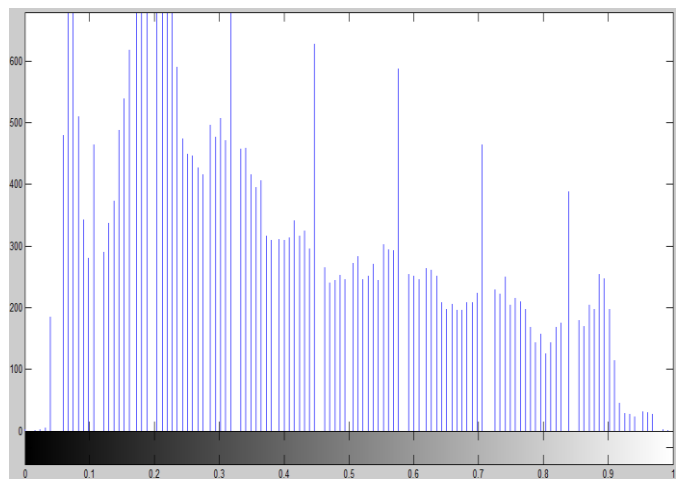
(a)



(b)



(c)



(d)

**Fig.3.8. Example for contrast stretching,(a) Original image,(b) histogram of original image,(c) Contrast Stretched image,(d) Histogram of contrast stretched image.**

### 3.2.5. HISTOGRAM EQUALISATION:

Histogram equalization is a method in image processing of contrast adjustment using the image's histogram. This method usually increases the global contrast of many images, especially when the usable data of the image is represented by close contrast values. Through this adjustment, the intensities can be better distributed on the histogram. This allows for areas of lower local contrast to gain a higher contrast. Histogram equalization accomplishes this by effectively spreading out the most frequent intensity values.

The method is useful in images with backgrounds and foregrounds that are both bright or both dark. In particular, the method can lead to better views of bone structure in x-ray images, and to better detail in photographs that are over or under-exposed. A key advantage of the method is that it is a fairly straight forward technique and an invertible operator. So in theory, if the histogram equalization function is known, then the original histogram can be recovered. The calculation is not computationally intensive. A disadvantage of the method is that it is indiscriminate. It may increase the contrast of background noise, while decreasing the usable signal.

In scientific imaging where spatial correlation is more important than intensity of signal (such as separating DNA fragments of quantized length), the small signal to noise ratio usually hampers visual detection.

Histogram equalization often produces unrealistic effects in photographs; however it is very useful for scientific images like thermal, satellite or x-ray images, often the same class of images to which one would apply false-color. Also histogram equalization can produce undesirable effects (like visible image gradient) when applied to images with low color depth. For example, if applied to 8-bit image displayed with 8-bit gray-scale palette it will further reduce color depth (number of unique shades of gray) of the image. Histogram equalization will work the best when applied to images with much higher color depth than palette size, like continuous data or 16 bit gray-scale images.

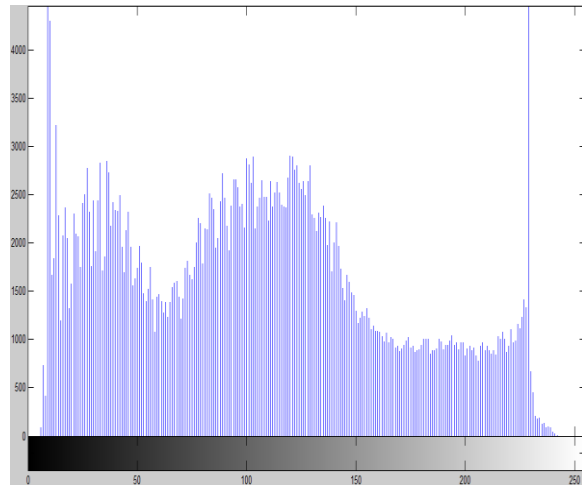
There are two ways to think about and implement histogram equalization, either as image change or as palette change. The operation can be expressed as  $P(M(I))$  where  $I$  is the original image,  $M$  is histogram equalization mapping operation and  $P$  is a palette. If we define a new palette as  $P'=P(M)$  and leave image  $I$  unchanged then histogram equalization is implemented as palette change. On the other hand if palette  $P$  remains unchanged and image is modified to  $I'=M(I)$  then the implementation is by image change. In most cases palette change is better as it preserves the original data.

Modifications of this method use multiple histograms, called sub-histograms, to emphasize local contrast, rather than overall contrast. Examples of such methods include adaptive histogram equalization, contrast limiting adaptive histogram equalization or CLAHE, multi peak histogram equalization (MPHE), and multipurpose beta optimized bi-histogram equalization (MBOBHE). The goal of these methods, especially MBOBHE, is to improve the contrast without producing brightness mean-shift and detail loss artifacts by modifying the HE algorithm.

A signal transform equivalent to histogram equalization also seems to happen in biological neural networks so as to maximize the output firing rate of the neuron as a function of the input statistics. This has been proved in particular in the fly retina.



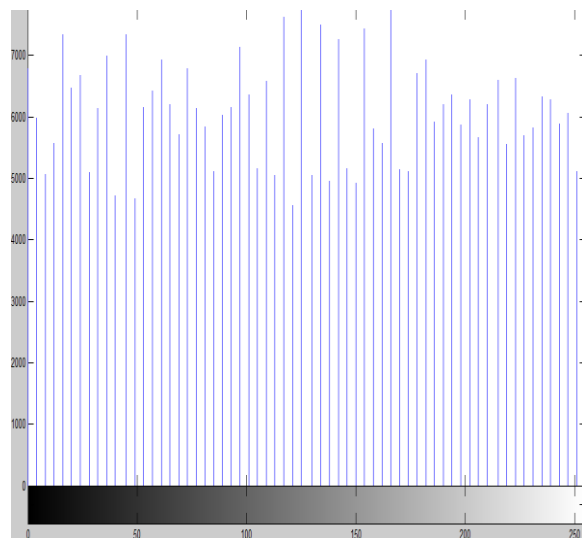
(a)



(b)



(c)



(d)

**Fig3.9 Example for histogram equalization,(a) Original image (b) histogram of original image (c) Equalized image (d) Histogram of equalized image.**

## ➤ APPLICATIONS:

- A popular tool for real-time processing: Histograms are simple to calculate in software and also lend themselves to economic hardware implementations
- Histograms are used to analyze image: We can predict the properties of an image just by looking at the details of the histogram.
- Histograms are used for brightness purpose: We can adjust the brightness of an image by having the details of its histogram.
- Histograms are used to adjust the contrast of an image: The contrast of an image is adjusted accordingly required need by having the details of x-axis or gray level intensities of a histogram.
- Histograms are used for image equalization: The gray level intensities are expanded along the x-axis to produce a high contrast image.
- Histograms are also used in thresholding.
- Histograms improve the visual appearance of an image.
- Histogram of an image depicts the problems that originate during image acquisition such as dynamic range of pixels, contrast, etc.
- Histograms reflect a wide range of vulnerabilities such as saturation, spikes, and gaps, the impact of image compression.

### 3.2.6.MEDIAN FILTERING:

The Median Filter is a non-linear digital filtering technique, often used to remove noise from an image or signal. Such noise reduction is a typical pre-processing step to improve the results of later processing (for example, edge detection on an image). Median filtering is very widely used in digital image processing because, under certain conditions, it preserves edges while removing noise.

Median filtering is one kind of smoothing technique, as is linear Gaussian filtering. All smoothing techniques are effective at removing noise in smooth patches or smooth regions of a signal, but adversely affect edges. Often though, at the same time as reducing the noise in a signal, it is important to preserve the edges. Edges are of critical importance to the visual appearance of images, for example. For small to moderate levels of Gaussian noise, the median filter is demonstrably better than Gaussian blur at removing noise whilst preserving edges for a given, fixed window size. However, its performance is not that much better than

Gaussian blur for high levels of noise, whereas, for speckle noise and salt-and-pepper noise (impulsive noise), it is particularly effective. Because of this, median filtering is very widely used in digital image processing.

