**CLUSTER TREE STRUCTURE BASED IMPLEMENTATION OF A FILTER FOR DENOISING TEXTURE PATTERNS**

**A PROJECT REPORT**

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

**MS**

in

**Software Engineering**

*By*

**S.SAHANA(11MSE0062)**

**Under the Guidance of**

**Prof. Hemalatha.S**

**Assistant Professor (Sr), SITE**

**VIT University**

****

**SCHOOL OF INFORMATION TECHNOLOGY AND ENGINEERING**

**SOFTWARE ENGINEERING DIVISION**

**November – 2014**

****

**School of Information Technology and Engineering**

**DECLARATION BY THE CANDIDATE**

I hereby declare that the thesis entitled “**Cluster tree structure based implementation of a filter for denoising texture patterns”** submitted by us to VIT University Vellore, in partial fulfillment of the requirement for the award of the degree of **MS-Software Engineering** is a record of bonafide project work carried out by me under the supervision of **Prof.Hemalatha.S**, VIT University. I further declare that the work reported in this project has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

**Place**: Vellore

**Date**: 11-11-2014 **Signature of the Candidate(s)**

****

**School of Information Technology and Engineering**

**BONAFIDE CERTIFICATE**

This is to certify that the project work entitled “**Cluster tree structure based implementation of a filter for denoising texture patterns”** by **S.Sahana(11MSE0062)** and **Aparna.R(11MSE0216)** to VIT University, Vellore, in partial fulfillment of the requirement for the award of the degree of **MS-Software Engineering**, is a record of bonafide work carried out by her under my supervision. The project fulfills the requirement as per the regulations of this Institute and in my opinion meets the necessary standards for submission. The contents of this report have not been submitted and will not be submitted either in part or in full, for the award of any other degree or diploma in this Institute or any other Institute or University.

Place: Vellore Signature of Guide

Date:08/11/2014 Prof.Hemalatha.S

School of Information and Technology and Engineering

VIT University

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

The thesis is satisfactory/unsatisfactory

Internal Examiner External Examiner

Date : 12-11-2014 Date :12-11-2014

**Acknowledgement**

I wish to express our heartfelt gratitude to **Dr.G.Viswanathan**, President/ Chancellor, Vellore Institute of Technology University, Vellore for providing facilities for the fourth semester project.

I am highly grateful to our Vice Presidents, **Shri. Sankar Viswanathan** ,

**Shri. Sekar Viswanathan and Shri.G.V.Selvam**, **Dr. V. Raju,** Vice Chancellor, **Dr. S. Narayanan,** Pro-Vice Chancellor for providing necessary resources.

My sincere gratitude to **Dr.K.Ganesan** Dean(i/c), School of Information Technology and Engineering for giving us the opportunity to undertake the project.

I wish to express my sincere gratitude to **Prof.R.Srinivasa Perumal** Programme Manager of M.S-SE, and **Prof.Hemalatha.S** project coordinator/MS School of Information Technology and Engineering for providing me an opportunity to do my project work in the VIT University*.*

I would like to express my special gratitude and thanks to my internal guide

**Prof.Hemalatha.S,** School of Information Technology and Engineering whose esteemed guidance and immense support encouraged to complete the project successfully.

I thank the Management of VIT University for permitting me to use the library resources. I also thank all the faculty members of VIT University for giving me the courage and strength I needed to complete my goals. This acknowledgement would be incomplete without expressing my whole hearted thanks to my family and friends who motivated me during the course of the work.

I thank the Almighty, whose showers of grace were felt during the entire course of my project.

Place: Vellore

Date: 12-11-2014

Name of Student(s): Aparna R, Sahana S

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO.** | **TITLE** | **PAGE NO.** |
|  |  |  |
|  | **ABSTRACT** | 8 |
|  | **LIST OF FIGURES** | 9 |
|  | **LIST OF TABLES** | 10 |
|  | **LIST OF ACRONYMS** | 11 |
|  |  |  |
| 1 | **INTRODUCTION** | 12 |
|  | 1.1 BACKGROUND | 12 |
|  | 1.2PROBLEM STATEMENT AND MOTIVATION | 13 |
|  | 1.3 CHALLENGES | 13 |
|  | 1.4 ASSUMPTIONS | 14 |
|  | 1.5 ORGANIZATION OF THE REPORT | 14 |
|  | 1.6 AIMS AND OBJECTIVES | 15 |
|  |  |  |
| 2 | **literature survey** |  |
|  | 2.1 PROJECT OVERVIEW | 16 |
|  | 2.2 PROJECT PRELIMINARY DESIGNX | 17 |
|  | 2.3 PROJECT REQUIREMENTS | 17 |
|  | 2.4 PROJECT ANALYSIS AND USER REQUIREMENTS | 18 |
|  | 2.5 LITERATURE REVIEW | 18 |
|  | 2.6 APPROACHES | 19 |
|  | 2.6.1 INPUT | 20 |
|  | 2.6.2 PROCESS | 20 |
|  | 2.6.3 PERFORMANCE METRICS | 20 |
|  | 2.6.4 OUTPUT | 21 |
|  |  |  |
| **3** | **DESIGN** | **22** |
|  | 3.1 OVERVIEW | 22 |
|  | 3.2ARCHITECTURE/MODULE DESCRIPTION | 22 |
|  | 3.2.1 IMAGE CAPTURING | 23 |
|  | 3.2.2 ADDITION OF THE NOISE TO THE ‘ IMAGE | 23 |
|  | 3.2.3 APPLICATION OF THE FILTERS  ON THE IMAGE | 24 |
|  | 3.2.4 APPLICATION OF THE EFFICIENT  NON-LOCAL MEAN FILTER | 26 |
|  | 3.2.5 DENOISED IMAGE | 29 |
|  | 3.3 ENGINEERING USECASE | 30 |
|  | 3.4 SEQUENCE DIAGRAM | 31 |
|  | 3.5 ACTIVITY DIAGRAM | 32 |
|  | 3.6 CLASS DIAGRAM | 32 |
|  | 3.7 ISSUES AND CONCERNS | 32 |
|  |  |  |
| 4 | **IMPLEMENTATION OF THE SYSTEM** |  |
|  | 4.1 PROJECT OVERVIEW | 33 |
|  | 4.2 CODING AND IMPLEMENTATION | 33 |
|  | 4.2.1 DATA IMPORTING | 33 |
|  | 4.2.1 CODING AND SIMULATION | 33 |
|  | 4.3 TOOLS AND TECHNIQUES USED | 42 |
|  | 4.4 TESTCASE SPECIFICATION | 43 |
|  | 4.4.1 UNIT TESTCASE SPECIFICATION | 43 |
|  | 4.4.2 INTEGRATION TESTCASE  SPECIFICATION | 44 |
|  | 4.4.3 TESTDATA REQUIRED | 45 |
|  |  |  |
| 5 | **RESULTS AND DISCUSSION** | 46 |
|  | 5.1 PERFORMANCE METRICS USED | 46 |
|  | 5.2 RESULTS | 47 |
|  | 5.3 SCREENSHOTS/PERFORMANCE  METRICS TABLES | 47 |
|  |  |  |
| 6 | **CONCLUSION AND FUTURE WORK** |  |
|  | 6.1 CONCLUSION | 52 |
|  | 6.2 FUTURE WORK | 53 |
|  |  |  |
|  | **APPENDIX** | 54 |
|  | **REFERENCES** | 61 |

**ABSTRACT**

Digital Image Processing forms an important element in the generation as well as transformation of an image. As far as an image is concerned, its quality and clarity matters. To restore the images, there already exists several image filters belonging to both local and non-local categories. Each of these filters have their own advantages and drawbacks especially in terms of clarity and computational speed. To overcome these issues, we propose an efficient non-local means filter for the restoration of the image based on arranging the data in the form of a cluster tree. Structuring of the data is done on the basis of preselection of patches and we make use of distance as the parameter while patch selection. Hence, this approach contributes to larger speedups, better quality and less computational cost especially when the filter is truly non-local.

