TRIBHUVAN UNIVERSITY

INSTITUTE OF ENGINEERING

PULCHOWK CAMPUS

THESIS NO: 072/MSI/613

Genetic Algorithm Based Approach to Image Denoising Problem

by

Sanjay Bhandari

A THESIS

SUBMITTED TO THE DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF SCIENCE IN INFORMATION AND COMMUNICATION ENGINEERING

DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING

Lalitpur, Nepal

November, 2017

Genetic Algorithm Based Approach to Image Denoising Problem

By

Sanjay Bhandari

072/MSI/613

Thesis Supervisor

Daya Sagar Baral

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in Information and Communication Engineering

Department of Electronics and Computer Engineering
Institute of Engineering, Pulchowk Campus
Tribhuvan University
Lalitpur, Nepal

November, 2017

Head
Department of Electronics and Computer Engineering
Institute of Engineering, Pulchowk Campus
Pulchowk, Lalitpur, Nepal

Recommendation

The undersigned certify that they have read and recommended to the Department of Electronics and Computer Engineering for acceptance, a thesis entitled **"Genetic Algorithm Based Approach to Image Denoising Problem"**, submitted by **Sanjay Bhandari** in partial fulfillment of the requirement for the award of the degree of "**Master of Science in Information and Communication Engineering**".

....

Supervisor: Daya Sagar Baral

Department of Electronics and Computer Engineering

....

External Examiner: Dr. Pradip Paudyal

Assistant Director,

Nepal Telecommunications Authority

....

Committee Chairperson: Dr. Dibakar Raj Pant

Head of the Department,

Department of Electronics and Computer Engineering

Date: 29th November, 2017

Departmental Acceptance

The thesis entitled **"Genetic Algorithm Based Approach to Image Denoising Problem",** submitted by **Sanjay Bhandari** in partial fulfillment of the requirement for the award of the degree of "**Master of Science in Information and Communication Engineering**" has been accepted as a bonafide record of work independently carried out by him in the department.

Dr. Dibakar Raj Pant

Head of the Department

Department of Electronics and Computer Engineering,

Pulchowk Campus,

Institute of Engineering,

Tribhuvan University,

Nepal.

ACKNOWLEDGEMENT

I am indebted to my thesis supervisor, **Mr. Daya Sagar Baral**, Department of Electronics & Computer Engineering, for providing me constant coordinate and helpful hints from the beginning of this work to the very end.

I would like to thank **Dr. Dibakar Raj Pant**, Head of Electronics and Computer Engineering Department, for his precious support and guidance. I would like to express my sincere gratitude to **Dr. Sanjeeb Prasad Pandey**, Department of Electronics & Computer Engineering, whose guidance during this work was very helpful.

My special thanks to our program coordinator, **Dr. Basanta Joshi** for extraordinary coordination and support. I am grateful to all the faculty members of Department of Electronics & Computer Engineering for the kind co-operation and guidance.

My thank goes to all my friends and family members for providing me necessary help and moral support.

Finally, I express my thanks to all those who helped me directly or indirectly in course of this work.

ABSTRACT

Digital images can be degraded by noise during the process of acquisition, transmission, storage or compression. It is necessary to remove the noise in the image before the image is suitable for different processing operations. Image denoising is a process which is deployed to remove the noise through the manipulation of image data to recover quality image from the noisy image. The image denoising process should be such that the original image can be recovered without losing important features such as edges, corners and textures. One of the powerful and perspective approaches in this area is image denoising using discrete wavelet transform. This work combines genetic algorithm with wavelet based denoising methods. During the evolutionary process, wavelet based denoising methods are applied as local search operators and filtering techniques are applied as mutation operators. A set of digital images, commonly used by the scientific community as benchmarks, is contaminated by different level of additive Gaussian noise and the proposed algorithm is used to reduce the noise level in the image. The results in terms of PSNR & SSIM values obtained by the proposed method shows that application of genetic algorithm can improve the result obtained from wavelet based denoising methods. Also the proposed method is compared against denoising methods in the literature. On average it outperforms the compared methods in terms of PSNR & SSIM values

Keywords: Image denoising, Wavelet transform, Threshold, Feature preservation, Genetic Algorithm, Mutation

Table of Contents

List of Figures

List of Tables

List of Abbreviations

GA	Genetic Algorithm
TV	Total Variation
DWT	Discrete Wavelet Transform
MSE	Mean Square Error
PSNR	Peak Signal to Noise Ratio
SSIM	Structural Similarity Index
AWGN	Additive White Gaussian Noise
SPN	Salt and Pepper Noise
RVIN	Random-Valued Impulse Noise
SN	Speckle Noise
AD	Anisotropic Diffusion
BM3D	Block Matching & 3D filtering
HGA	Hybrid Genetic Algorithm

CHAPTER 1: INTRODUCTION

1.1. Background

The use of digital images has been rapidly increased in the applications of digital world such as Digital cameras, Satellite Television, Magnetic Resonance Imaging (MRI), Geographical Information System (GIS) etc. Image information i.e. information transmitted in the form of digital images, is one of the recent trends. One of the most interesting aspects of this information revolution is the ability to send and receive complex data that are beyond the capability of ordinary written text. Image processing is one form of signal processing for which the input is an image i.e. photographs or frames of video and the output can be either an image or a set of characteristics or parameters related to the image. The majority of image processing techniques involve treating the image as a two-dimensional signal and applying standard signal processing techniques to it.

Generally, data sets collected by image sensors are contaminated by noise. Imperfect instruments, problems with data acquisition process, and interfering natural phenomena can all corrupt the data of interest. Various types of noise present in image are Gaussian noise, Salt &Pepper noise and Speckle noise. Image denoising is necessary to obtain best approximation of the original digital image from the received noisy image. Image denoising techniques are used to suppress these types of noises while retaining the important image features such as corners & edges. Noise suppression can introduce artifacts or cause image blurring, which makes image denoising a complex task. Several approaches have been proposed to remove noise in digital images; however, each one explores specific aspects of the problem.