➤ **Median filtering algorithm:**

- Read the image from left to right, top to bottom pixel by pixel.
- Initiate a 3 x 3 mask (neighborhood windows), starting from the pixel, whose value is going to change after filtering.
- Extract all 3 x 3 mask elements and put into the 1-D element array.
- Sort the 1-D element array in ascending order.
- Extract the middle value of sorted element array Replace the current pixel value in the image with the medium pixel value in the 1-D element array, Move to the next pixel.
- Repeat steps 3 to 6 until end of the image.

➤ **Boundary issues:**

- Avoid processing the boundaries, with or without cropping the signal or image boundary afterwards, entries from other places in the signal.
- With images for example, entries from the far horizontal or vertical boundary might be selected.
- Shrinking the window near the boundaries, so that every window is full.

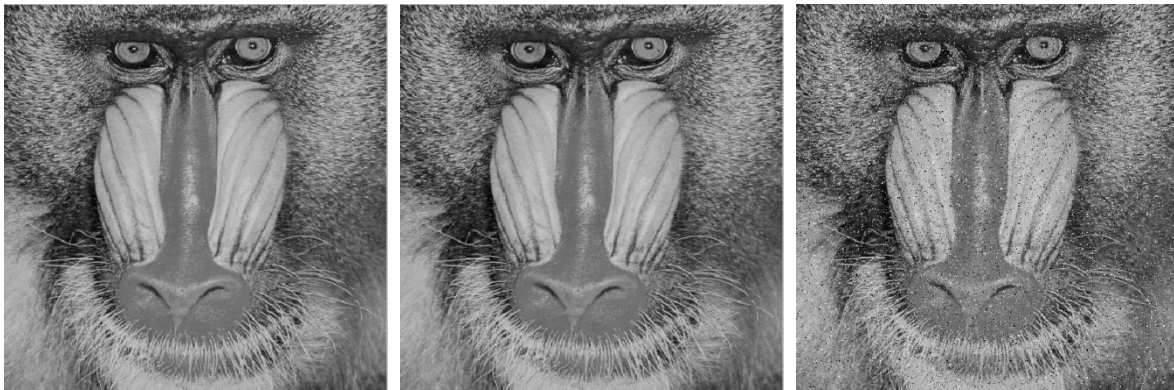
➤ **MSE and PSNR in median filtering:**

The mean-square error (MSE) and the peak signal-to-noise ratio (PSNR) are used to compare image compression quality. The MSE represents the cumulative squared error between the compressed and the original image, whereas PSNR represents a measure of the peak error. The lower the value of PSNR, the lower the error.

The experimental values of MSE and PSNR are calculated by adding different noises to the input image. From the results it is clear that median filtering can remove salt and pepper noise effectively as it has the lowest PSNR value.

➤ **Drawbacks of Median filtering:**

- Common drawbacks of the median filtering are it destroys small features in the image and computational cost.
- Computing a two-dimensional median for a  $N \times N$  window, requires sorting of  $N \times N$  elements for every image pixel. Therefore, using median filtering in any real-time vision system requires a significant computational power.



(a)

(b)

(c)

**Fig.3.10 Example for median filtering, (a) Original image,(b) Image with noise,(c) Image after median filtering.**

### **3.3. PERFORMANCE MEASURED USED:**

The performance measures used are MSE and PSNR.

MSE is nothing but mean square error, it is defined as the average squared difference between the estimated values and the actual value. MSE can be calculated by using formula

$$\text{MSE} = \frac{1}{N} \sum_{i=1}^N (y_1 - y_2)^2$$

#### **Properties of MSE:**

- Non-negativity:  $d_p(x, y) \geq 0$
- Identity:  $d_p(x, y) = 0$  if and only if  $x = y$ .
- Symmetry :  $d_p(x, y) = d_p(y, x)$
- Triangular inequality :  $d_p(x, z) \leq d_p(x, y) + d_p(y, z)$



PSNR is nothing but peak signal to noise ratio, it is defined as the ratio between the maximum possible power of a [signal](#) and the power of corrupting [noise](#) that affects the fidelity of its representation. PSNR can be calculated by using the formula

$$\text{PSNR} = \frac{1}{mn} \sum \sum [I(i,j) - k(i,j)]^2$$

By using these to we can measure the performance if the psnr value is high then the errors in that is very less and vice versa

### 3.4 IMAGE ENHANCEMENT USING MATLAB:

#### Image processing toolbox:

The basic data structure in MATLAB is the array, an ordered set of real or complex elements. This object is naturally suited to the representation of images, real-valued ordered sets of color or intensity data. MATLAB stores most images as two-dimensional arrays (i.e., matrices), in which each element of the matrix corresponds to a single pixel in the displayed image. (Pixel is derived from picture element and usually denotes a single dot on a computer display.) For example, an image composed of 200 rows and 300 columns of different colored dots would be stored in MATLAB as a 200-by-300 matrix. Some images, such as RGB, require a three-dimensional array, where the first plane in the third dimension represents the red pixel intensities, the second plane represents the green pixel intensities, and the third plane represents the blue pixel intensities. This convention makes working with images in MATLAB similar to working with any other type of matrix data, and makes the full power of MATLAB available for image processing applications. For example, you can select a single pixel from an image matrix using normal matrix subscripting. `I(2,15)` This command returns the value of the pixel at row 2, column 15 of the image I. Storage Classes in the Toolbox. By default, MATLAB stores most data in arrays of class double. The data in these arrays is stored as double precision (64-bit) floating-point numbers. All of MATLAB's functions and capabilities work with these arrays. For image processing, however, this data representation is not always ideal. The number of pixels in an image may be very large; for example, a 1000-by-1000 image has a million pixels. Since each pixel is represented by at least one array element, this image would require about 8 megabytes of memory. To reduce memory requirements, MATLAB supports storing image data in arrays of as 8-bit or 16-bit unsigned integers, class `uint8` and `uint16`. These arrays require one eighth or one fourth as much memory as double arrays. . The toolbox supports a wide range of image processing operations, including:

- Geometric operations
- Neighborhood and block operations
- Linear filtering and filter design
- Transforms

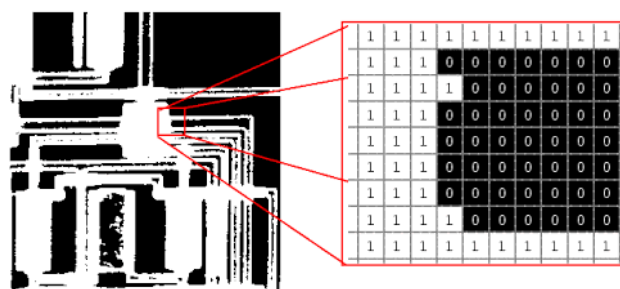
- Image analysis and enhancement
- Binary image operations
- Region of interest operations

MATLAB can import/export several image formats:

- BMP (Microsoft Windows Bitmap)
- GIF (Graphics Interchange Files)
- HDF (Hierarchical Data Format)
- JPEG (Joint Photographic Experts Group)
- PCX (Paintbrush)
- PNG (Portable Network Graphics)
- TIFF (Tagged Image File Format)
- XWD (X Window Dump)
- raw-data and other types of image data

Data types in MATLAB:

- Double (64-bit double-precision floating point)
- Single (32-bit single-precision floating point)
- Int32 (32-bit signed integer)
- Int16 (16-bit signed integer)
- Int8 (8-bit signed integer)
- Uint32 (32-bit unsigned integer)
- Uint16 (16-bit unsigned integer)
- Uint8 (8-bit unsigned integer)