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Figure Name** | **Page No.** |
| 1. | Preliminary Project Design |  |
| 2. | Architecture overview for Denoising of image using non-local mean filter |  |
| 3. | Unfiltered values |  |
| 4. | the mean filtered values of a 3x3 kernel |  |
| 5 | Arrangement of the data as clusters of similar patches |  |
| 6. | Illustration of clustering concept. |  |
| 7. | Matlab Environment |  |
| 8. | Mathematical formulae for PSNR and MSE |  |

**LIST OF TABLES**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Table Name** | **Page No** |
| 1 | Table 4.1 UTC001 |  |
| 2 | Table 4.2 ITC001 |  |
| 3. | Table 5.1 PSNR and MSE values for noises |  |

**LIST OF ACRONYMS**

**PSNR-**Peak-Signal-to-Noise-Ratio

**MSE**-Mean Square Error

**UINTA-**Unsupervised Information-Theoritic Adaptive filter

**NL**-Non-Local

**RGB-**Red Green Blue

**BMP**-Bit Map

**JPG/JPEG-** Joint Photographic Experts Group

**GIF-** Graphics Interchange Format

**PNG-** Portable Networks Graphic

**TSVQ**-Tree Structured Vector Quantization

**UTC-**Unit Test Case

**ITC**-Integration Test Case

**CHAPTER 1**

**INTRODUCTION**

**1.1 BACKGROUND**

Image denoising is one of the most important concepts in computer vision. It is widely used in various image related applications such as MRI analysis, 3-D object recognition e.t.c. Most digital images contain some degree of noise. The main goal of image denoising is to restore the details of an image by removing unwanted noise. Theoretically, the denoised image should not contain any form of noise. Over the years, several denoising approaches have been proposed. Some of them are

1. Gaussian filtering that is used to blur or smooth the image in a fashion similar to that of a mean filter and the degree of smoothing is determined by the degree of standard deviation of the Gaussian.

2. Weiner filter that makes use of inverse filtering (by blurring the image using a known low-pass filter) and removes the additive noise as well as blurring simultaneously.

3. Median filter that removes the noise and functions similar to the mean filter. This performs better than the mean filter and preserves useful detail in the image.

4. Bilateral Filter that makes use of a simple, non-iterative scheme for edge-preserving smoothing.

**1.2 PROBLEM STATEMENT AND MOTIVATION**

Non-local means is an algorithm in digital image processing for denoising the image that takes the mean value of all the pixels present in an image weighed by how similar these pixels are with the target pixel. On comparison with the existing local and semi-local mean filters, the non-local method of denoising preserves more details of an image, thereby resulting in an increased post-filtering clarity.

The computational complexity of non-local filters is usually more and more amounts may need to be paid for achieving greater results i.e, at each pixel, weights to all other pixels have to be computed. Hence, in case of larger images, this complexity becomes quite a burden. This has been the main motivation for this project. In order to resolve this issue, we have found that other than the existing approximations, the best method is to carry out the process of pre-selection of patches in the form of a binary tree using distance as the parameter for the pre-selection. Thus, the cluster tree approach forms an answer for the question of how to find the approximate neighbors in high-dimensional data with sub-linear time complexity. The calculated PSNR value also highlights its importance.

**1.3** C**HALLENGES**

Improving the quality of the restored images with less computational complexity and time is an important and difficult task in many applications. There exists many algorithms for bringing out an improved version of non-local means filter. But still the research for improving these algorithms has not stopped. In case of a non-local filter, most of the time gets consumed for computing the distance between patches.

In the nonlocal means ﬁlter, almost all computation time is spent on computing distances between patches. Thus, in order to speed up the filter, preselection of patches can be done using some alternative distances, that can be carried out quickly, such as the difference of the patches’ means or variance. Even though, it results in a speedup, this method has a drawback i.e, preselection criterion doesn’t have much relation with the distance of the patches. For example, two patches having the same variances and means contain very different textural structures. Although, the above mentioned idea does not affect the filtering outcome much, it reduces the efficiency of the method. This stands up as a challenge for the project. So, we need a preselection technique that will adopt the preselection technique making use of the concept of a cluster tree.

**1.4 ASSUMPTIONS**

The assumptions made for this project is that the images/textures were taken from the existing standard applications such as cameraman.jpg, lena.jpg , check.jpg e.t.c. These are the most commonly images for obtaining the filtering results. We have also made use of biometric fingerprint images that belongs to CASIA database.

**1.5 ORGANIZATION OF THE REPORT**

The report is organized as follows .i.e, a literature review of the previous works in the filtering mechanisms and algorithms. Further, the design that our project adopts is explained. The details about the implementation of the proposed methodology are explained in the next section after design. This is followed up by the results and discussions about the future works, and the conclusions drawn are explained in the further sections.

**1.6 AIMS AND OBJECTIVES**

The objective of this project is to improve the existing non-local mean filter using a cluster tree based approach and hence restore the image preserving the texture patterns.

The main aim is to study the application of Digital Image Processing concepts of image enhancement and restoration on images using filtering mechanisms.

**CHAPTER 2**

**LITERATURE SURVEY**

**2.1 PROJECT OVERVIEW**

The project focuses on the existing concepts of non-local mean filter mechanisms and tries to come up with an algorithm to implement efficient non-local mean filter for the denoising of the texture patterns by adopting the concept of clustering. Here, first of all, standard images such as lena.jpg, cameraman.jpg e.t.c were taken as sample input images. After this, we have carried out the process of image enhancement on these sample images by applying several existing filters such as Gaussian smoothing filter, median filter, Weiner filter e.t.c. on them. Later, a non-local mean filter that clusters the data in the form of patches was implemented.

The preselection of patches were carried out on the basis of distance and similarity as measure.

The output that we obtained as a result of the implementation was the enhanced images and their performance were evaluated by calculating PSNR values and Mean Square Errors (MSE) associated with each enhanced image. The values were tabulated and further compared to find the most efficient method and the non-local mean filter based on the cluster tree approach was found to be an efficient one that reduced the computational complexity and preserved the finer details of the texture patterns.

**2.2. PROJECT PRELIMINARY DESIGN**

The following figure represents the preliminary approach used for denoising the sample image using a non-local mean filter.

ADD NOISE TO THE INPUT IMAGE

APPLY FILTERS ON THE IMAGE

DENOISED IMAGE

STANDARD SAMPLE IMAGE

Figure 1.Preliminary Project Design

**2.3 PROJECT REQUIREMENTS**

For carrying out the coding and implementation of the project, our system requires Matlab for importing the images, applying noise and obtains the output i.e, the restored image on the output window.

Since Matlab is processor intensive, it is necessary that the system has 4GB RAM minimum and a hard disk space of minimum 5GB.These are the hardware requirements.

**2.4 PROJECT ANALYSIS & USER REQUIREMENTS**

The proposed methodology is analyzed depending on the user requirements and a feasibility study is done on the given requirements and it has been found that the project is feasible to be implemented.

The user requirements are as follows.

A clear and quality denoised image with maximum preservation of the minute details of the texture patterns. Since, the images need to be fed as the input to the Matlab; the entire process should be smooth and hustle-free, supporting different types of images.

The proposed method should result in an easier generation and calculation of PSNR and MSE values in a considerably speedy manner.

**2.5 LITERATURE REVIEW**

There have been enough on-going researches regarding a better method for restoration of images by retaining texture patterns with less detail loss. Through thorough collection of information, different filters along with their advantages and disadvantages were found and few of the filters were non-linear diffusion filters, ROF filters and certain types of wavelet shrinkage. They were helpful in removing noise from the images while preserving the structures. But, even after being successful in various fields, they come up with a major drawback in case of oscillatory structures. The texture structures are often confused with noise and are removed.

Few other filters that are applied are as follows:-

* Yaroslavsky neighborhood filter-Here, the smoothed image is obtained by taking the weighted average of pixels of the original image. The weights are found by a non-negative kernel function K, which decays with distance.
* Bilateral filter- This is an iterative version of the filter where it systematically loops through each pixel and adjust weights to the adjacent pixels accordingly.
* UINTA filter-This filter helps in the restoration of the pixels by comparing pixel values with the other pixels in the image that have similar neighbourhood, thereby ensuring more preservation of small-scale texture patterns.

Information regarding more filters is provided in Appendix 1.

There are further major classifications for applying these filters and they are:

* Local means, where target pixel is compared with the pixels surrounding(neighbours) it and their mean value is found to smooth the image.
* Non-local means, where the mean of all the pixels in an image is taken and weighed by how similar these pixels are with respect to the target pixel.
* Semi-local filters, forms as a combination of local and non-local filters.

**2.6** **APPROACHES:-**

In [1], the author proposed a NL-means algorithm by using the concept of “method noise”, defined as the difference between a digital image and denoised version.

In [2], the author implemented the NL-means method by the preselection of neighbourhood on the basis of average value and gradient.

In [3], the author shows that there exists an easier implementation of NL-means method by using convolution routines, which accelerates the method significantly in comparison with the traditional method. Also, their algorithm made use of infinite-size non-binary patches, which improves the denoising quality.

**2.6.1 INPUT**

The input that are generally accepted are the images, say standard sample images such as Barbara.jpg, lena.jpg, Cameraman.jpg e.t.c. Images were also taken from various databases.

**2.6.2 PROCESS**

Primarily, the image is imported to Matlab. The next step involved the addition of noise to the imported image followed by its removal on the application of non-local smoothing filters. These filters and the non-local means algorithm are compared.

**2.6.3 PERFORMANCE METRICS**

Performance of various filters are measured using

* PSNR (Peak-to-Signal-Noise) - PSNR is the ratio between maximum possible power of a signal and the power of distorting noise which affects the quality of its representation.
* MSE (Mean Square Error): The MSE is the cumulative square error between the encoded and the original image. The size of the original image is mxn and lesser the value of MSE, more is the compression possible.