Spatial filters like mean and median filter are used to remove the noise from image. But the disadvantage of spatial filters is that these filters not only smooth the data to reduce noise but also blur edges in image. Therefore, Wavelet Transform is used to preserve the edges of image. It is a powerful tool of signal or image processing for its multi-resolution possibilities.

For this work, Genetic Algorithm (GA) is proposed for image denoising that integrates GA with image denoising method. The method evolves the images that are restorations of noisy images. Mutation is applied over such noisy images to create an initial population. The population is evolved for certain time and the best found individual is returned as restored image.

1.2. Problem Statement

Noise can get added in images during acquisition or during transmission. The main aim of an image denoising algorithm is to reduce the noise level, while preserving the image features.

The multi resolution analysis performed by the wavelet transform has been shown to be a powerful tool to achieve these goals. In wavelet domain, the noise is uniformly spread throughout the coefficients, while most of the image information is concentrated in the few largest coefficients. The most straightforward way of distinguishing information from noise in the wavelet domain consists of thresholding the wavelet coefficients.

GA in image denoising evolves images that are restorations of noisy images. Combination of GA with denoising methods can bring significant gain as compared to using GA only. In this research the GA is proposed to be combined with wavelet denoising methods.

1.3. Objectives

a) To denoise the noisy image by wavelet thresholding method.

b) To improve the image obtained in (a) by application of Genetic Algorithm.

CHAPTER 2: LITERATURE REVIEW

The most common techniques for performing image denoising are based on filters that smooth the images in order to suppress noise. However, these techniques in general also degrade important features of the images, such as edges, corners and texture. The filters used to suppress noise in images are classified as linear and non-linear filters. Linear filters can be expressed as a convolution of a kernel (filter) through a noisy image to produce the resulting image. On the other hand, any filter that cannot be represented as a convolution operation is a non-linear filter. A linear filter that is widely used for image denoising is the Wiener filter, which works by minimizing the mean squared error between the recovered and the original images. On the other hand, a commonly used non-linear filter is the median filter, which replaces the value of each pixel by the median value of neighbourhood pixels [3].

There are other techniques that aim at removing as much noise as possible, trying to preserve important features of the images. The total variation (TV) methods consider that the noisy signals in an image have high total variation and perform the denoising process by minimizing these signals [4–6]. Methods such as anisotropic and isotropic diffusion, on the other hand, use a function to identify the edges present in an image. These techniques diffuse the image continuously, smoothing it in the process, but they are able to identify when to stop the diffusion process through this edge-aware function. Therefore, they can produce an image that is smoothed and preserve its edges [7, 8].

An example of anisotropic diffusion method is presented by Black et al. [9]. This method assumes that the noisy image is a piecewise defined function that has been corrupted by a Gaussian noise with zero mean and a small variance. Moreover, it is also assumed that the difference between a pixel and its neighbours must be small and follow a normal distribution with zero mean. When the difference between a pixel and its neighbours does not fit in this pattern, it must be an edge region. Based on these assumptions and using statistical analysis, they were able to create a new edge stopping function that makes it possible to smooth the image without suppressing relevant information about edges.

Many denoising methods operate in the frequency domain, where techniques as Fourier or wavelet transforms are used, such that an image is represented by its frequencies instead of being represented by a spatial function (f(x, y)). The BM3D [10] is one of these methods, which uses sliding windows to run through the image and create blocks in a first step. In the second step, similar blocks are stacked together and transformed to the frequency domain. The blocks are filtered in a third step by an adapted Wiener filter and, finally, the restored image is constructed by weighing the values of the blocks that were grouped together.

Some of the most effective image denoising techniques rely on wavelet transforms. A common approach consists in searching for thresholds that limit the wavelet coefficients linked to the noisy frequencies. This process, commonly called wavelet shrinkage, is basically composed of three phases: (i) transform the image to the wavelet domain; (ii) estimate the thresholds and suppress the noise through a shrinkage rule; (iii) perform the inverse transformation and, therefore, retrieve the restored image [17, 11].

A method based on the concepts of wavelet shrinkage is introduced in [11]. First, the image is divided into a set of blocks that are transformed to the wavelet domain. Next, an edge detection algorithm is applied and the thresholds for the sub-band are estimated. Then, the wavelet coefficients have their threshold limited adaptively regarding their sub-bands. After this step, a shrinkage rule is applied to identify and suppress the noisy coefficients in the image. Finally, the inverse transform is performed on the blocks and the restored image is reconstructed. A different approach to wavelet shrinkage was proposed by Ghael et al. [13]. In this technique, a wavelet shrinkage estimate is used to create a Wiener filter in the wavelet domain. Due to the fact that the filter is specially designed and takes into account the wavelet coefficients of the image, the technique becomes able to produce high quality outputs.

Most of the image denoising techniques consider a model where an image was corrupted by an additive white Gaussian noise. Other methods have been specially proposed to suppress non-Gaussian or non-additive noise. The work by Deledalle et al. [14] shows a non-local mean method where, instead of simply calculating an Euclidean distance to define the averages of similar pixels, it provides statistical basis to define a weighted maximum likelihood estimator. This estimator takes into account the distribution of the noise, reaching impressive results for SAR images corrupted with a multiplicative speckle noise.

Ishikawa [15] considers an image as a Markov Random Field (MRF), which is described as an undirected graph that represents a set of random variables (vertices). In this case, the pixels are the vertices and the edges in the graph are the neighbourhood relationship between pixels. Each vertex of the graph can assume a range of values L, where the possibility of a vertex assuming a determined value is given by a probability function P(X), with X being a state of the graph. The image denoising problem is modelled as a minimum cut problem, where it is expected that the cost of the graph cut is the same as the energy function of the MRF. Therefore, minimizing the energy function could be considered as finding the minimum cut of the graph [15].