**fig 3.11 Example of binary image**

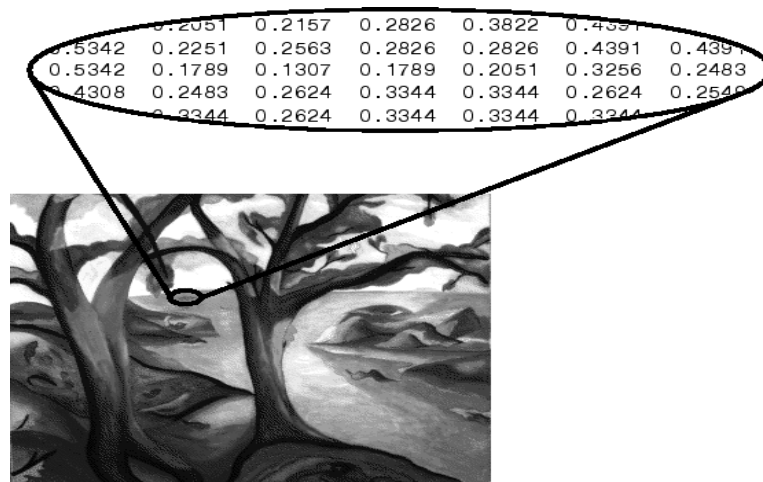


fig 3.12 example of intensity image

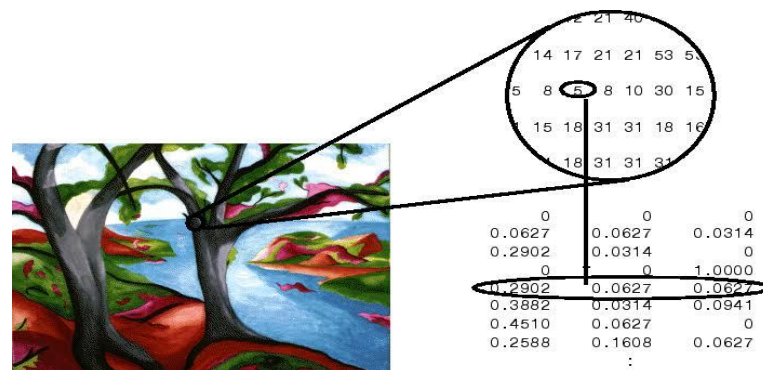


Fig 3.13 example of RGB image

### Image Import and Export:

In MATLAB, the primary way to display images is by using the image function. This function creates a Handle Graphics image object, and it includes syntax for setting the various properties of the object. MATLAB also includes the imagesc function, which is similar to image but which automatically scales the input data. The Image Processing Toolbox includes an additional display routine called imshow. Like image and imagesc, this function creates a Handle Graphics image object. However, imshow also automatically sets various Handle Graphics properties and attributes of the image to optimize the display. This section discusses displaying images using imshow. In general, using imshow for image processing applications is preferable to using image and imagesc. It is easier to use and in most cases, displays an image using one image pixel per screen pixel.

## Read and write images in matlab

- `img = imread('apple.jpg');`
- `dim = size(img);`
- `figure;`
- `imshow(img);`
- `imwrite(img, 'output.bmp', 'bmp');`

### Alternatives to imshow

- `imagesc(I)`
- `imtool(I)`
- `image(I)`

### Image Display in MATLAB:

- `image` - create and display image object
- `imagesc` - scale and display as image
- `imshow` - display image
- `colorbar` - display colorbar
- `getimage` - get image data from axes
- `trueSize` - adjust display size of image
- `zoom` - zoom in and zoom out of 2D plot

### Image Conversion in MATLAB:

- `gray2ind` - intensity image to index image
- `im2bw` - image to binary
- `im2double` - image to double precision
- `im2uint8` - image to 8-bit unsigned integers
- `im2uint16` - image to 16-bit unsigned integers
- `ind2gray` - indexed image to intensity image
- `mat2gray` - matrix to intensity image
- `rgb2gray` - RGB image to grayscale
- `rgb2ind` - RGB image to indexed image

### Image Operations in MATLAB:

- RGB image to gray image
- Image resize
- Image crop
- Image rotate
- Image histogram
- Image histogram equalization
- Image DCT/IDCT
- Convolution

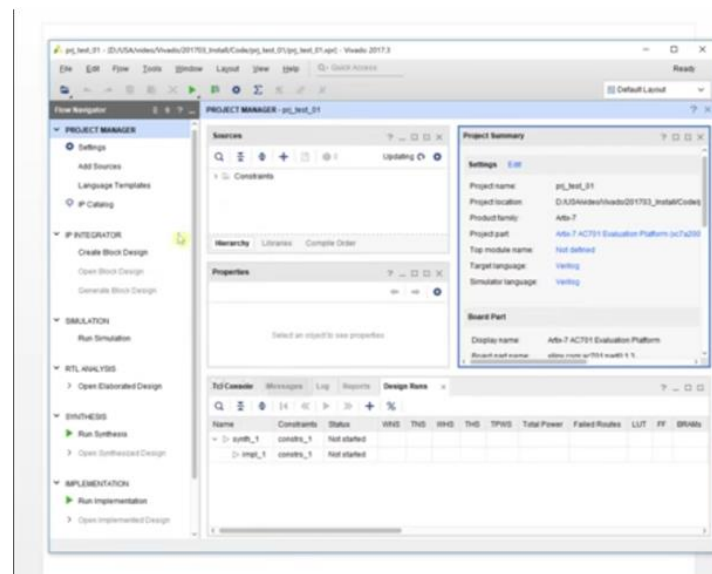
### 3.5 IMAGE ENHANCEMENT USING XLINIX - VIVADO:

In xilinx- vivado we cannot upload the image directly so we give binary values as input. The binary values are obtained by converting the input image pixel values into binary values using matlab . After simulating in the xilinx, we will get the output in the form of binary code. The binary code obtained from xilinx should match the output of the matlab binary code.

We can create or open a project, and add source files to define your design. The Quick Start section of the Getting Started Page provides links for easy access to the following steps:

- Create a project using the New Project wizard.
- Open existing projects.
- Open example projects provided by Xilinx.

After creating a new project we will get a window in which we have to add sources and have to select the language as verilog. Now a default module is displayed in which we have to add no. of inputs and outputs, then a window is displayed as shown in below fig.



**Fig3.14 Xilinx-vivado project navigator**

In project manager our project will be present. If we open that we will get a editor window in which we can write our code. After that in the file navigator we have run the synthesis in order check the errors, if we have any errors it will show after the completion of synthesis. If the synthesis is done perfectly then in the same navigator click on the run simulation then we will get the output in the form of wave forms.

### 3.6 IMAGE ENHANCEMENT USING GOOGLE-COLAB PYTHON:

Google is quite aggressive in AI research. Over many years, Google developed AI framework called Tensor Flow and a development tool called Collaboratory. Today Tensor Flow is open-sourced and since 2017, Google made Collaboratory free for public use. Collaboratory is now known as Google Colab or simply Colab. Another attractive feature that Google offers to the developers is the use of GPU. Colab supports GPU and it is totally free. The reasons for making it free for public could be to make its software a standard in the academics for teaching machine learning and data science. It may also have a long term perspective of building a customer base for Google Cloud APIs which are sold per-use basis. Irrespective of the reasons, the introduction of Colab has eased the learning and development of machine learning applications.

Most importantly, it does not require a setup and the notebooks that you create can be simultaneously edited by team members - just the way you edit documents in Google Docs. Colab supports many popular machine learning libraries which can be easily loaded.

#### Steps involved:

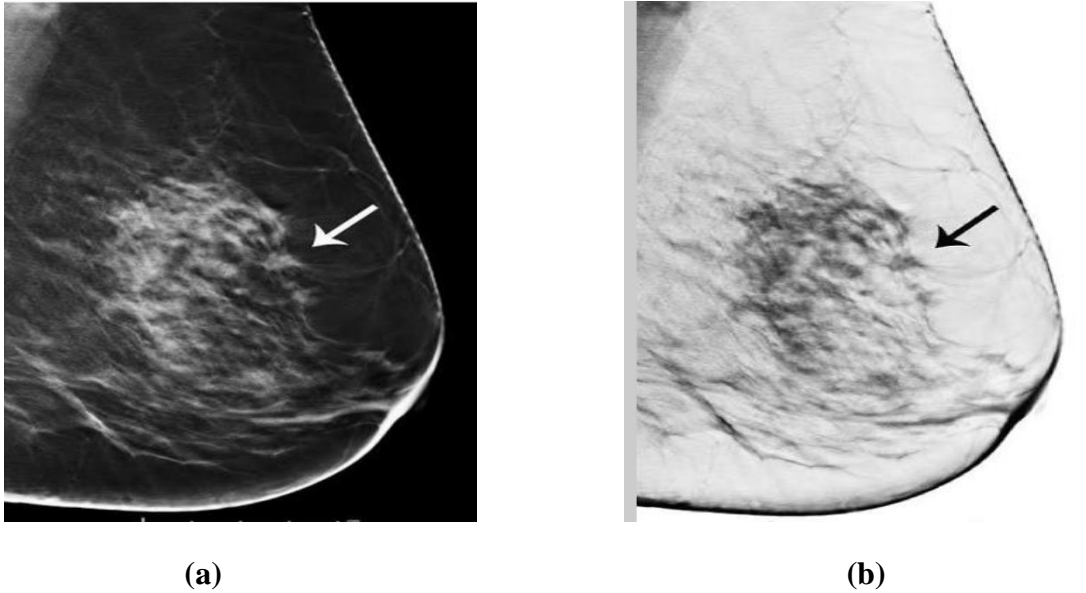
- Upload images to Google drive
- Open Google colab  
<https://colab.research.google.com/>
- Connect Google Drive to Google colab
- Create a New Note-book in Google colab
- Select GPU or CPU for processing
- Write code in cell
- Execute cell on clicking run all option, then we will get the output if there are no errors