**2.6.4 OUTPUT**

The restored image is obtained as the final output.

# CHAPTER-3

**DESIGN**

**3.1 OVERVIEW**

The major modules involved in the implementation of non-local mean filter are given below:-

Fig.2 Architecture overview for Denoising of image using non-local mean filter

**3.2 ARCHITECTURE/MODULE DESCRIPTION**

The denoising usually consists three main processes-image capturing, apply filters to the image and denoise the image.

The different components for obtaining a denoised image are as follows.

**3.2.1 IMAGE CAPTURING**

Captured images are usually of the format types such as

* Binary images
* Gray scale images
* True or RGB images
* Indexed images
* JPG,JPEG
* PNG
* BMP
* GIF e.tc.

Each of these images is used in different situations.

**3.2.2 ADDITION OF THE NOISE TO THE IMAGE**

The main source of noise in digital images arises during image acquisition (digitization) .The performance of image sensor is affected by variety of reasons such as environmental condition during image acquisition or by the quality of the sensing element themselves. For instance, during acquiring images with CCD camera, sensor temperature and light levels are major factors that affect the amount of noise in the resulting image. The principal reason of noise is due to interfering in the channel which is used for the images transmission .We can model a noisy image as follows:

C(x, y) = A(x, y)+B(x, y)

Where A(x, y) is the original image pixel value and B(x, y) is the noise in the image and C(x, y) is the resulting noise image.

Some of the commonly added noises have been explained below.

* **Uniform Noise**: The uniform noise caused by quantizing the pixels of image to a number of distinct levels is known as quantization noise. It has approximately uniform distribution. In the uniform noise, the level of the gray values of the noise is uniformly distributed across a specified range. Uniform noise can be used to generate any different type of noise distribution. This noise is often used to degrade images for the evaluation of image restoration algorithms. This noise provides the most neutral or unbiased noise.
* **Gaussian noise**: It is also known as Gaussian distribution. It is a major part of the read noise of an image sensor that is of the constant level of noise in the dark areas of the image.
* **Salt and Pepper noise**: The salt-and-pepper noise are also called shot noise, impulse noise or spike noise that is usually caused by faulty memory locations ,malfunctioning pixel elements in the camera sensors, or there can be timing errors in the process of digitization .

**3.2.3 APPLICATION OF FILTERS ON IMAGE**

To the corresponding image that contains the added noise, we need to apply several common filters. Filtering in an image processing is a basis function that is used to achieve many tasks such as noise reduction, interpolation, and re-sampling. Filtering image data is a standard process used in almost all image processing systems. The choice of filter is determined by the nature of the task performed by filter and behavior and type of the data. Filters are used to remove noise from digital image while keeping the details of image preserved is an necessary part of image processing. Filters can be described by different categories:--

* **Linear filters**: Linear Filters: Linear filters are used to remove certain type of noise. Gaussian or Averaging filters are suitable for this purpose. These filters also tend to blur the sharp edges, destroy the lines and other fine details of image, and perform badly in the presence of signal dependent noise.
* **Mean filter**: The mean filter is a simple spatial filter .It is a sliding-window filter that replaces the center value in the window. It replaces with the average mean of all the pixel values in the kernel or window. The window is usually square but it can be of any shape. For instance,

|  |  |  |
| --- | --- | --- |
| 8 | 4 | 7 |
| 2 | 1 | 9 |
| 5 | 3 | 6 |

Fig 3. Unfiltered values.

The calculated mean value is 8+4+7+2+1+9+5+3+6=45/9=5

|  |  |  |
| --- | --- | --- |
| \* | \* | \* |
| \* | 5 | \* |
| \* | \* | \* |

Fig.4 shows the mean filtered values of a 3x3 kernel

* **Weiner filter:** The purpose of the Wiener filter is to filter out the noise that has corrupted a signal. This filter is based on a statistical approach. Mostly all the filters are designed for a desired frequency response. Wiener filter deal with the filtering of an image from a different view. The goal of wiener filter is reduced the mean square error as much as possible. This filter is capable of reducing the noise and degrading function.

**3.2.4APPLICATION OF THE EFFICIENT NON-LOCAL MEAN FILTER**

Non-local means is an algorithm where the mean of all the pixels in an image are taken and are weighed with the target pixel to find out how similar it is with respect to the target pixel. The approach of Non Local Means filtering is based on estimating eachpixel intensity from the information provided from the entire image and hence it exploits the redundancy caused due to the presence of similar patterns and features in the image. Usually, the computational complexity is given by O (DN2) where D is the patch size and N is the number of pixels in the image. As the size of the images become large, computational complexity increases as well as the expenses also rises. It was found that by carrying out the patch comparison for a subset of reference patches lying on a coarser grid of the image, the computed weights helped in the restoration of a whole block of pixels at once. Definitely, this approach can be used to gain a signiﬁcant constant factor in the computational complexity, yet for the sake of quality the grid cannot be made arbitrarily coarse. In case of non- local ﬁltering, we are hence left with the quadratic time complexity of the original ﬁlter.

In patch based nonlocal ﬁltering, most of the time is consumed for computing distances between patches. However, only a relatively small part of all patches is similar for their kernel weights K(x, y) to play a role in the averaging. Hence, in order to speed up the ﬁlter, the basic idea of the approaches in has been to compute distances only for a reduced set of patches. Preselection of patches is performed by some alternative distances, which can be computed very quickly, such as the difference of the patches’ means or variances. Indeed, this strategy leads to a signiﬁcant speedup, particularly in case of large patches. The disadvantage of with this approach is that the preselection criterion is hardly related to the distance of patches. Two patches with same means and variances often contain different textural structures. This problem reduces the efﬁciency of the method, as the preselected set still contains a large number of dissimilar patches.

**ALGORITHM USED (PROPOSED STRATEGY):** **Cluster trees implementation**

The above encountered problem during preselection of the patches can be resolved by creating a set of similar patches. Here, the patches are selected on the basis of distance measure and is used in filtering. Thus, the basic idea to achieve this methodology is by arranging the data (image patches) in the form of a binary tree.

The concept can be understood by the following figure:

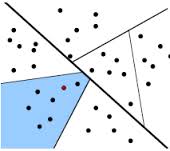


Fig.5 Arrangement of the data as clusters of similar patches.

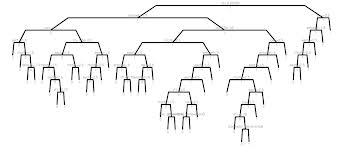


Fig.6 Illustration of clustering concept.

With respect to figure 6, the root node of this tree contains all patches in the image. We perform a k-means clustering with k = 2 which splits the patches into two clusters represented by the cluster means. These clusters represent the tree nodes at the ﬁrst level. Each of these nodes can again be separated via k-means. This way, we can recursively build up the cluster tree. The separation is stopped as soon as the number of patches in a node falls below a certain threshold Nmin = 30. So each leaf node comprises at least Nmin patches.

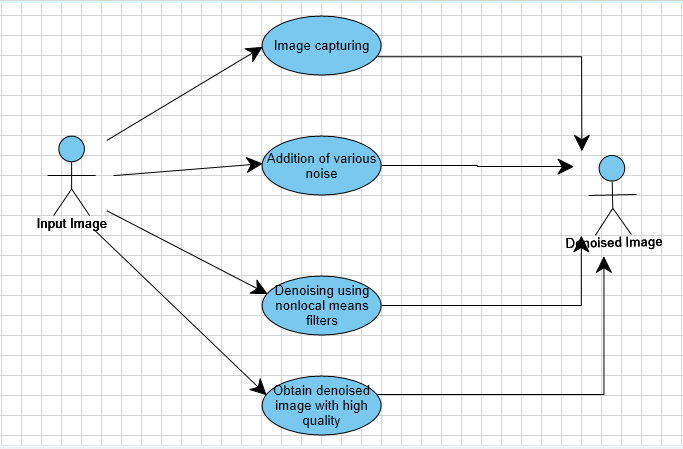
Finding a local optimum with k-means takes linear time. It has to be applied at each level of the tree, and there are log N levels. Thus, building the tree runs in O (DN log N). Once the tree is built up, we have immediate access to a set of similar patches in constant time. This is achieved by saving for each image patch its corresponding leaf node in an index table. The typical weighted averaging can then be efﬁciently applied on this subset. In comparison with the previous preselection techniques, usage of the same distance for clustering and for ﬁltering ensures preselection and ﬁltering by means of one consistent criterion. Fig. 2 illustrates this preselection in patch space. The concept of using tree structures to arrange data for efﬁcient access is well known in the field of computer science.