In the literature, there are also different evolutionary approaches to dealing with noisy images. Some of these methods are used to estimate the thresholds to perform wavelet shrinkage, as described in [16] that applied a Differential Evolution Strategy. The authors in [12] find threshold values using a Multi-Objective Genetic Algorithm.

A genetic algorithm is proposed by [1] to perform image denoising, where the images are individuals and a population evolves applying tailor-made crossover and mutation operators. The new individuals are created by crossover operators exchanging pieces of images, while the mutation operators are simple filters such as averaging filters, median filters and Gaussian filters applied over the images.

CHAPTER 3: RELATED THEORY

3.1. Noise in Images

In this section various types of noise corrupting an image signal are studied. The sources of noise are discussed and mathematical models for the different types of noise are presented.

3.1.1. Sources of Noise

During acquisition, transmission, storage and retrieval processes an image signal gets contaminated with noise. Acquisition noise is usually Additive White Gaussian Noise (AWGN) with very low variance. In many engineering applications, the acquisition noise is quite negligible. It is mainly due to very high quality sensors. In some applications like remote sensing, biomedical instrumentation, etc., the acquisition noise may be high enough. But in such a system, it is basically due to the fact that the image acquisition system itself comprises of a transmission channel. Hence the researchers are mainly concerned with the noise in a transmission system; usually the transmission channel is linear but dispersive due to a limited bandwidth. The image signal may be transmitted either in the analog form or in digital form.

When an analog image signal is transmitted through a linear dispersive channel, the image edges get blurred and image signal gets contaminated with AWGN since no channel is noise free. The noise introduced in the transmission channel of a communication system will be considered in analog form. If the channel is so poor that the noise variances is high enough and make the signal excursive to very high positive or high negative value, the thresholding operation which is done at the front end of the receiver will contribute to saturated maximum and minimum values. Such noisy pixels will be seen as white and black spots. Therefore this type of noise is known as Salt and Pepper Noise (SPN). If analog image signal is transmitted the signal gets corrupted with AWGN and SPN as well. Thus there is an effect of mixed noise. If the image signal is transmitted in digital form through a linear dispersive channel, then a noise is introduced due to Bit Error called Inter Symbol Interference (ISI) which takes place along with AGWN which makes the situation worse. Due to ISI and AWGN, it may happen that 1 may be recognized as 0 and vice versa. Under such circumstance, the image pixel values have changed to some random values at random positions in the image frame. Such type of noise is known as Random-Valued Impulse Noise (RVIN).

3.2. Mathematical Representation of Noise

The mathematical representation of AWGN, SPN and SN are discussed in this section.

3.2.1. Gaussian Noise

It is evenly distributing over the signal. This means that each pixel in the noisy image is the sum of true pixel values and random Gaussian distributed noise value is given by,

$$n_{AWGN}(t) = \eta_G(t) \tag{3.1}$$

$$f_{AWGN} = f(x,y) + \eta_G(x,y) \tag{3.2}$$

where η_G a random variable which has a Gaussian probability distribution with bell shaped probability distribution function given by

$$F(g) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(g-m)^2/2\sigma^2} \tag{3.3}$$

where g represents the gray level, m is the average or mean of the function and σ is standard deviation of the noise.

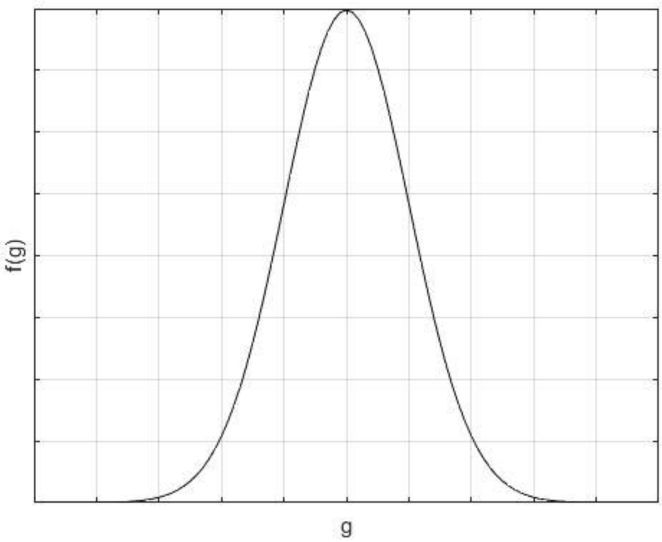


Figure 3.1: Representation of Additive White Gaussian Noise Distribution

Graphically it is represented as shown in Figure 3.1. In Equation 3.2 the noisy image is represented as the sum of original uncorrupted image and Gaussian distributed random noise η_G . When the variance of the random noise η_G is very low, η_G(x,y) is zero or very close to zero with many pixel locations. Under such circumstances the noisy image f_{AWGN} is same or very close to original image at many pixel location (x,y).

3.2.2. Salt & Pepper Noise

SPN is caused generally due to error in data transmission. It has only two possible values a and b. The probability of each is less than 0.1. The corrupted pixels are set alternatively to the minimum or to the maximum value, giving the image a "salt and pepper" like appearance. Unaffected pixels remain unchanged. For an 8-bit image, the typical value for pepper noise is 0 and for salt noise 255. The salt and pepper noise is generally caused by malfunctioning of pixel elements in the camera sensors, faulty memory locations, or timing errors in the digitization process. The probability density function for this type of noise is shown in Fig. 1.3. The impulse noise occurs at random locations (x,y) with a probability of d. The SPN and RVIN are substitute in nature. An image corrupted with RVIN of density d, f_{RVIN}(x,y)is mathematically represented as

$$f_{RVIN}\ (x, y) = \begin{cases} f(x, y) & \text{with probability } p = 1 - d \\ \eta(x, y) & \text{with probability } p = d \end{cases} \tag{3.4}$$

Here η(x,y) represents a uniformly distributed random variable, ranging from 0 to 1 that replaces the original pixel value f (x, y) . The noise magnitude at any noisy pixel location (x, y) is independent of the original pixel magnitude.