# **CHAPTER 4**

## RESULTS AND DISSCUSSIONS

After performing different enhancement techniques using different softwares the obtained results are as follows

### RESULTS OBTAINED BY USING MATLAB:



**Fig 4.1.Result of image negative, (a) original image, (b) negative of the image**

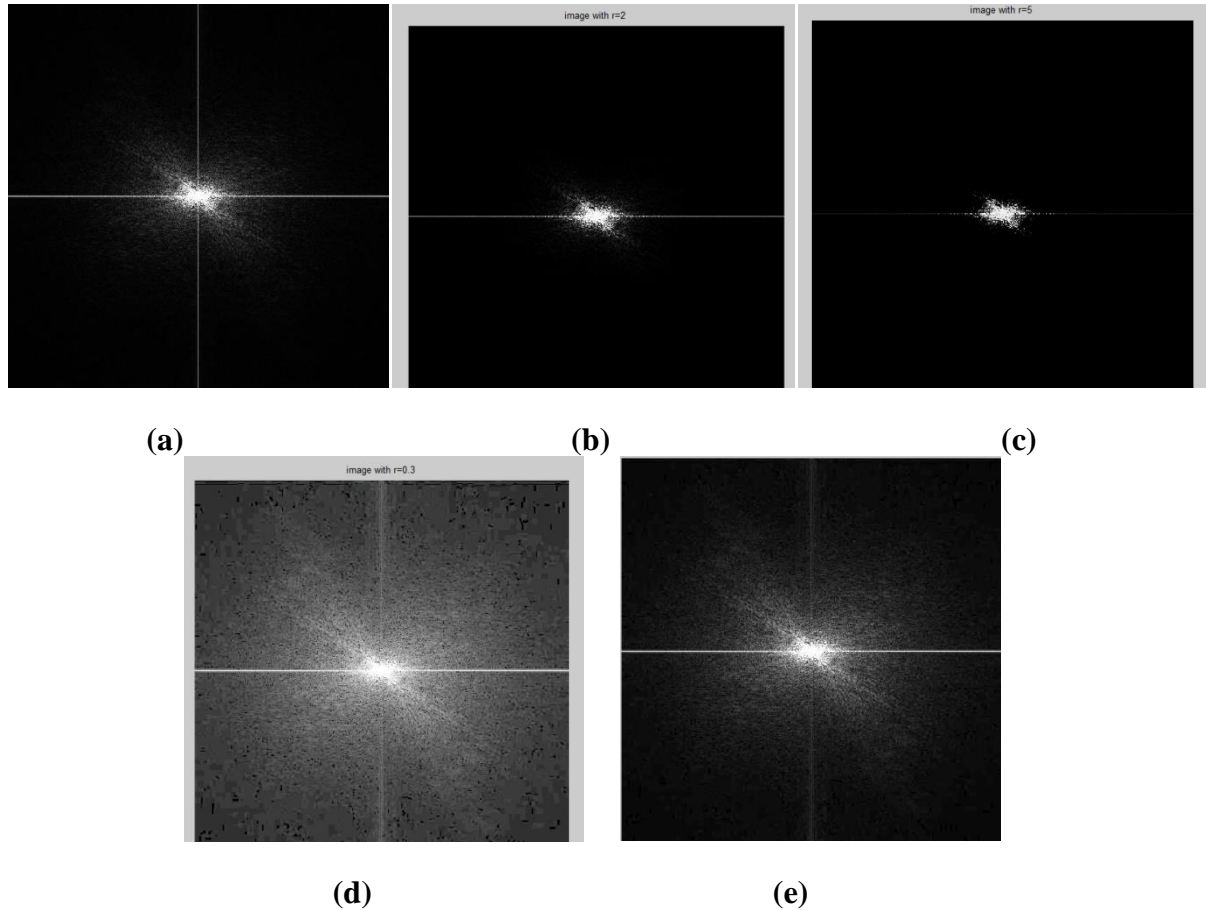
From the above results we can observe that the dark part of the image became bright and the bright part of the image became dark. So we can get better information from output than the input.



**Fig 4.2 Results obtained by Log transform, (a) Original image, (b) Image after log transform.**

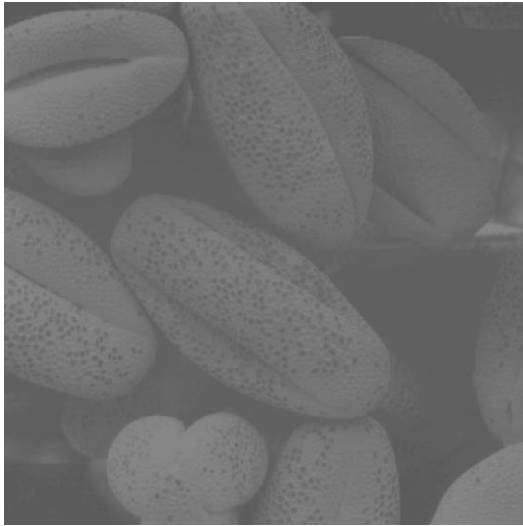


From the above result we can observe that the dark pixels in the image original are expanded as compared to the brighter pixels and the output image became bright when compare to original image.

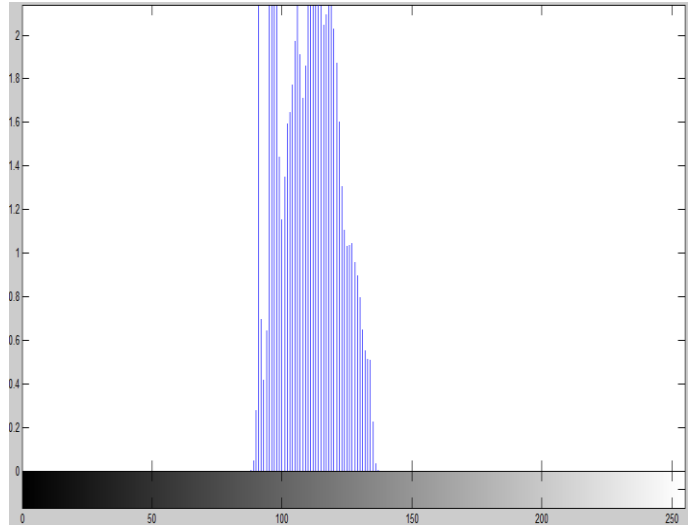


**Fig 4.3 Results obtained by gamma correction or power law transform, (a) Original image, (b) image with gamma=5, (c) image with gamma=2, (d) image with gamma=0.3, (e) image with gamma=0.6**

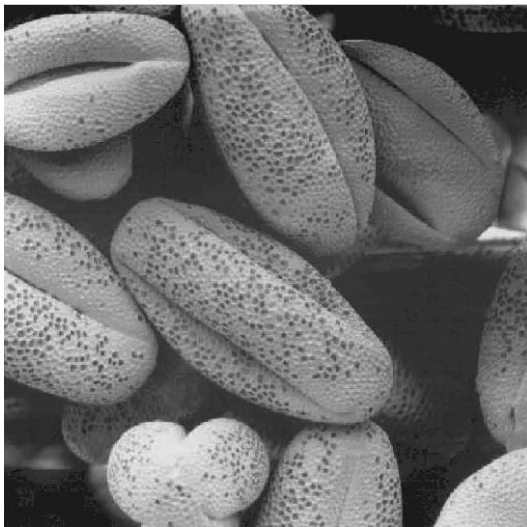
From the above result we can observe that for different gamma values the contrast of the image changing if we compare the original image with the images (b) and (c) they became dark when compared to the original because the gamma value we applied is greater than 1. Whereas when we compare the original image with (d) and (e) they became bright as their gamma value is less than 1.