The earliest version of the above cluster tree is the so-called kd-tree. The kd-tree always splits the space along coordinate axes, whereas the cluster tree splits the space by means of arbitrary hyperplanes. Moreover, whereas the kd-tree is optimized and leads to more balanced tree, the cluster tree seeks to obtain leafs that correspond to natural clusters of the data. The cluster tree concept is also known as tree structured vector quantization (TSVQ) and has been applied in speech processing, image retrieval. Thus, the concept of cluster tree gives the answer for how to find approximate neighbors with less time complexity.

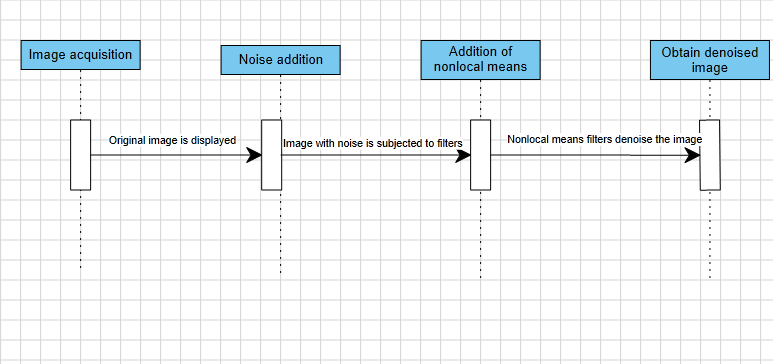
**3.2.5 DENOISED IMAGE**

A denoised image, preserving the details of the texture pattern is obtained as a result of the On the application of the non-local mean filter (that works on a cluster tree mechanism) on an image that contains noise, the final result obtained is a denoised image preserving the details of the texture patterns (especially edges) .Also, it required a linear computational complexity for the entire operation, thereby resulting in an increased speedup.

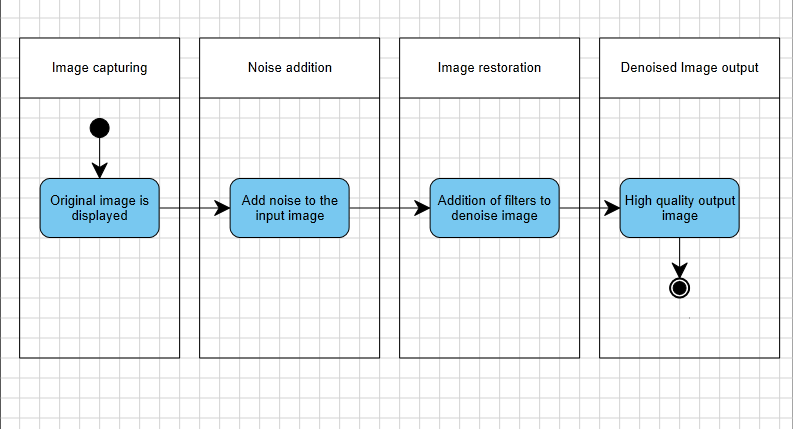
**3.3 ENGINEERING USECASE**

****

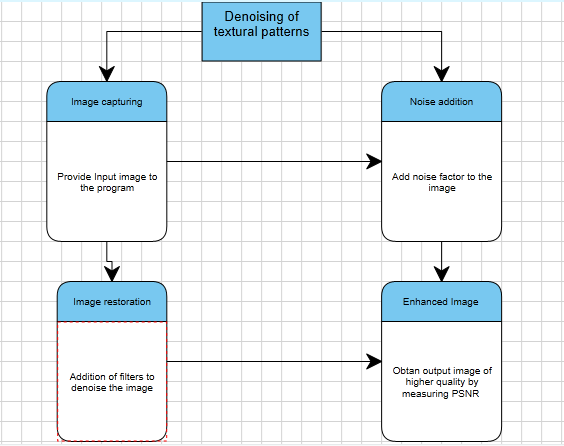
**3.4 SEQUENCE DIAGRAM**

****

**3.5 ACTIVITY DIAGRAM**

****

**3.6 CLASS DIAGRAM**

****

**3.7 ISSUES AND CONCERNS**

* The output obtained is a denoised image on the output window of MATLAB.
* The tool Matlab needs to be present on the system to import as well as obtain the final result image.

**CHAPTER 4**

**IMPLEMENTATION**

**4.1 PROJECT OVERVIEW**

The project aims at providing an efficient non-local means filter for denoising the image such that its details are preserved, using the cluster tree implementation. Here, the image data is clustered in the form of trees based on the similarity and distance measure.

**4.2 CODING AND IMPLEMENTATION**

The implementation strategy used included the following steps:

* Data importing
* Coding and Simulation

**4.2.1 Data Importing**

The data was imported to the Matlab tool. Digital Images such as true/RGB, gray scale images e.t.c were used. This formed as the input. There are certain standard images lena.jpg, Barbara.jpg upon which we can observe the filtering result.

**4.2.2 Coding and Simulation**

A major part of the project included Matlab coding for implementing the concept.

Few important modules were coded and the simulation results are obtained in the output window. It contained a denoised image with better clarity and less loss of details. Their respective codes have been mentioned below.

**Module-2** included addition of noise to the captured image.

**Coding for Noise 1:**

**Coding for Noise 2:**

**Coding for Noise 3:**

**Module-3** included application of certain existing simple filter on the above noisy images.

**Coding for Median Filter:**

%% read the image from the file system to Matrix I

I = imread('cameraman.tif');

%% create a new figure to show the image .

figure(1);

%% show the loaded image.

imshow(I);

%% apply median filter using the internal matlab function medfilt2 .

Y=medfilt2(I,[5 5]);

figure(2);

%% show image after applying the filter

imshow(Y);

**Coding for Wiener Filter:**

%%%% Median Filter %%%%

%% read the image from the file system to Matrix I

I = imread('cameraman.tif');

%% create a new figure to show the image .

figure(1);

%% show the loaded image.

imshow(I);

%% apply median filter using the internal matlab function medfilt2 .

Y=medfilt2(I,[5 5]);

figure(2);

%% show image after applying the filter

imshow(Y);

**Coding for averaging filter:**

%%applying averaging filter with gaussian noise

newimg=imread('cameraman.png');

%% Show current image.

imshow(I);

figure;

G=imnoise(I,'gaussian',0.0005,0.0019);

imshow(G);

figure;

h = fspecial('average', 3);

F=imfilter(G,h);

imshow(F);

**Coding for Laplacian filter:**

%%laplacian filter

I = imread('cameraman.tif');

imshow(I);

figure;

H = fspecial('motion',20,45);

MotionBlur = imfilter(I,H,'replicate');

imshow(MotionBlur);

figure;

H = fspecial('disk',10);

blurred = imfilter(I,H,'replicate');

imshow(blurred);

**Module-4** included implementation of non-local mean filter using cluster tree mechanism

Coding :

function [output]=NLmeansfilter (input,t,f,h)

% input: image to be filtered

% t: radio of search window

% f: radio of similarity window

% h: degree of filtering

% Size of the image

[m n]=size (input);

% Memory for the output

Output=zeros (m,n);

% Replicate the boundaries of the input image

input2 = padarray (input, [f f],'symmetric');

% used kernel

kernel = make\_kernel(f);

kernel = kernel / sum (sum(kernel));

h=h\*h;

for i=1:m

for j=1:n

i1 = i+ f;

j1 = j+ f;

W1= input2(i1-f:i1+f , j1-f:j1+f);

wmax=0;

average=0;

sweight=0;

rmin = max(i1-t,f+1);

rmax = min(i1+t,m+f);

smin = max(j1-t,f+1);

smax = min(j1+t,n+f);

for r=rmin:1:rmax

for s=smin:1:smax

if(r==i1 && s==j1) continue; end;

W2= input2(r-f:r+f , s-f:s+f);

d = sum(sum(kernel.\*(W1-W2).\*(W1-W2)));

w=exp(-d/h);

if w>wmax

wmax=w;

end

sweight = sweight + w;

average = average + w\*input2(r,s);

end

end

average = average + wmax\*input2(i1,j1);

sweight = sweight + wmax;

if sweight > 0

output(i,j) = average / sweight;

else

output(i,j) = input(i,j);

end

end

end

function [kernel] = make\_kernel(f)

kernel=zeros(2\*f+1,2\*f+1);

for d=1:f

value= 1 / (2\*d+1)^2 ;

for i=-d:d

for j=-d:d

kernel(f+1-i,f+1-j)= kernel(f+1-i,f+1-j) + value ;

end

end

end

kernel = kernel ./ f;

Demo.m program:

clear

clc

clf

colormap(gray)

% create example image ima1

I=imread('ima2.jpg');

imshow(I);

ima=100\*ones(100);

ima(50:100,:)=50;

ima(:,50:100)=2\*ima(:,50:100);

fs=fspecial('average');

ima=imfilter(ima,fs,'symmetric');

% add some noise

sigma=10;

rima=ima+sigma\*randn(size(ima));

% show it

imagesc(rima)

drawnow

% denoise it

fima=NLmeansfilter(ima,5,2,sigma);

% show results

clf

subplot(2,2,1),imagesc(ima),title('original');

subplot(2,2,2),imagesc(rima),title('noisy');

subplot(2,2,3),imagesc(fima),title('filtered');

subplot(2,2,4),imagesc(rima-fima),title('residuals');

The performance of the above specified filters is compared using PSNR and MSE values. The coding for calculating this value is explained below.