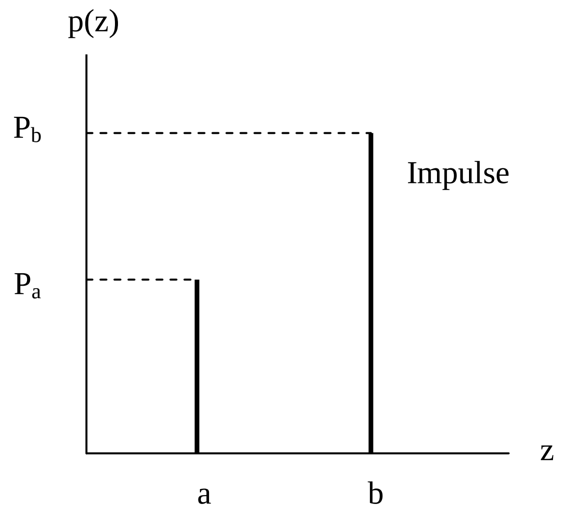


Figure 3.2: Representations of Salt and Pepper Noise Distribution

3.2.3. Speckle Noises

This type of noise occurs in almost all coherent imaging systems such as laser, acoustics, SAR (Synthetic Aperture Radar) in bio medical applications like ultrasonic imaging. The SN is a signal dependent noise i.e., if the image pixel magnitude is high, then the noise is also high. Therefore it is also known as multiplicative noise and is given using Equation 3.5 & 3.6.

$$\eta_{SN}(t) = \eta(t).s(t) \tag{3.5}$$

$$f_{SN}(x, y) = f(x, y) + \eta(x, y).f(x, y) \tag{3.6}$$

where $\eta(t)$ is a random variable and $s(t)$ is the magnitude of the signal. The noise is multiplicative since the imaging system transmits a signal to the object and the reflected signal is recorded. Speckle Noise follows a gamma distribution given using Equation (3.7).

$$F(g) = \frac{g^{\alpha-1}}{(\alpha-1)!\,a^{\alpha}} e^{-\frac{g}{a}} \tag{3.7}$$

where $a^2\alpha$ is variance and g is the gray level and is given below in Figure 3.3.

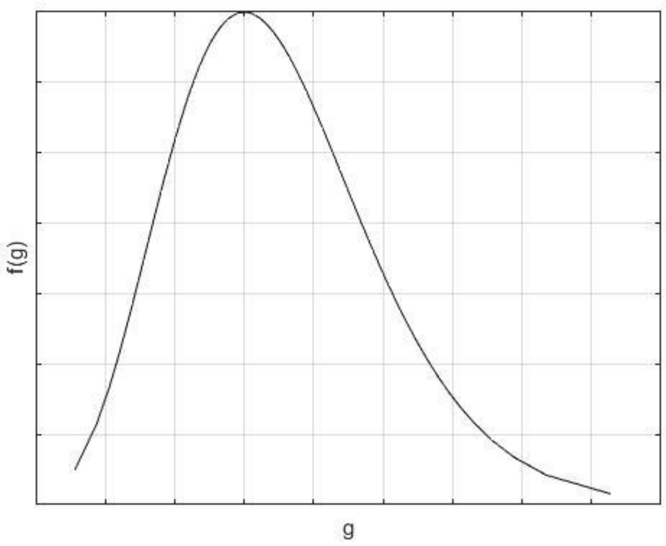


Figure 3.3: Representation of Speckle Noise Distribution

The speckle noise is encountered only in a few applications like ultrasonic imaging and SAR, whereas all other types namely AWGN, SPN and RVIN occur in almost all applications. The AWGN is the most common among all. In general, some combinations of AWGN, SPN and RVIN may represent a practical noise. Such type of noise is known as Mixed Noise.

3.3. Denoising

De-noising plays a important role in the field of the image pre-processing. It is often a necessary step to be taken, before the image data is analyzed. It attempts to remove whatever noise is present and retains the significant information, regardless of the frequency contents of the signal. De-noising has to be performed to recover the useful information. In this process much concentration is spent on, how well the edges are preserved and, how much of the noise granularity has been removed.

3.4. Classification of Denoising Methods

There are two basic approaches to image denoising, spatial domain filtering methods and transform domain filtering methods.

3.4.1. Spatial Domain Filtering Methods

A traditional way to remove noise from image data is to employ spatial filters. Spatial filters are further classified into linear filters and non linear filters.

3.4.1.1. Linear Filters

Most classical linear image processing techniques are based on the assumption that image processing applications in which both edge enhancement and noise reduction are desired, linear filters tend to blur sharp edges, destroy lines and other fine image details and perform poorly in the presence of signal dependent noise.

3.4.1.2. Non Linear Filters

Non linear filters modify the value of each pixel in an image based on the value returned by a non linear filtering function that depends on the neighbouring pixels. Non linear filters are mostly used for noise removal and edge detection. The traditional non linear filters are the median filter. Spatial filters employ a low pass filtering on groups of pixels with the assumption that the noise occupies the higher region of frequency spectrum. Generally spatial filters remove noise to a reasonable extent but at the cost of blurring images which in turn make the edges in pictures invisible.

3.4.2. Transform Domain Filtering Methods

The Transform Domain Filtering methods can be classified according to the choice of the basis or analysis function. The analysis functions can be further classified as Spatial Frequency Filtering and Wavelet domain

3.4.2.1. Spatial Frequency Filtering

Spatial Frequency Filtering refers to low pass filters using Fast Fourier Transform (FFT). In frequency smoothing methods the removal of the noise is achieved by designing a frequency domain filter and adapting a cut-off frequency to distinguish the noise components from the useful signal in the frequency domain. These methods are time consuming and depend on the cut-off frequency and the filter function behavior. Furthermore they may produce frequency artifacts in the processed image.

3.4.2.2. Wavelet Domain

Noise is usually concentrated in high frequency components of the signal which corresponds to small detail size when performing a wavelet analysis. Therefore removing some high frequency (small detail components) which may be distorted by noise is a denoising process in the wavelet domain. Filtering operations in wavelet domain can be categorized into wavelet thresholding, statistical wavelet coefficient model and undecimated wavelet domain transform based methods.