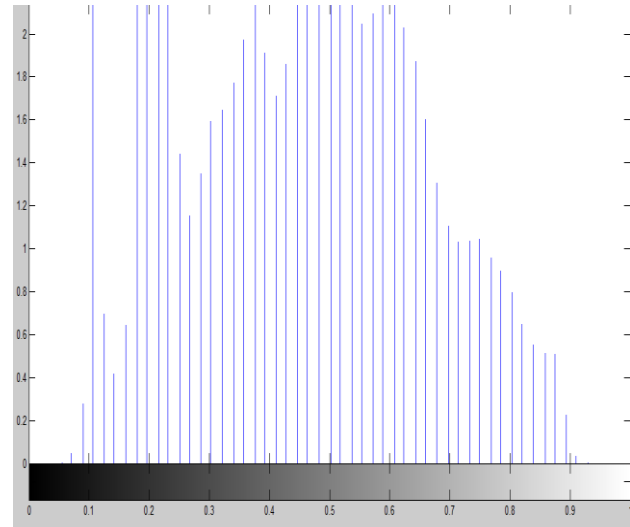
(a)



(b)



(c)



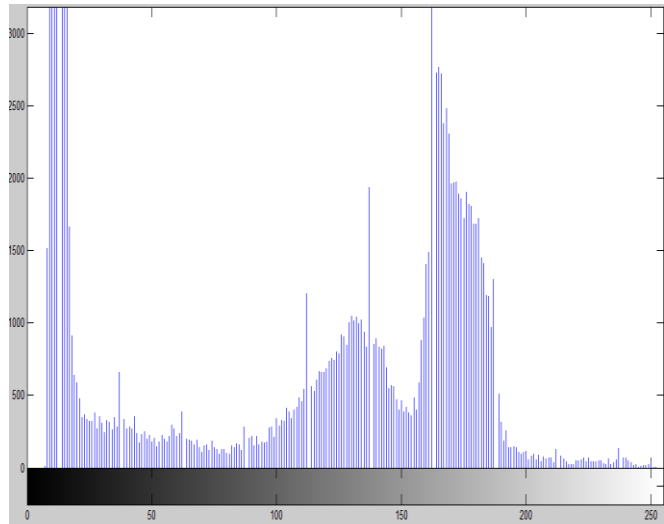
(d)

**Fig 4.4 Results obtained for contrast stretching, (a) Original image, (b) Histogram of original image, (c) Contrast stretched image, (d) Histogram of contrast stretched image.**

From the above results we can observe that the low contrast image is converted into a good contrast image by stretching the pixel values of the low contrast image. We can observe the improvement of its contrast by its histogram. If we observe the histogram of the original image all the pixels are concentrated at the center so it is termed as very dark image, whereas the histogram of output image is distributed i.e. pixel values are at stretched. Hence we can say the output image is a good contrast image.



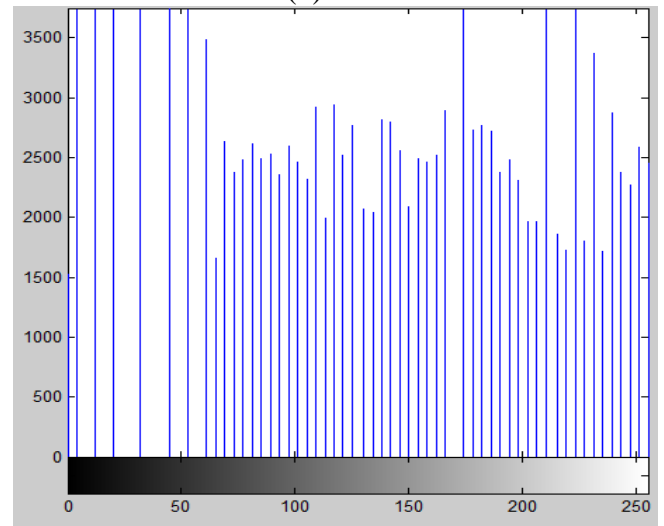
(a)



(b)



(c)



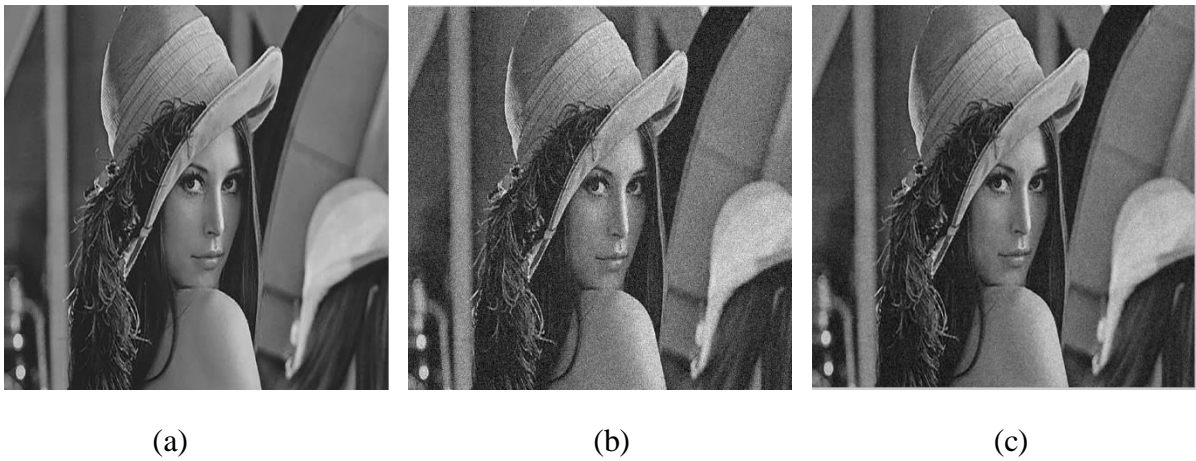
(d)

**Fig 4.5 Results of histogram equalization, (a) original image (b) histogram of original image, (c) equalized image (d) histogram of equalized image.**

From the above results we observe that histogram equalization also increases the contrast of the image by spreading the most frequent intensity values. When we compare the images (b) and (d) we observe that all the pixel values are spread by increasing the image contrast.



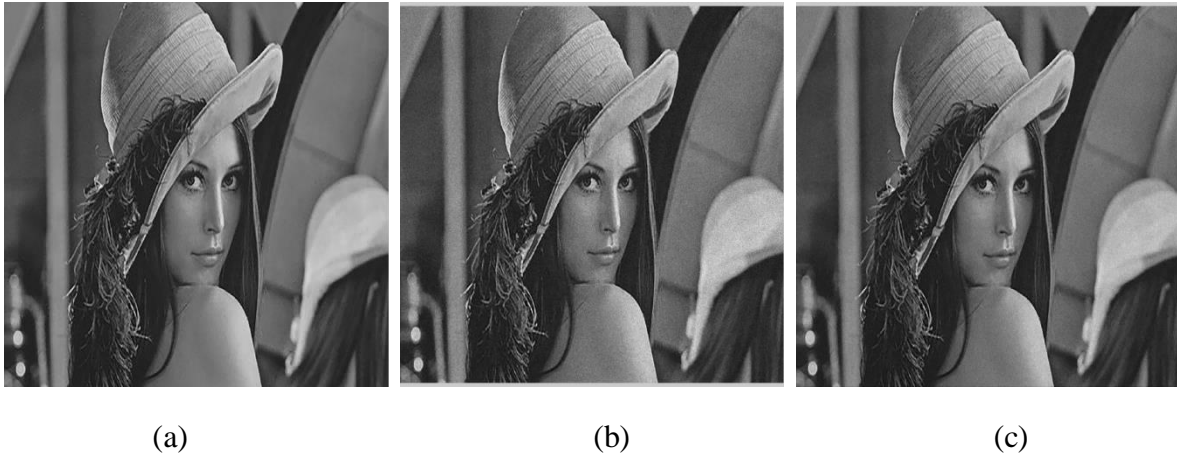
**Fig 4.6 Result of median filtering, (a) Original image (b) image with salt and pepper noise (c) image after median filtering.**



**Fig 4.7 Result of median filtering, (a) Original image (b) image with salt Gaussian noise (c) image after median filtering.**



**Fig 4.8 Result of median filtering, (a) Original image (b) image with salt speckle noise (c) image after median filtering.**