% Demo to calculate MSE and PSNR of a gray scale image.

% Clean up.

clc; % Clear the command window.

close all; % Close all figures

clear; % Erase all existing variables.

workspace; % Make sure the workspace panel is showing.

format long g;

format compact;

fontSize = 20;

%------ GET DEMO IMAGES ----------------------------------------------------------

% Read in a standard MATLAB gray scale demo image.

grayImage = imread('cameraman.tif');

[rows columns] = size(grayImage);

% Display the first image.

subplot(2, 2, 1);

imshow(grayImage, []);

title('Original Gray Scale Image', 'FontSize', fontSize);

set(gcf, 'Position', get(0,'Screensize')); % Maximize figure.

% Get a second image by adding noise to the first image.

noisyImage = imnoise(grayImage, 'gaussian', 0, 0.003);

% Display the second image.

subplot(2, 2, 2);

imshow(noisyImage, []);

title('Noisy Image', 'FontSize', fontSize);

%------ PSNR CALCULATION ----------------------------------------------------------

% Now we have our two images and we can calculate the PSNR.

% First, calculate the "square error" image.

% Make sure they're cast to floating point so that we can get negative differences.

% Otherwise two uint8's that should subtract to give a negative number

% would get clipped to zero and not be negative.

squaredErrorImage = (double(grayImage) - double(noisyImage)) .^ 2;

% Display the squared error image.

subplot(2, 2, 3);

imshow(squaredErrorImage, []);

title('Squared Error Image', 'FontSize', fontSize);

% Sum the Squared Image and divide by the number of elements

% to get the Mean Squared Error. It will be a scalar (a single number).

mse = sum(sum(squaredErrorImage)) / (rows \* columns);

% Calculate PSNR (Peak Signal to Noise Ratio) from the MSE according to the formula.

PSNR = 10 \* log10( 256^2 / mse);

% Alert user of the answer.

message = sprintf('The mean square error is %.2f.\nThe PSNR = %.2f', mse, PSNR);

msgbox(message);

**4.3 TOOLS AND TECHNIQUES USED**

This project was implemented completely using Matlab tool developed Mathworks. Matlab is widely used in academic and research institutions as well as industrial enterprises. Operations such as Image enhancement, segmentation, analysis, registration and geometric transformations. It also allows the implementation of algorithms, creation of user interfaces, and interfacing with programs written in other languages, including C, C++, Java, Fortran and Python.

A screenshot of the respective Matlab tool is given below:

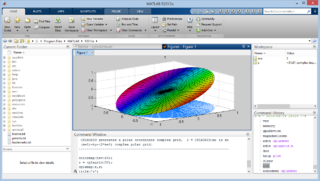


Fig 7.Matlab Environment

**4.4. TEST CASE SPECIFICATION**

**4.4.1 UNIT TEST-CASE SPECIFICATION**

The unit-test case **UTC001-Tests whether addition of the noise gives the output as noisy image for the input image.**

|  |  |
| --- | --- |
| **Test Case ID** | **UTC001** |
| **Test Case Description** | Tests whether addition of the noise gives the output as noisy image for the input image. |
| **Expected Output** | A noise added image is the expected output. |
|  |  |
| **Test Procedure** | |
| |  |  |  | | --- | --- | --- | | **Step** | **Input** | **Actual Output** | | **1.** | The captured input image is fed as the input for the code that adds noise to the image | An image containing noise is obtained as the result. The testcase hence passes as the expected and actual output matches with each other. | | |

**Table 4.1 UTC001**

**4.4.2 INTEGRATION TEST-CASE SPECIFICATION**

The integrated testcase **ITC001 – Test whether noise is successfully added to the image and simulates the final output by the application of proposed filter on it.**

|  |  |
| --- | --- |
| **Test Case ID** | **ITC001** |
| **Test Case Description** | Test whether noise is successfully added to the image and simulates the final output by the application of proposed filter on it. |
| **Expected Output** | The final output expected is a denoised image with better clarity and more detail preservation of texture patterns. |
| **Test Procedure** | |
| |  |  |  | | --- | --- | --- | | **Step** | **Input** | **Expected Output** | | **1.** | The obtained noisy image from the previous module is integrated and subjected to the proposed filter**.** | The final output obtained is the denoised image having better texture preservation. Thus, the testcase passes as both the expected and actual output matches with each other. | | |

**Table 4.2 ITC001**

**4.4.3 TESTDATA REQUIRED**

The testdata required or used while building the testcases in our project were the standard sample images, intermediate noisy images as well as the filtered images of respective sizes mxn. Final performance was measured using PSNR values and MSE values.

**CHAPTER-5**

**RESULTS AND DISCUSSIONS**

**5.1 PERFORMANCE METRICS USED**

The performance of the above proposed method is evaluated with the help of two statistical metric namely Peak-to-Signal-Noise-Ratio (PSNR),Mean Squared Error (MSE).Also, the speed of the method can be found by finding the computational time.

Here, MSE gives the cumulative square error between the encoded and the original image

PSNR gives the ratio of maximum possible value of a signal and the value of the noise.

Higher quality denoised image is indicated by the presence of smaller MSE values and higher PSNR values.

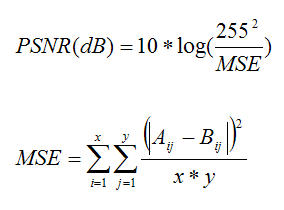


Fig.8 Mathematical formulae for PSNR and MSE

The above figure gives the mathematical formulae for the calculation of PSNR and MSE values.