3.5. Wavelets

The concept of wavelet was hidden in the works of mathematicians even more than a century ago. In 1873, Karl Weirstrass mathematically described how a family of functions can be constructed by superimposing scaled versions of a given basis function. The term wavelet was originally used in the field of seismology to describe the disturbances that emanate and proceed outward from a sharp seismic impulse. Wavelet means a “small wave”. The smallness refers to the condition that the window function is of finite length compactly supported. A wave is an oscillating function of time or space and is periodic. In contrast, wavelets are localized waves. They have their energy concentrated in time and are suited to analysis of transient signals.

In wavelet analysis, the signal to be analyzed is multiplied with a wavelet function and then the transform is computed for each segment generated. The Wavelet Transform, at high frequencies, gives good time resolution and poor frequency resolution, while at low frequencies; the Wavelet Transform gives good frequency resolution and poor time resolution. An arbitrary signal can be analyzed in terms of scaling and translation of a single mother wavelet function (basis). Wavelets allow both time and frequency analysis of signals simultaneously because of the fact that the energy of wavelets is concentrated in time and still possesses the wave-like (periodic) characteristics. As a result, wavelet representation provides a versatile mathematical tool to analyze transient, time-variant (non stationary) signals that are not statistically predictable especially at the region of discontinuities -a feature that is typical of images having discontinuities at the edges.

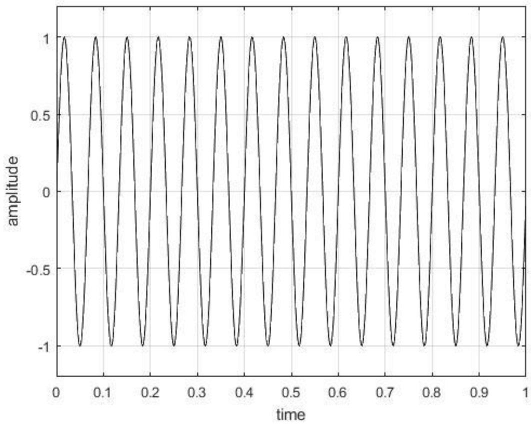


Figure 3.4: Wave

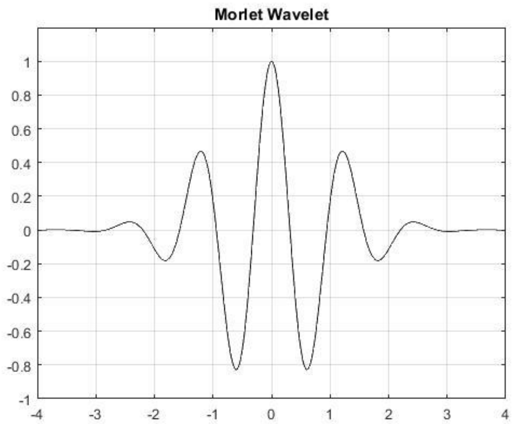


Figure 3.5: Wavelet

3.5.1. Types of Wavelet Transform

Wavelets capability to give spatial frequency information is the main reason for this investigation. This property promises the possibility for better discrimination between the noise and the data. Successful exploitation of wavelet transform might lessen the blurring effect or even overcome it completely. There are mainly two types of wavelet transform namely Continuous Wavelet Transform (CWT) and Discrete Wavelet Transform (DWT).

3.5.1.1. Continuous Wavelet Transform (CWT)

CWT is an implementation of the wavelet transform using an arbitrary scales and almost arbitrary wavelets. Non-orthogonal wavelets are used for its development in the data obtained by this transform for highly correlated. CWT works by computing a convolution of the signal with the scaled wavelet .

3.5.1.2. Discrete Wavelet Transform (DWT)

DWT of image signals produces a non-redundant image representation, which provides better spatial and spectral localization of image formation compared with other multi scale representation such as Gaussian and Laplacian pyramid. The DWT can be interpreted as signal decomposition in a set of independent spatially oriented frequency channels. The signal is passed through two complementary filters and emerges two signals, approximation and details. This is called decomposition or analysis.

The components can be associated back into the original signal without loss of information. This process is called reconstruction or synthesis. The mathematical manipulation, which implies analysis and synthesis, is called Discrete Wavelet Transform and Inverse DWT.

3.6. Discrete Wavelet Transform

In numerical analysis and functional analysis, a discrete wavelet transform (DWT) is any wavelet transform for which the wavelets are discretely sampled. Image is filtered by low pass (for smooth variation between gray level pixels) and high pass filter (for high variation between gray level pixels). Image is decomposed into multilevel which include approximation details (LL sub-band), horizontal detail (HL sub-band), vertical detail (LH sub-band) and diagonal details (HH sub-band).

The discrete wavelet transform uses low-pass and high-pass filters, h(n) and g(n), to expand a digital signal. They are referred to as analysis filters. The dilation performed for each scale is now achieved by a decimator. The coefficients c_k & d_k are produced by convolving the digital signal, with each filter, & then decimating the output. The c_k coefficients are produced by the low-pass filter, h(n) and called coarse coefficients. The d_k coefficients are produced by the high pass filter and called detail coefficients.

Coarse coefficients provide information about low frequencies, & detail coefficients provide information about high frequencies. Coarse & detail coefficients are produced at multiple scales by iterating the process on the coarse coefficients of each scale. The entire process is computed using a tree-structure filter bank, as shown in figure 3.6.