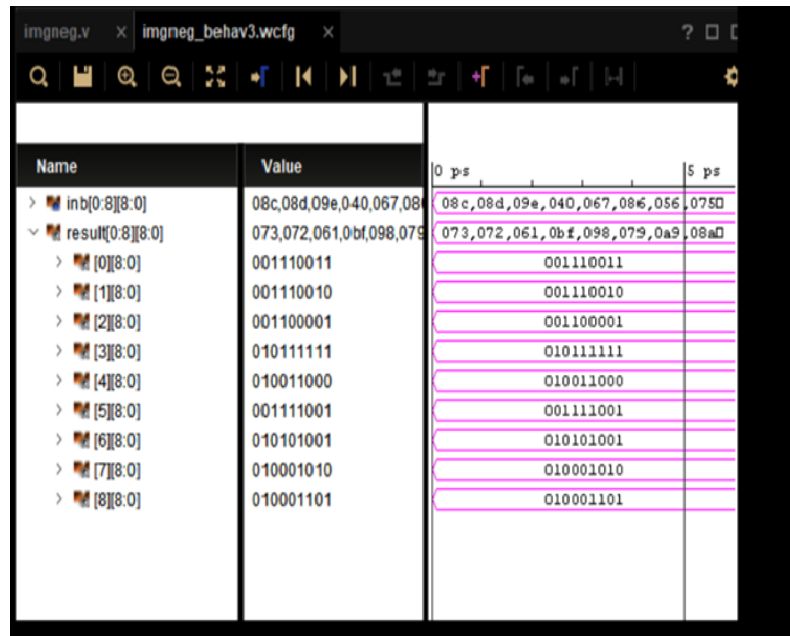
**Fig 4.9 Result of median filtering, (a) Original image (b) image with salt Poisson noise (c) image after median filtering.**

From the above results we can observe that median filtering is used for the removal of different noises. When we observe image (a) in all the figures are original images and image (b) are images added with different noises. In order to remove these noises we apply median filtering and results obtained are represented in image (c). Median filter perfectly removes salt and pepper noise when compared to other noises. We can conclude this by observing the PSNR and MSE values. We get more PSNR value for median filtering of salt and pepper noise.

S.NO	DIFFERENT NOISES	PSNR VALUE	MSE VALUE
1	Salt and pepper noise	40.6186	5.6392
2	Gaussian noise	27.9653	103.8846
3	Speckle noise	30.5311	57.5395
4	Poisson noise	30.8784	53.1183

**Table 4.1 comparison of PSNR and MSE values obtained by different noises.**

## RESULTS OBTAINED BY XILINX-VIVADO:



Name	Value	0 ps	5 ps
inb[0:8][8:0]	08c,08d,09e,040,067,08e,073,072,061,0bf,098,079	08c,08d,09e,040,067,08e,073,072,061,0bf,098,079,0a9,08a	
result[0:8][8:0]			
> [0][8:0]	001110011	001110011	
> [1][8:0]	001110010	001110010	
> [2][8:0]	001100001	001100001	
> [3][8:0]	010111111	010111111	
> [4][8:0]	010011000	010011000	
> [5][8:0]	001111001	001111001	
> [6][8:0]	010101001	010101001	
> [7][8:0]	010001010	010001010	
> [8][8:0]	010001101	010001101	

Fig 4.10 Result of image negative

## RESULTS OBTAINED BY USING PYTHON

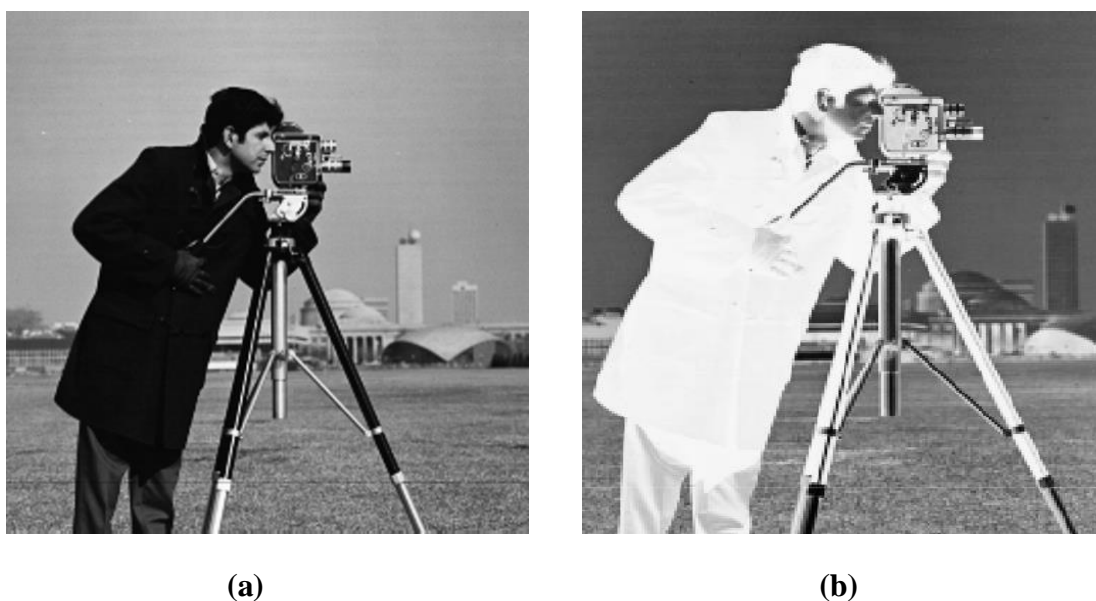
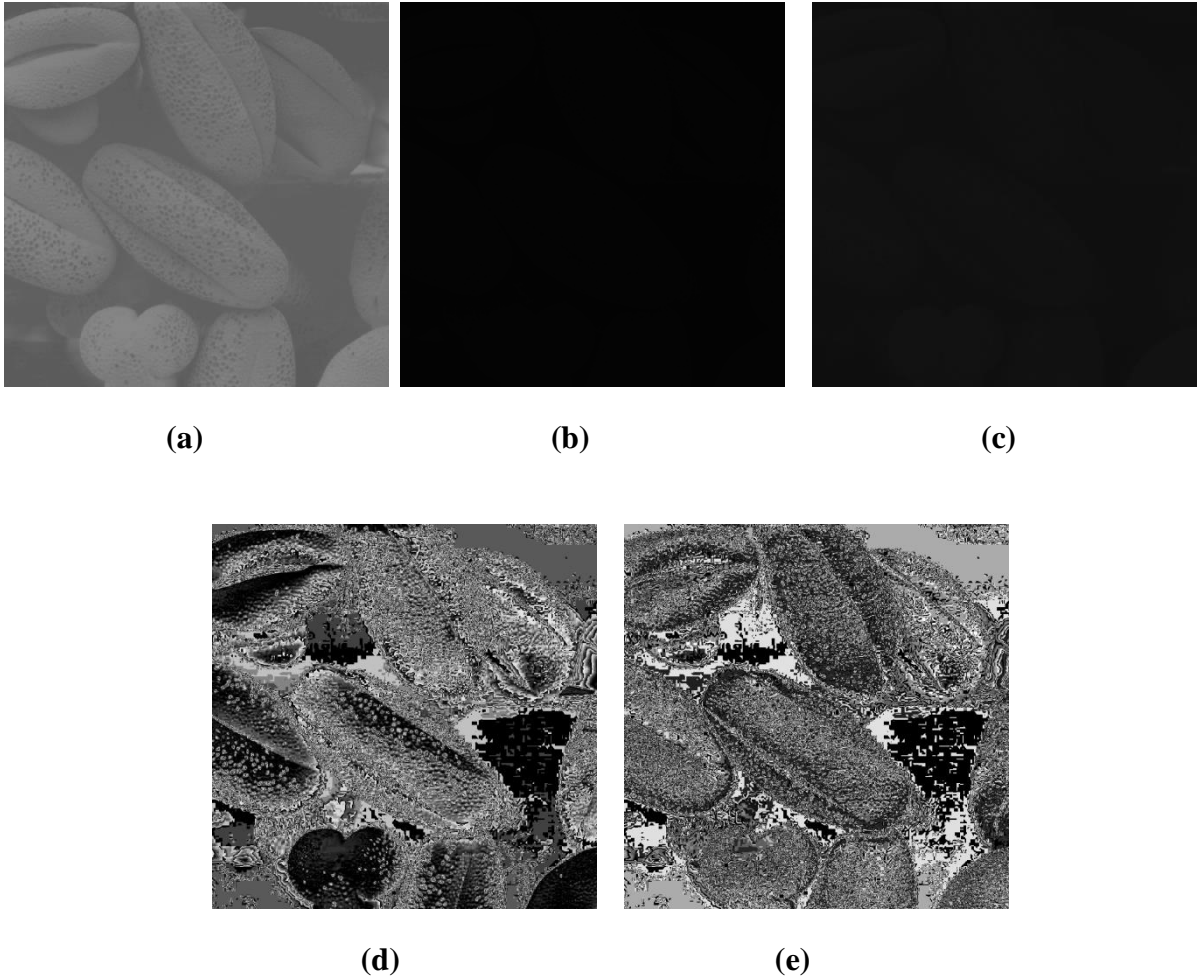
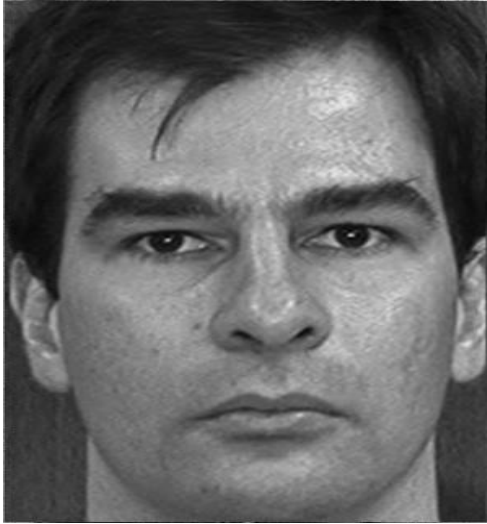