**5.2 RESULTS**

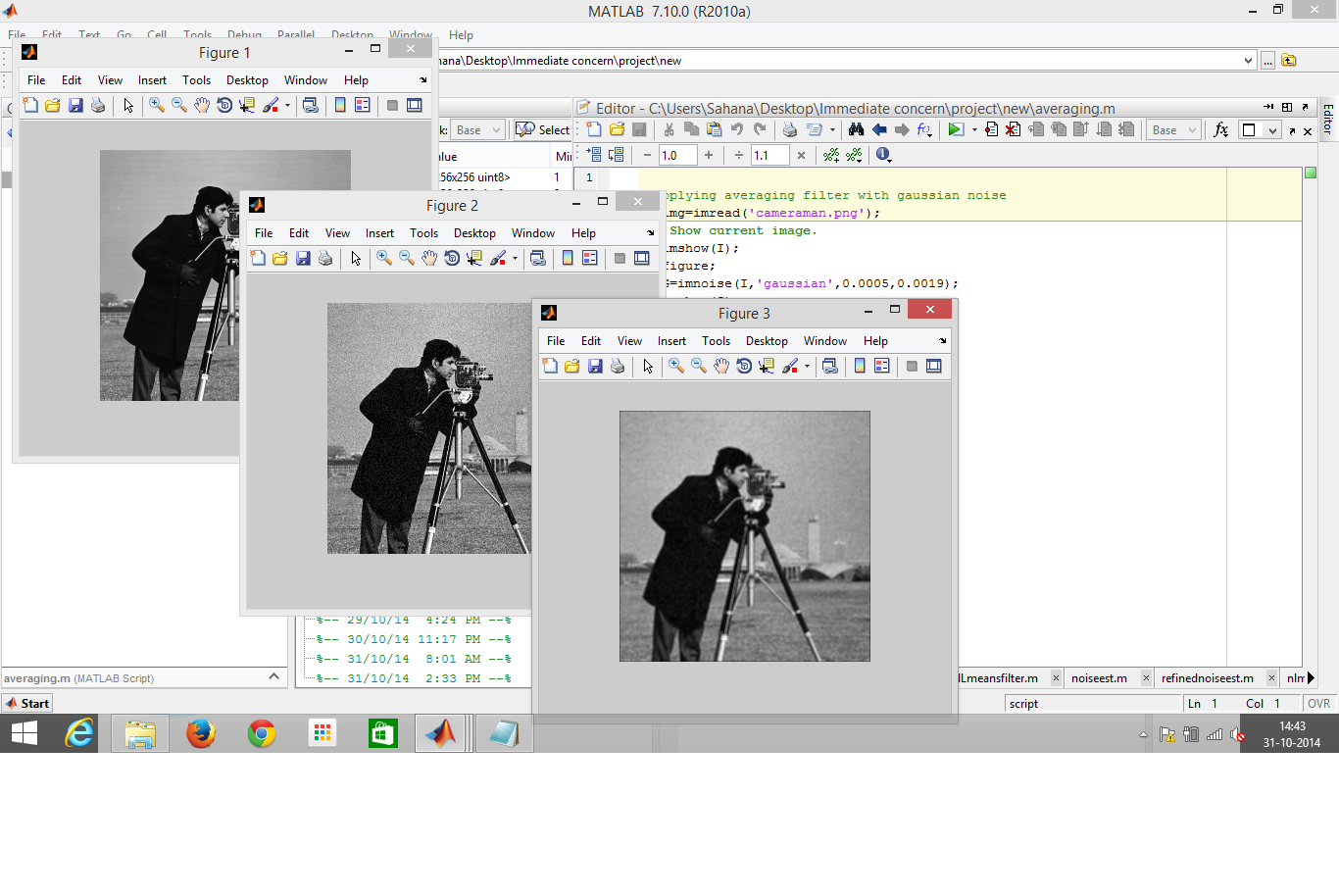
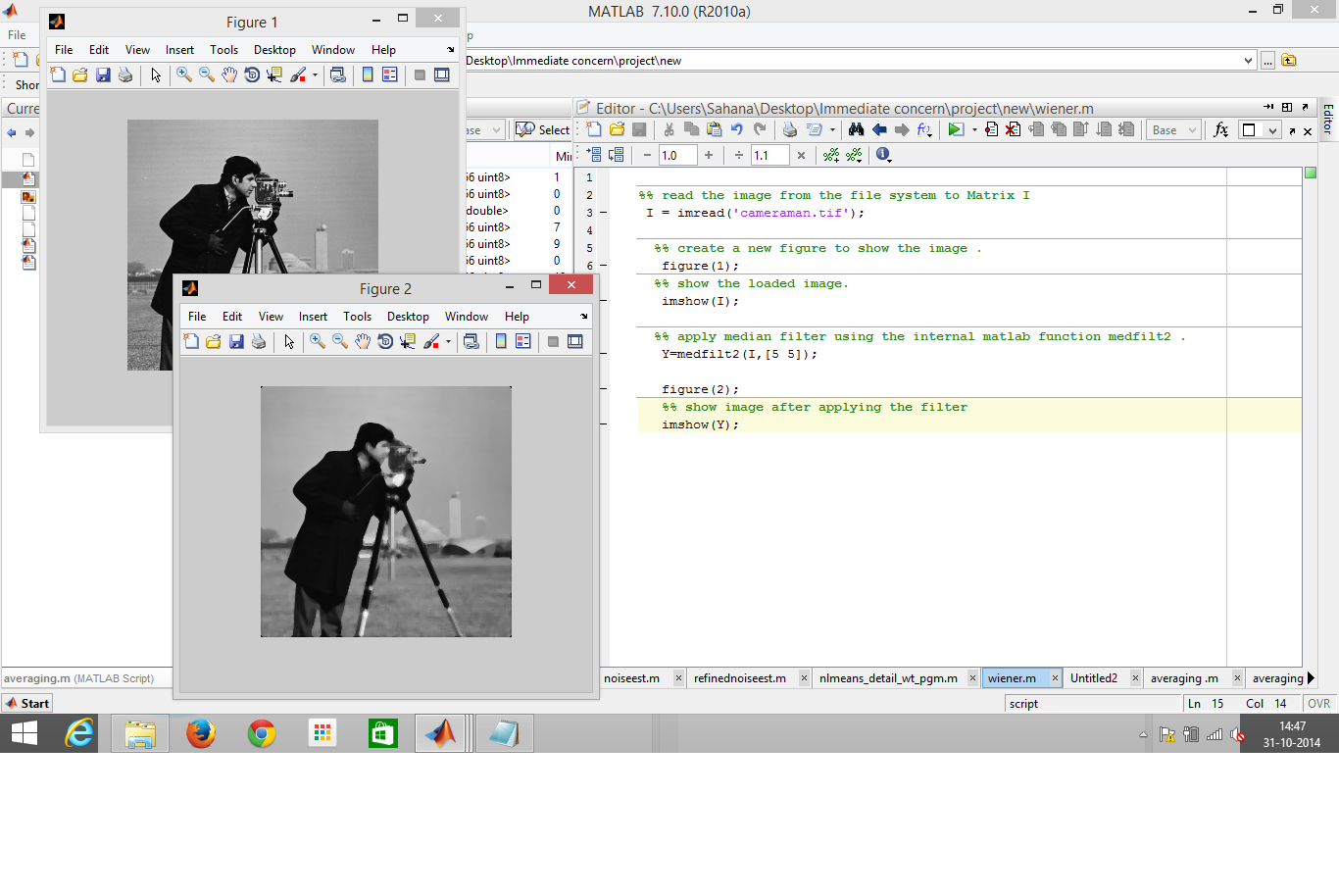
Based on the performance metrics for evaluating the results, we found out that application of non-local means following the cluster tree mechanism as the second fastest technique with less computational complexity and time. As expected, the non-local means did a better job of preserving edges than the other methods.

It was observed that even though the spatial sub-sampling method is faster than the cluster tree method, the computation came at a higher cost. This problem was overcome by the cluster tree method.

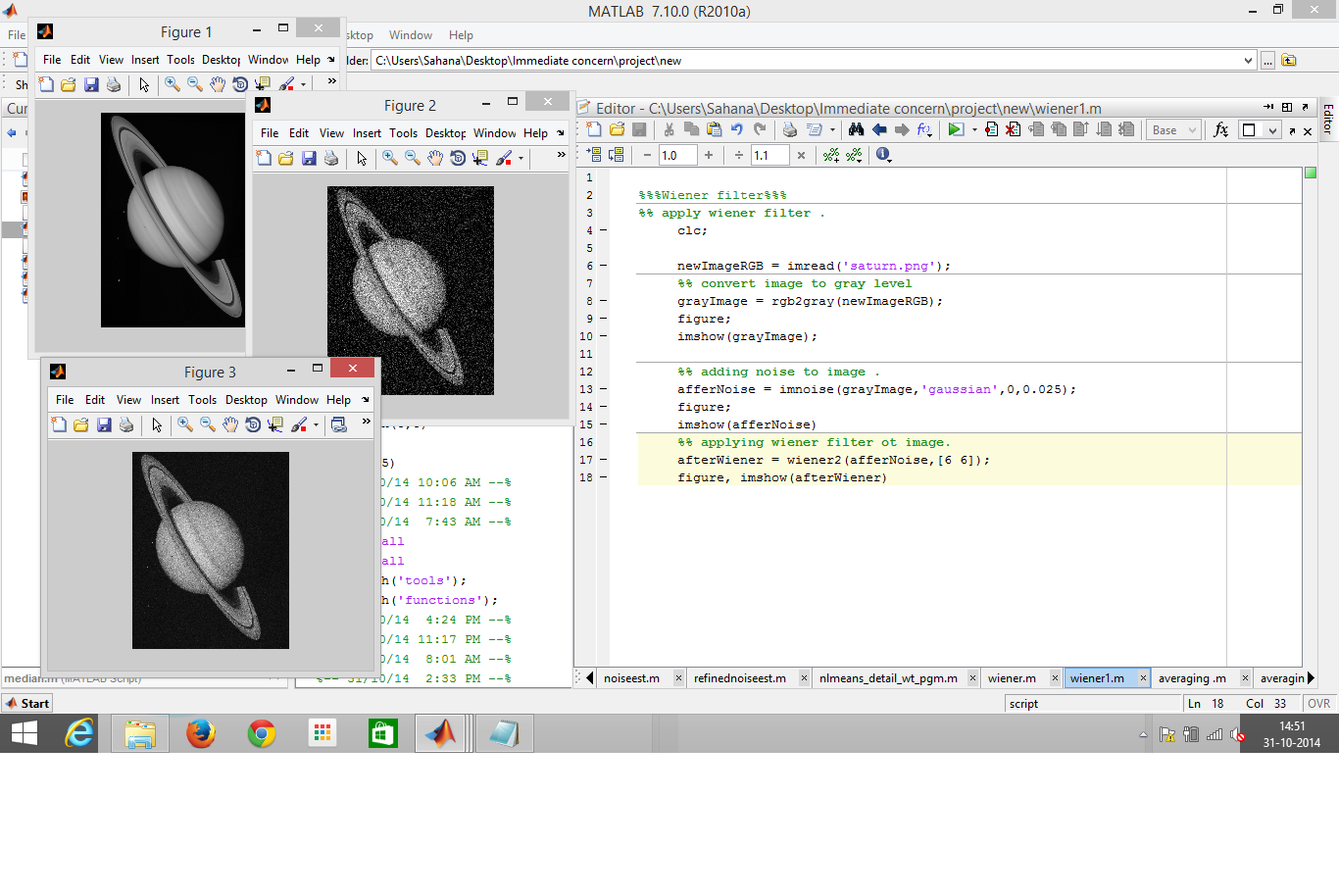
Advantages of cluster tree method over the other existing methods are:

* It did not cause any blurring of the images
* They helped in handling large volume of data
* It outperformed the preselection of patches by mean and variance method
* Due to the sub-linear time complexity for computation, the speedup in case is highly significant, especially, when the search window size increases