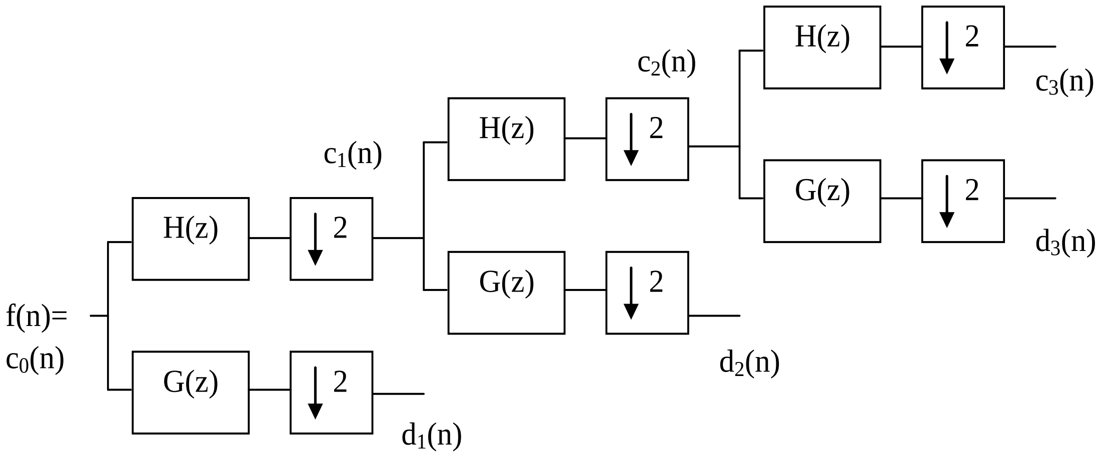


Figure 3.6: Analysis filter bank

After analyzing, or processing, the signal in the wavelet domain it is often necessary to return the signal back to its original domain. This is achieved using synthesis filters and expanders. The wavelet coefficients are applied to a synthesis filter bank to restore the original signal, as shown in Figure 3.7.

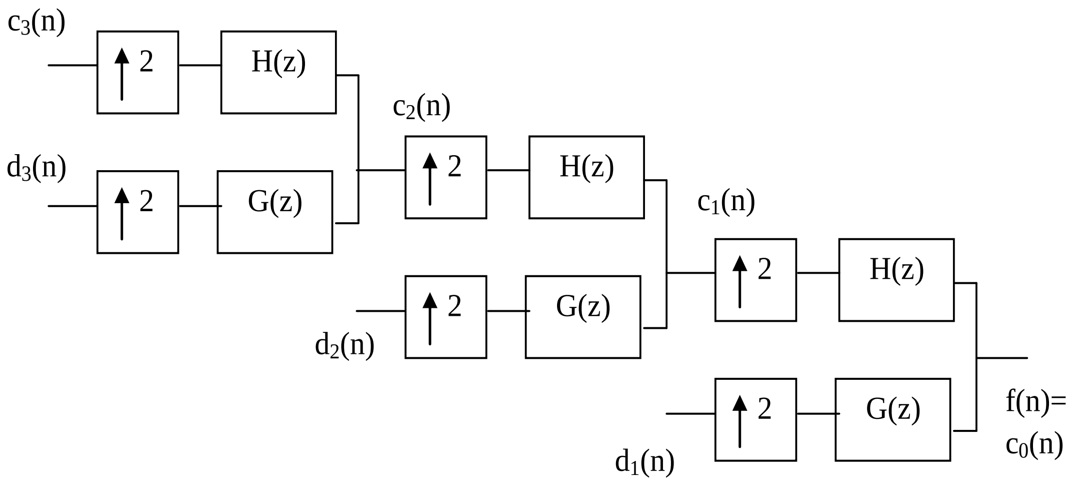


Figure 3.7: Synthesis filter bank

The discrete wavelet transform has a huge number of applications in science, engineering & mathematics and computer science. The wavelet domain representation of an image, or any signal, is useful in many applications, such as compression, noise reduction, watermarking etc.

The two dimensional discrete wavelet transform is essentially a one dimensional analysis of two dimensional signal. It operates on one dimension at a time, by analyzing the rows & columns of an image in a separable fashion. The first step applies the analysis filters to the rows of an image. This produces two new images, where one image is set of coarse row coefficients, & the other a set of detail row coefficients. Next analysis filters are applied to the column of each new images, to produce four different images called sub bands. Rows and columns analyzed with a high pass filter are designated with H. Similarly, rows and columns analyzed with a low pass filter are designated with L. For example, if a sub-band image was produced using a high pass filter in the rows and a low pass filter on the columns, it is called the HL sub-band. Figure 3.8 shows this process in its entirety.