Fig 4.11 Result of image negative, (a) original image, (b) negative of the image

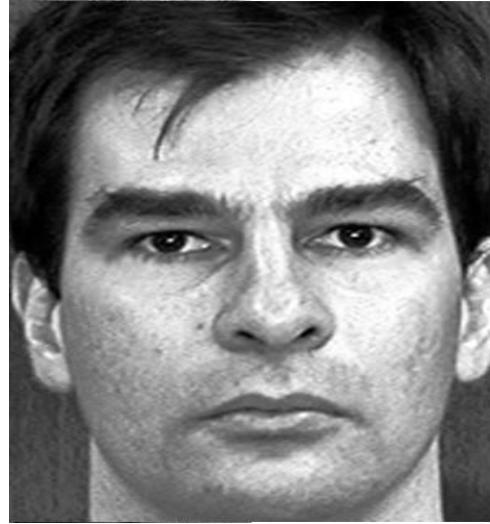


**Fig 4.12 Results obtained by gamma correction or power law transform, (a) Original image, (b) image with  $\gamma=5$ , (c) image with  $\gamma=2$ , (d) image with  $\gamma=0.3$ , (e) image with  $\gamma=0.6$**





(a)



(b)

**Fig 4.13 Results of histogram equalization, (a) Original image, (b) Equalized image.**



## **CHAPTER 5**

# CONCLUSION AND FUTURE SCOPE

## 5.1 CONCLUSION

The image enhancements techniques have become important pre-processing tool for digital vision processing applications like satellite imaging medical imaging etc. Many image enhancement techniques are used to explore the detail hidden in an image or to improve contrast in the low contrast image. Image enhancement process try to find to increase the interpretability or perception of information in the images to provide better input for other automated image processing steps.

In this we discussed about the image processing, image enhancement and different image enhancement techniques. We examined different enhancement techniques in different environments i.e. different software's such as MATLAB, XILINX-VIVADO and python. We observed that different techniques give different results. Image negative is used in medical imaging in order to detect tumors. Log transform is used to enhance satellite images so that we can extract the exact information. We observed that Histogram equalization improved the contrast of the image by effectively spreading out the most frequent intensity values and contrast stretching increased the contrast of the image by stretching the pixel values. Median filtering is used for the removal of noises and we observed that it removes salt and pepper noise effectively.

We also measured MSE and PSNR for all the techniques and measured the performance. If the PSNR is high then the error in that will be very less. Using this parameters only we conclude that median filtering effectively removes the salt and pepper noise.

## 5.2 FUTURE SCOPE

The proposed techniques for this enhancement has been done for improving quality of the image. Depends on the type of the image suitable enhancement method is performed on the image. In future a complete framework can be developed which can check the image and automatically choose the method and enhances the image irrespective of the input type. This framework can reduce the manual work and computation time still further.

The image has been enhanced by using matlab for checking their performance results. Some image enhancement techniques has been implemented by using python and one image enhancement technique has been implemented in Xilinx vivado for measuring their performance. In future all image enhancement techniques has been implemented by using xilinx vivado for checking their measures.

## REFERENCES

- [1] T.M.Bittibssi, G.I.Salama, Y.Z.Mehaseb, Adel E. Henawy, Image enhancement algorithms using FPGA, International Journal of Computer science and communication networks, Vol2(4), 536-542, july 2013.
- [2] S.Sowmya, Roy Paily, "FPGA implementation of image enhancement algorithms", International Conference Communications and Signal Processing (ICCSP), pp.584-588, Feb.2011.
- [3] Karan Kumar, Aditya Jain and Atul Kumar Srivastava "FPGA implementation of image enhancement techniques", Proc. Of SPIE, Vol.750208-7, September 2011.
- [4] Lee, Eunsung, et al. "Contrast Enhancement Using Dominant Brightness Level Analysis and Adaptive Intensity Transformation for Remote Sensing Images": pp.62-66 IEEE, 2013.
- [5] Veena, G., V. Uma, and Ch.Ganapathy Reddy. "Contrast Enhancement for Remote Sensing Images with Discrete Wavelet Transform." International Journal of Recent Technology and Engineering (IJRTE), IEEE, 2013.
- [6] Srivastava, Gaurava, and Tarun Kumar Rawat. "Histogram equalization: A comparative analysis & a segmented approach to process digital images." Contemporary Computing (IC3), 2013 Sixth International Conference on. IEEE, 2013.
- [7] Cheng, H. D., and Yingtao Zhang. "Detecting of contrast over-enhancement." Image Processing (ICIP), 2012 19th IEEE International Conference on. IEEE, 2012.
- [8] Ghimire, Deepak, and Joonwhoan Lee. "Nonlinear transfer function-based local approach for color image enhancement." Consumer Electronics, IEEE Transactions on 57.2: pp.858-865, 2011.
- [9] Maragatham, G., S. MdMansoorRoomi, and T. Manoj Prabu. "Contrast enhancement by object based Histogram Equalization." Information and Communication Technologies (WICT), 2011 World Congress on. IEEE, 2011.
- [10] Chauhan, Ritu, and Sarita Singh Bhadoria. "An improved image contrast enhancement based on histogram equalization and brightness preserving weight clustering histogram equalization." Communication Systems and Network Technologies (CSNT), 2011 International Conference on. IEEE, 2011.
- [11] Josephus, ChelsySapna, and S. Remya. "Multilayered Contrast Limited Adaptive Histogram Equalization Using Frost Filter." Recent Advances in Intelligent Computational Systems (RAICS), 2011 IEEE. IEEE, 2011.