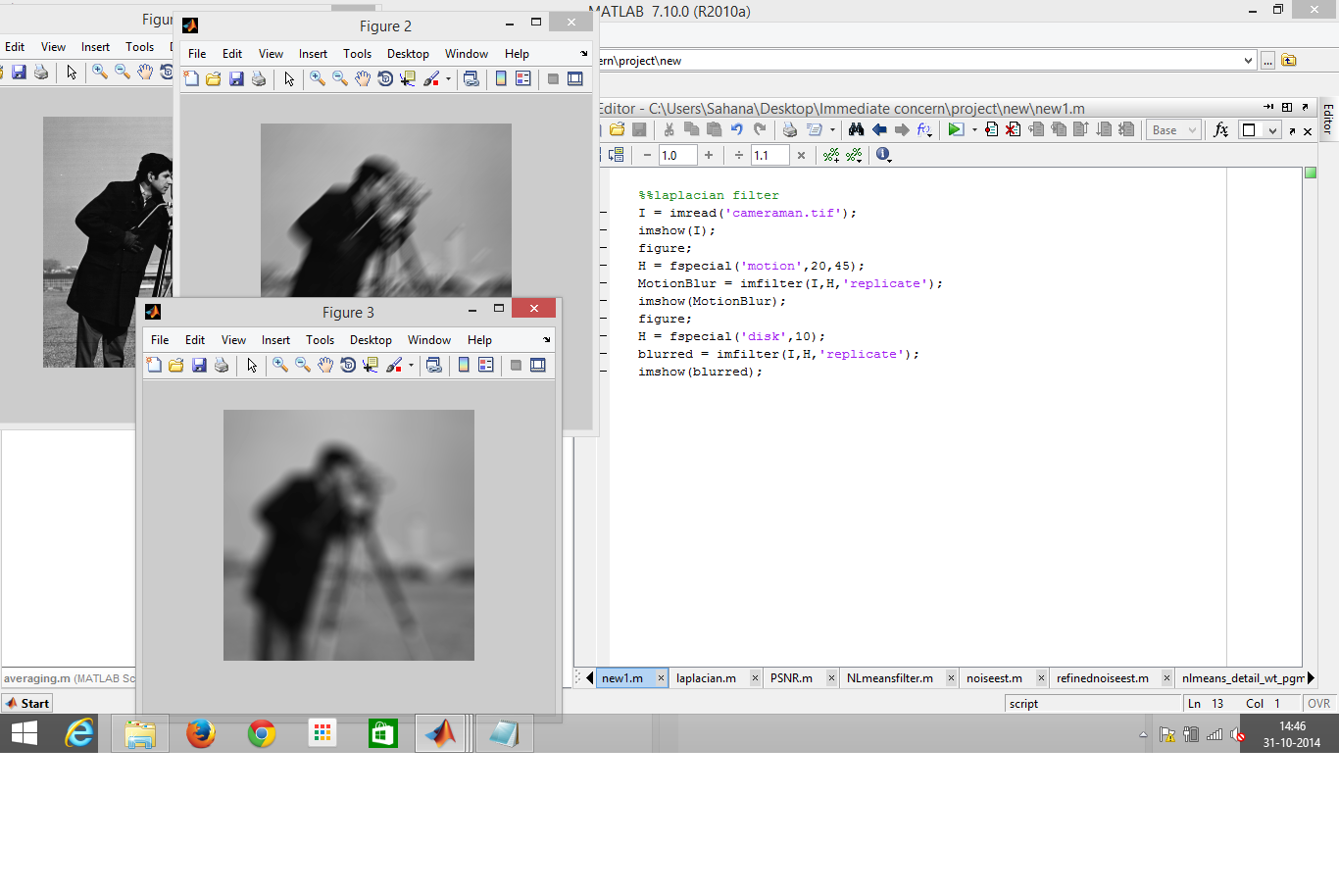
**5.3 SCREENSHOTS/PERFORMANCE RESULT TABLES**

**1)Averaging2)Median Filter**

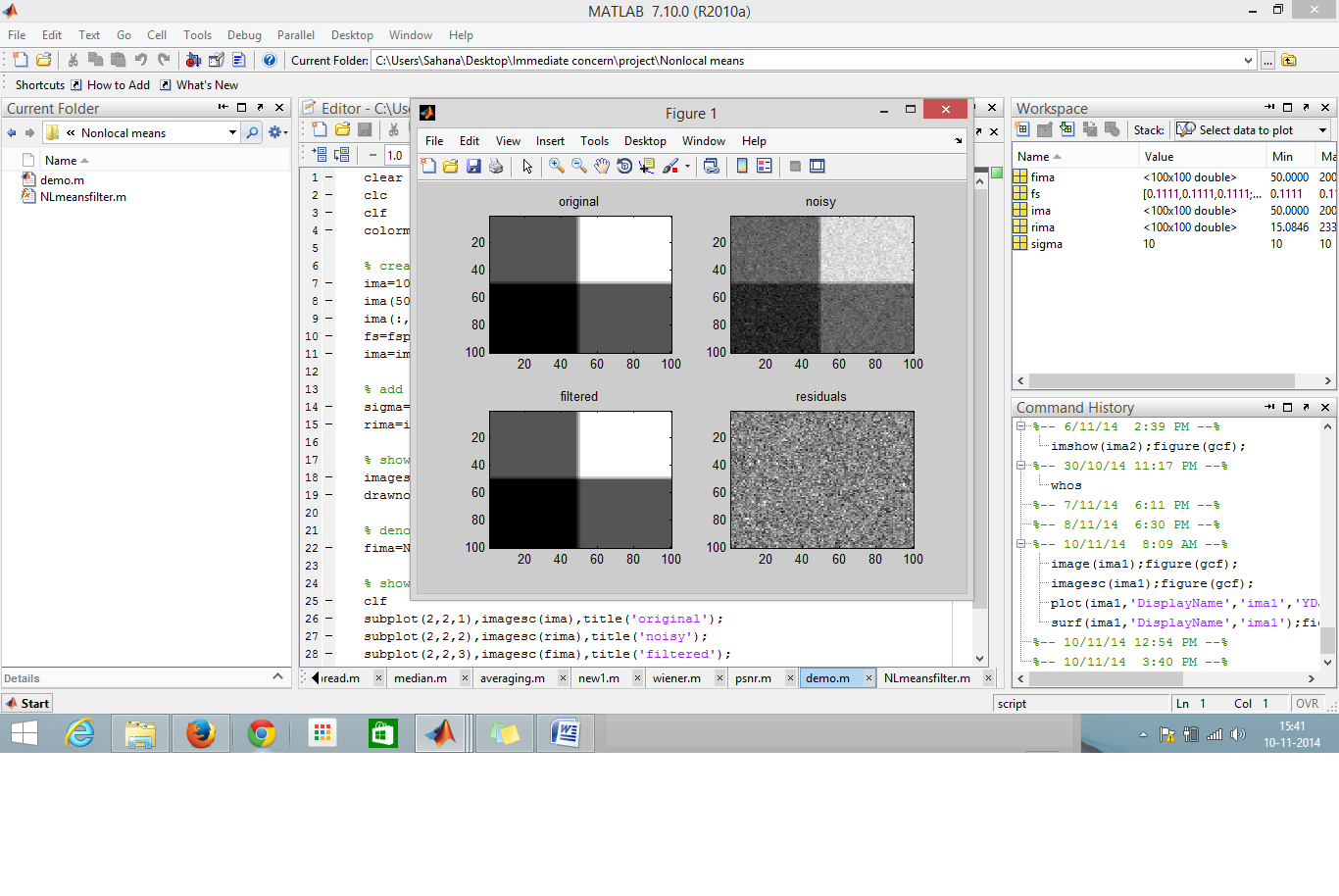
**3)Wiener Filter**

****

**4)Laplacian Filter**

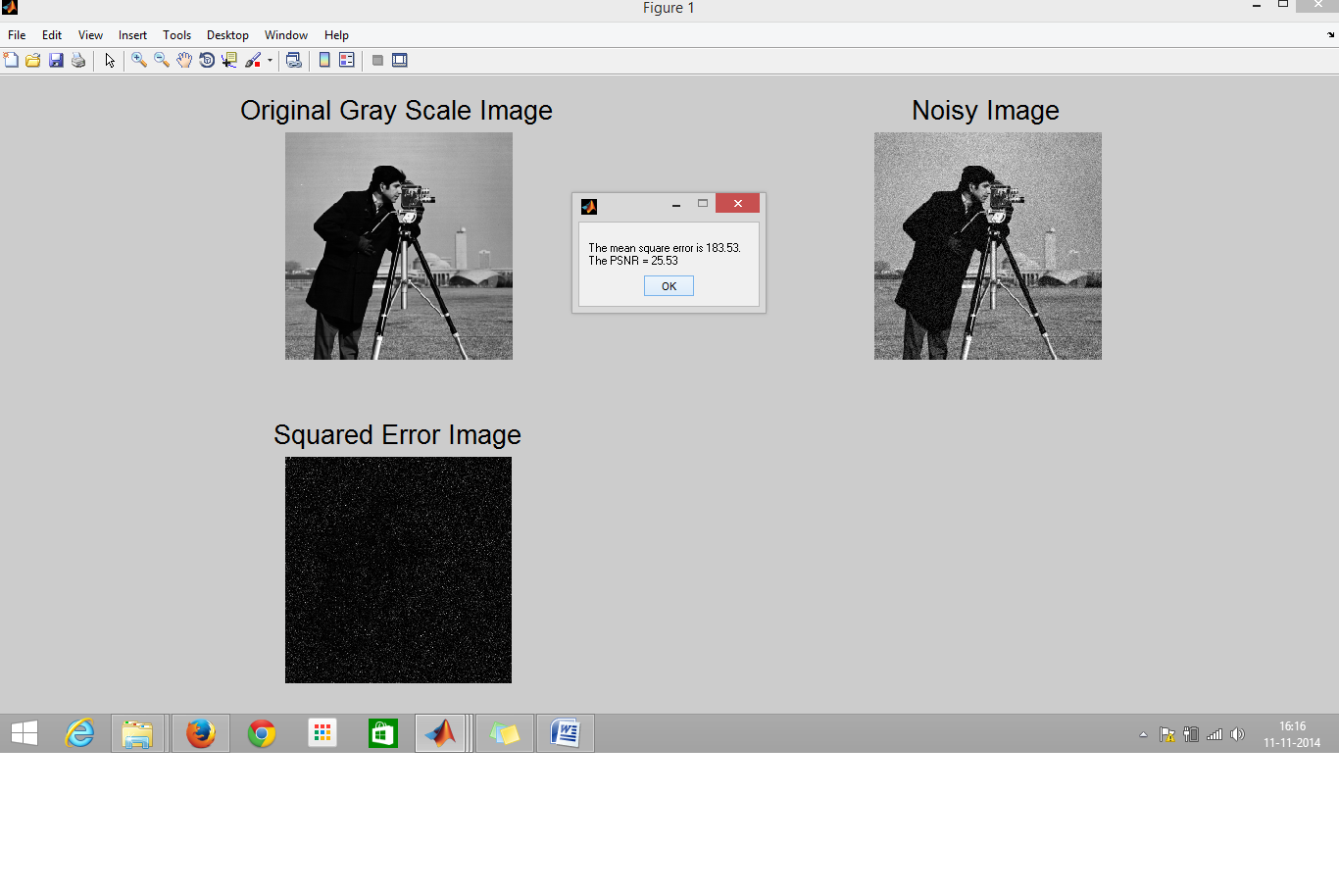
****

**NONLOCAL MEANS-DENOISING:**

****

**PSNR AND MSE VALUE CALCULATION :**

1.Calculation of PSNR and MSE using original and noisy( Gaussian) image

**TABLE 5.1- PSNR AND MSE VALUES FOR NOISES**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S.NO** | **IMAGE** | **NOISE** | **MSE** | **PSNR** |
| 1 | Cameraman.tif | Gaussian | 183.53 | 25.53 |
| 2. | Biometric.jpg | Gaussian | 2607.69 | 14.00 |
| 3. | Original.png | Salt&Pepper | 778.25 | 807.76 |
| 790.01 | 19.25 |
| 19.09 | 19.19 |
| 4. | Original.png | Speckle noise | 383.41 | 792.92 |
| 385.41 | 19.15 |
| 22.32 | 19.24 |

**CHAPTER-6**

**CONCLUSION AND FUTURE WORK**

**6.1 CONCLUSION**

The project study gave rise to the following conclusions.

Denoising method for texture patterns have been already done using several approaches such as non-local means. Through this project, we tried to explain the shortcomings associated with standard non-local means. To overcome their shortcomings, a better method was used where the similar patches were selected and arranged in the form of a cluster tree.

This project also helped to learn more about the various types of noises and possible filters that can be used to denoise them.

The denoised image obtained using Gaussian filters were found to be of poor quality in terms of texture preservation as expected. Also, the computational complexity and MSE values were found to be maximum, giving a blurred image on comparison with the remaining methods. Weiner filter performed better than Gaussian filter but still gave a blurred image as the result. Standard Non-local means method performed significantly well in comparison with Gaussian and Weiner filter. The MSE values also showed that they performed well but the computational complexity in terms of time was found to be more. Similarly, computational complexity in case of Median and standard non-local means was also found to be more.

The non-local mean filter implemented using cluster tree mechanism was found to perform significantly better within less time i.e, with lesser computational complexity and lowest MSE values thereby forming an ideal method with respect to the other methods. Also, it was found that the size of the search window did not affect the computational complexity unlike the other mechanisms.

**6.2 FUTURE WORK**

I would like to improve the corresponding project by extending it and can be carried out as a future work. Some of the improvements would be:

* The code can be improved more so that the performance can be increased. Even though, it performs better than few of its previous filters, the code cannot completely satisfy the requirements of the today’s changing digital technology.
* Also, the code can be implemented using some other programming languages such as C++ or C# so that it can be run on all platforms and optimal performance results with more reduced computational complexity can be obtained.

**APPENDIX**

**APPENDIX 1:**

FREQUENCY FILTER