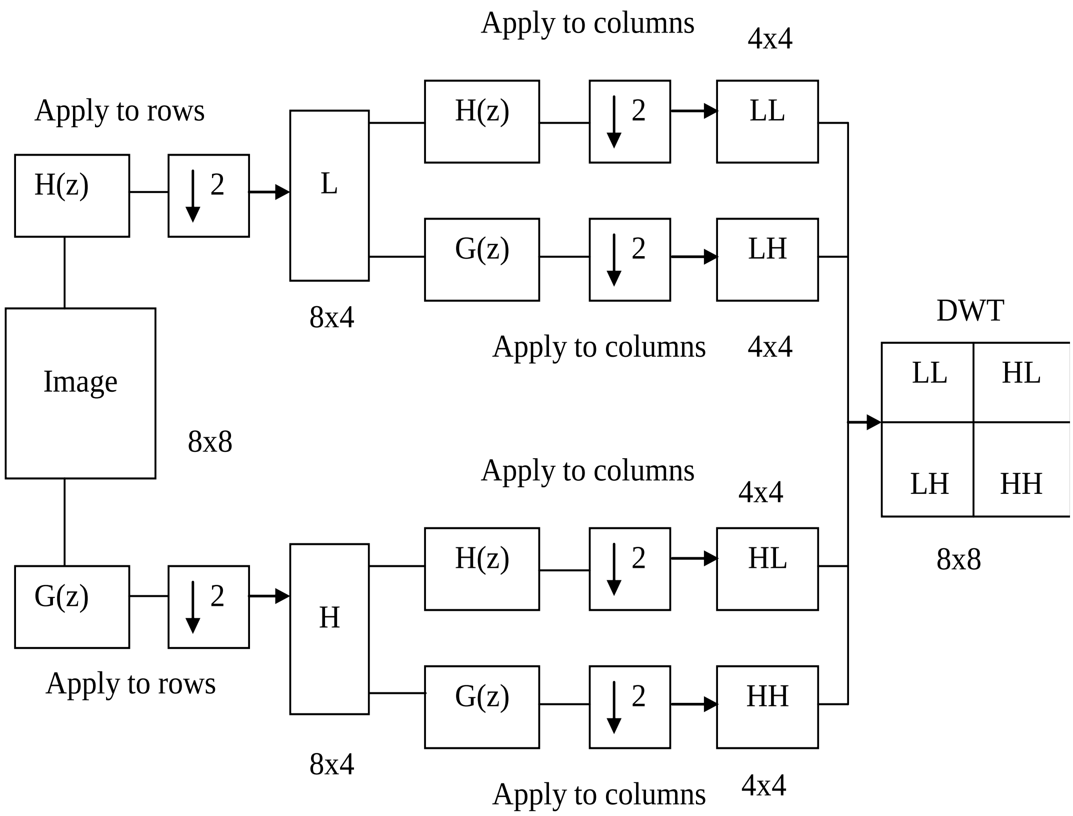


Figure 3.8: Two Dimensional DWT

Each sub-band provides different information about the image. The LL sub-band is a coarse approximation of the image & removes all high frequency information. The LH sub-band removes high frequency information along the rows & emphasizes high frequency information along the columns. The result is an image in which vertical edges are emphasized. Similarly, the HL sub-band emphasizes horizontal edges, & the HH sub-band emphasizes diagonal edges. To compute DWT of the image at the next scale the process is applied again to the LL sub-band.

Each level of the wavelet decomposition, four new images are created from the original N*N pixel image. The size of these new images is reduced to one-fourth of the original size i.e. the new size is N/2 * N/2. The new images are named according to the filter (low-pass or high-pass) which is applied to the original image in horizontal and vertical directions. For example, the LH image is a result of applying the low-pass filter in horizontal direction and high-pass filter in vertical direction. Thus, the four images produced from each decomposition level are LL, LH, HL & HH. The LL image is considered a reduced version of the original as it retains most details. The LH image contains horizontal edge features, while the HL contains vertical edge features. The HH contains high frequency information only & is typically nosy & is, therefore, not useful for the registration. In wavelet decomposition, only LL image is used to produce the next level of decomposition.

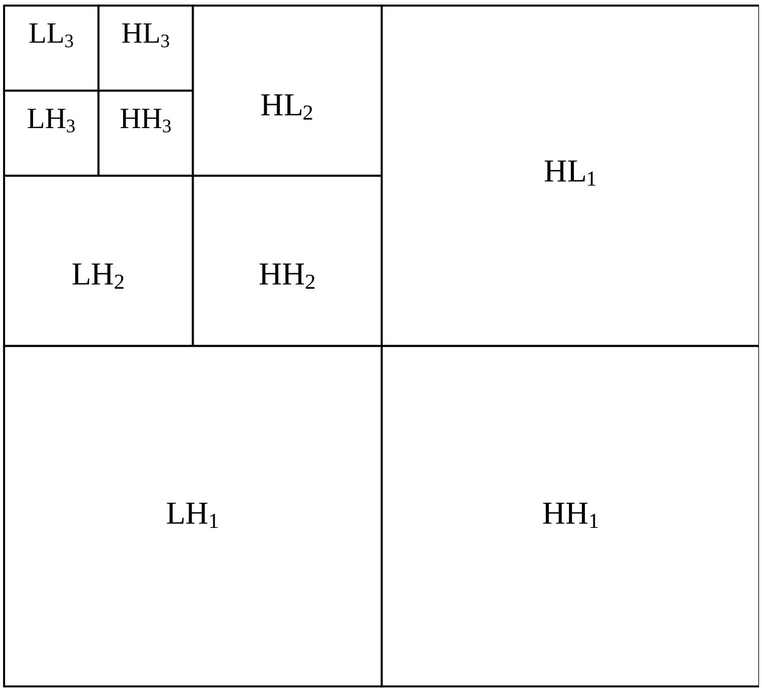


Figure 3.9: Image Decomposition using DWT

3.7. Wavelet Thresholding

In wavelet, coefficients with small absolute value are dominated by noise, while coefficients with large absolute value carry more signal information than noise. Replacing noisy coefficients (small coefficient below a certain threshold value) by zero and an inverse wavelet transform may lead to a reconstruction that has lesser noise. The idea of thresholding was motivated based on the following assumptions:

- The decor relating property of a wavelet transform creates a sparse signal most untouched coefficients are zero or close to zero.
- Noise is spread out equally along all coefficients.
- The noise level is not too high so that the signal wavelet coefficients can be distinguished from the noisy ones.

This method is a simple and efficient for noise reduction. Further, inserting zeros creates more scarcity in the wavelet domain.

3.7.1. Thresholding Method

Hard and soft thresholding techniques are used for purpose of image denoising. In both cases the coefficients with magnitudes less than the threshold are set to zero. The difference between these two thresholding operations lies in how they deal with coefficients that are greater in magnitude than the threshold. In the case of soft thresholding, the coefficients greater in magnitude than the threshold are shrunk toward zero by subtracting the threshold value from the coefficient value whereas in hard thresholding the coefficients greater in magnitude than threshold are left unchanged. As soft thresholding gives more visually pleasant image and reduces the abrupt sharp changes that occurs in hard thresholding, therefore soft thresholding is preferred over hard thresholding.