- [12] Demirel, Hasan, CagriOzcinar, and GholamrezaAnbarjafari. "Satellite image contrast enhancement using discrete wavelet transform and singular value decomposition." *Geoscience and Remote Sensing Letters, IEEE* 7.2: pp.333-337, 2010.
- [13] Ke, Wei-Ming, Chih-Rung Chen, and Ching-Te Chiu. "BiTA/SWCE: Image enhancement with bilateral tone adjustment and saliency weighted contrast enhancement." *Circuits and Systems for Video Technology, IEEE Transactions on* 21.3: pp.360-364, 2010.
- [14]Murahira, Kota, Takashi Kawakami, and Akira Taguchi. "Modified histogram equalization for image contrast enhancement." *Communications, Control and Signal Processing (ISCCSP), 2010 4th International Symposium on. IEEE*, 2010.
- [15] Sasi Gopalan, Madhu S Nair and Souriar Sebastian “Approximation Studies on Image Enhancement Using Fuzzy Technique” *International Journal of Advanced Science and Technology*, Vol. 10, pp.11-26, September, 2009.
- [16] Zhengmao Ye, Habib Mohamadin, Su-Seng Pang,Sitharama Iyengar “Contrast Enhancement and Clustering Segmentation of Gray Level Images with Quantitative Information Evaluation” *Weas Transaction on Information Science & Application* Vol. 5, pp.181-188 , February 2008.
- [17] Bhabatosh Chanda and Dwijest Dutta Majumder, *Digital Image Processing and Analysis*, Prentice-Hall of India, 2002.
- [18] Raman Maini and Himanshu Aggarwal” A Comprehensive Review of Image Enhancement Techniques” *Journal of Computing*, Vol. 2, pp.8-13, March 2010.
- [19] S. C. Pei, Y. C. Zeing, and C. H. Chang, "Virtual restoration of ancient chinese painting using color contrast enhancement and lacuna texture synthesis," *IEEE Trans. Image Processing*, Vol. 13, pp. 416-429,2004.
- [20] W. A., S. H. Chin, and E. C. Tan, "Novel approach to automated fingerprint recognition," *IEEE Proceedings Vision Image and Signal Processing*, Vol. 145, pp. 160-166, 1998.
- [21] A.de la Torre, A. M. Peinado, J. C. Segura “Histogram equalization of speech representation for robust speech recognition," *IEEE Trans. Speech Audio Processing*, Vol. 13, pp.355-366, 2005.
- [22] S. M. Pizer, "The medical image display and analysis,"*IEEE Trans. Med. Image*, Vol. 22, pp.2-100, 2003.
- [23] A. Rafael C. Gonzalez, and Richard E. Woods, “Digital Image Processing,” 2nd edition, Prentice Hall, 2002.
- [24] Joung-Youn Kim, Lee-Sup Kim and Seung-Ho Hwang, “An advanced contrast enhancement using partially overlapped sub-block histogram equalization,” *IEEE Trans. Circuits Syst. Video Technol.*, Vol. 11, pp.475-484, April 2001.

- [25] Y.-T. Kim, "Contrast enhancement using brightness preserving bi-histogram equalization," IEEE Trans. on Consumer Electronics, Vol. 43, pp.1-8, February 1997.
- [26] Y. Wang, Q. Chen, and B. Zhang, "Image enhancement based on equal area dualistic sub-image histogram equalization method," IEEE Trans. on Consumer Electronics, Vol. 45, pp.68-75, February 1999.
- [27] S.-D. Chen and A. Ramli, "Minimum mean brightness error bi-histogram equalization in contrast enhancement," IEEE Trans. on Consumer Electronics, Vol. 49, pp.1310-1319, November 2003.
- [28] Soong-Der Chen and Abd. Rahman Ramli, "Contrast enhancement using recursive mean-separate histogram equalization for scalable brightness preservation, IEEE transactions on Consumer Electronics, Vol.49, pp.1301-1309, November 2003.
- [29] David Menotti, Laurent Najman, Jacques Facon, and Arnaldo de A. Araújo" Multi-Histogram Equalization Methods for Contrast Enhancement and Brightness Preserving" IEEE Transactions on Consumer Electronics, Vol. 53, pp.1186-1194, August 2007.
- [ 30] Fan Yang, Jin Wu "An Improved Image Contrast Enhancement in Multiple-Peak Images Based on Histogram Equalization" IEEE International Conference on Computer Design and Applications, Vol.1, pp.346-349, 2010.
- [31] P. Rajavel "Image Dependent Brightness Preserving Histogram Equalization" IEEE Transactions on Consumer Electronics, Vol. 56, pp.756-763, May 2010.

## APPENDIX

### CODES USED:

#### Image negative :

```
clc;
close all;
clear all;
img=imread('C:\Users\SAITEJA\Desktop\tulips.png');
img1=255-img;
figure
subplot(1,2,1)
imshow(img)
title('original image')
figure
subplot(1,2,2)
imshow(img1)
title('image after neg transform')
```

#### log transform:

```
clc;
close all;
clear all;
i=imread('C:\Users\SAITEJA\Desktop\cancer.png');
i=rgb2gray(i);
img2=log10(1+256*im2double(i));
img2= [img2-min(img2(:))]./max(img2(:)-min(img2(:)));
figure
subplot(1,2,1)
imshow(i)
title('original image')
subplot(1,2,2)
imshow(img2)
title('image after log transform')
```

#### gamma correction:

```
clc;
clear all;
close all;
i=imread('C:\Users\SAITEJA\Desktop\ori.jpg');
img= double(i).^(0.3);
img2= double(i).^(0.6);
img3= double(i).^(2);
img4= double(i).^(5);
img=[img-min(img(:))]./max(img(:)-min(img(:)));
img2=[img2-min(img2(:))]./max(img2(:)-min(img2(:)));
img3=[img3-min(img3(:))]./max(img3(:)-min(img3(:)));
img4=[img4-min(img4(:))]./max(img4(:)-min(img4(:)));
figure(1)
imshow(i)
title('original image')
figure(2)
imshow(img)
title('image with r=0.3')
```

```

figure(3)
imshow(img2)
title('image with r=0.6')
figure(4)
imshow(img3)
title('image with r=2')
figure(5)
imshow(img4)
title('image with r=5')

```

contrast stretching:

```

clc;
clear all;
close all;
img=imread('C:\Users\SAITEJA\Desktop\seeds.jpg');
img=rgb2gray(img);
rmin=min(img(:));
rmax=max(img(:));
r=0:255;
s=zeros(size(r));
s(1:find(s==rmin))=0;
step=length(r(find(r==rmin):find(r==rmax)));
s(find(r==rmin):find(r==rmax))=0:255./step:255-255./step;
s(find(r==rmax)+1:end)=255;
img2=double(img);
img2=[img2-min(img2(:))]/max(img2(:)-min(img2(:)));
immean=round(mean(img(:)));img3=img;
img3(find(img>=immean))=255;img3(find(img<immean))=0;
subplot(2,3,1);plot(r,s);axis([0 255 -2 259]);
xlabel('input gray level,r');
ylabel('output gray level');
subplot(2,3,2);
imshow(img);
title('original image');
subplot(2,3,3);
imshow(img2);
title('contrast-stretched image');
subplot(2,3,4);
imshow(img3);
title('threshold image');

```

histogram equalization:

```

clc;
close all;
clear all;
a=imread('C:\Users\SAITEJA\Desktop\camera.png');
b=rgb2gray(a);
subplot(2,2,1);
imshow(b);
title('original image');
subplot(2,2,2);
imhist(b,256);
title('histogram plot of the image ');
J=histeq(b);
subplot(2,2,3);
imshow(J);
title('eqalized histogram plot of the image');
subplot(2,2,4);
imhist(J,256);

```



```
title('equalized image');
```

## median filtering

```
C=imread('C:\Users\SAITEJA\Desktop\lena.png');
A=rgb2gray(C);
figure
imshow(A)
title('input image')
A=imnoise(A,'salt & pepper',0.02);
figure,imshow(A);
title('image witj salt and papper noise')

%PAD THE MATRIX WITH ZEROS ON ALL SIDES
modifyA=zeros(size(A)+2);
B=zeros(size(A));

%COPY THE ORIGINAL IMAGE MATRIX TO THE PADDED MATRIX
    for x=1:size(A,1)
        for y=1:size(A,2)
            modifyA(x+1,y+1)=A(x,y);
        end
    end
    %LET THE WINDOW BE AN ARRAY
    %STORE THE 3-by-3 NEIGHBOUR VALUES IN THE ARRAY
    %SORT AND FIND THE MIDDLE ELEMENT

for i= 1:size(modifyA,1)-2
    for j=1:size(modifyA,2)-2
        window=zeros(9,1);
        inc=1;
        for x=1:3
            for y=1:3
                window(inc)=modifyA(i+x-1,j+y-1);
                inc=inc+1;
            end
        end

        med=sort(window);
        %PLACE THE MEDIAN ELEMENT IN THE OUTPUT MATRIX
        B(i,j)=med(4);

    end
end
%CONVERT THE OUTPUT MATRIX TO 0-255 RANGE IMAGE TYPE
B=uint8(B);
figure,imshow(B);
title('image after median filtering')
```