Frequency filters process an image in the frequency domain. The image is http://homepages.inf.ed.ac.uk/rbf/HIPR2/mote.gifFourier transformed, multiplied with the filter function and then re-transformed into the spatial domain. Attenuating high frequencies results in a smoother image in the spatial domain, attenuating low frequencies enhances the edges.

All frequency filters can also be implemented in the spatial domain and, if there exists a simple kernel for the desired filter effect, it is computationally less expensive to perform the filtering in the spatial domain. Frequency filtering is more appropriate if no straightforward kernel can be found in the spatial domain, and may also be more efficient.

UNSHARP FILTER

The unsharp filter is a simple http://homepages.inf.ed.ac.uk/rbf/HIPR2/mote.gifsharpening operator which derives its name from the fact that it http://homepages.inf.ed.ac.uk/rbf/HIPR2/mote.gifenhances edges (and other high frequency components in an image) via a procedure which subtracts an unsharp, or smoothed, version of an image from the original image. The unsharp filtering technique is commonly used in the photographic and printing industries for crispening edges. The unsharp filter is implemented as a window-based operator.

MEDIAN FILTER

Median filter is a best order static, non- linear filter, whose response is based on the ranking of pixel values contained in the filter region. Median filter is quite popular for reducing certain types of noise. Here, the center value of the pixel is replaced by the median of the pixel values under the filter region

**REFERENCES**

[1]Avindra Fernando University of Kansas (2010) avindra@ku.edu,

“Image Denoising with the Non-local Means Algorithm”

[2] Mona Mahmoudi and Guillermo Sapiro (2009), “ Fast Image and Video Denoising via non-local means of similar neighbourhoods”

[3] Laurent Condat, Member, IEEE, (2005) “A Simple Trick to Speed up the Non-Local Means”,

[4] Thomas Brox, Oliver Kleinschmidt, and Daniel Crème, (2008) ” Efficient Nonlocal Means for Denoising of Textural Patterns”,IEEE Transcations on Image Processing ,Vol.17,No.7,July 1083

[5] T. Brox and D. Cremers,(2007) “Iterated nonlocal means for texture restoration,”in *Scale Space and Variational Methods in Computer Vision*, F. Sgallari, A. Murli, and N. Paragios,Eds. New York: Springer,vol. 4485, pp. 13–24.

[6] T. Brox and D. Cremers, (2007)” On local region models and the statistical

interpretation of the piecewise smooth mumford”-shah functional supplementary

online material [Online].

[7] A. Buades, B. Coll, and J.M. Morel, (2005) “A non-local algorithm for image

denoising,” in *Proc. Int. Conf. Computer Vision and Pattern Recognition*, pp. 60–65.

[8] A. Buades, B. Coll, and J. M. Morel, (2005) “A review of image denoising

algorithms, with a new one,” *SIAM Interdisciplanary J.*, vol. 4, no. 2,

pp. 490–530.

[9] V.Karnati, M. Uliyar, and S. Dey,(2009) “Fast non-local algorithm for image denoising,” in *Proc. of IEEE ICIP*, Cairo, Egypt.

[10] A. Efros and T. Leung. (1999) “Texture synthesis by nonparametric

sampling.” *In Proc. Int. Conf. Computer Vision, volume 2, pages 1033-1038*.

[11]Mathworks-The Matlab Image proessing toolbox http://www.mathworks.com/access/helpdesk/help/toolbox/images/

[12] Antoni Buades, Bartomeu Coll and Jean-Michel Morel, (2005) “A non-local algorithm for image Denoising**”**