The Hard Thresholding T_H is defined as:

$$T_H = \begin{cases} x \text{ for } |x| \geq t \\ 0 \text{ in all other regions} \end{cases} \tag{3.8}$$

The Soft thresholding T_S is defined as:

$$T_S = \begin{cases} \text{sign}(x)(|x| - t) \text{ for} |x| \geq t \\ 0 \text{ in all other regions} \end{cases} \tag{3.9}$$

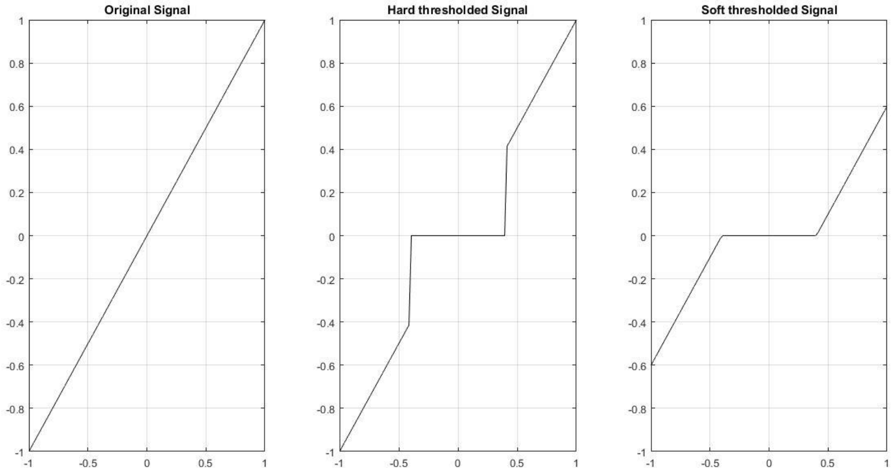


Figure 3.10: Hard & Soft Thresholding Operation

3.7.2. Threshold Selection Rules

In image denoising application, PSNR needs to be maximized, hence optimal value should be selected. Finding an optimal value for thresholding is not an easy task. If we select a smaller threshold then it will pass all the noisy coefficients and hence resultant images may still be noisy but larger threshold makes more number of coefficients to zero, which provides smoothness in image and image processing may cause blur and artifacts, and hence the resultant images may lose some signal values. Some of the thresholding schemes are discussed below:

3.7.2.1. Universal Threshold

The value of universal threshold is given by the following equation:

$$T = \sigma\sqrt{2\log M} \tag{3.10}$$

where σ^2 being the noise variance and M is the number of pixels. It is optimal threshold in asymptotic sense and minimizes the cost function of difference between the function. It is assumed that if number of samples is large, then the universal threshold may give better estimate for soft threshold.

3.7.2.2. Visu Shrink

Visu Shrink is thresholding by applying the threshold proposed by Donoho & Johnston. It follows hard threshold rule. The drawbacks of this shrinkage is that neither speckle noise can be removed nor MSE can be minimized. It can only deal with additive noise. The threshold T can be calculated using the formula:

$$T_V = \sigma\sqrt{2\log N} \tag{3.11}$$

where σ^2 is the noise variance and N represents the size of original image i.e. total number of pixels in the image.

3.7.2.3. Bayes Shrink

The Bayes Shrink method has been attracting attention recently as an algorithm for setting different thresholds for every sub band. Here sub-bands refer to frequency bands that are different from each other in level and direction. Bayes Shrink uses soft thresholding. The purpose of this method is to estimate a threshold value that minimizes the Bayesian risk assuming Generalized Gaussian Distribution (GGD) prior. Bayes threshold is defined as:

$$T_B = {\sigma_n}^2/\sigma_F \tag{3.12}$$

where ${\sigma_n}^2$ is the estimated noise variance by robust median estimator and σ_F is estimated signal standard deviation in wavelet domain. The robust median estimator is stated as in equation (3.13).

$${\sigma_n}^2 = \frac{\text{Median}(\{Y_{ij}\})}{0.6745}, Y_{ij}\,\varepsilon\ \text{subband}\ HH_1 \tag{3.13}$$

This estimator is used when there is no a priori knowledge about the noise variance. The estimated signal standard deviation is calculated using equation 3.14.

$$\sigma_F = \sqrt{\max(({\sigma_Y}^2 - {\sigma_n}^2), 0)} \tag{3.14}$$

where, ${\sigma_Y}^2$ is the variance of Y. Since Y is modelled as zero-mean, ${\sigma_Y}^2$ can be found empirically by equation 3.15.

$${\sigma_Y}^2 = \frac{1}{n^2}\sum_{i,j=1}^{n} {Y_{i,j}}^2 \tag{3.15}$$

Incase ${\sigma_n}^2 \geq {\sigma_Y}^2$, σ_F will become 0. That is, T_B becomes ∞. Hence, for this case $T_B=\max(\{|Y_{ij}|\})$.

3.7.2.4. Neigh Shrink

Chen et al. proposed a wavelet-domain image thresholding scheme by incorporating neighbouring coefficients, namely NeighShrink [19]. The method thresholds the wavelet coefficients according to the magnitude of the squared sum of all the wavelet coefficients, i.e., the local energy within the neighborhood window. The neighborhood window size may be 3×3, 5×5, 7×7, 9×9, etc. But, the authors have already demonstrated through the results that the 3×3 window is the best among all window sizes . The shrinkage function for NeighShrink of any arbitrary 3×3 window centred at (i,j) is expressed as in equation 3.16.

$$T_{ij} = \left(1 - \frac{{T_U}^2}{{s_{ij}}^2}\right)_+ \tag{3.16}$$

where T_U is the universal threshold and ${s_{ij}}^2$ is the squared sum of all wavelet coefficients in the respective 3×3 window given by equation 3.17

$$S_{ij}{}^2 = \sum_{n=j-1}^{j+1} \sum_{m=i-1}^{i+1} Y_{m,n}{}^2 \tag{3.17}$$

where, + sign at the end of the formula means to keep the positive values while setting it to zero when it is negative. The estimated center wavelet coefficient F_{ij} is then calculated from its noisy counterpart Y_{ij} as in equation 3.18.

$$F_{ij} = T_{ij} Y_{ij} \tag{3.18}$$

3.8. Genetic Algorithm

The genetic algorithm is a method for solving both constrained and unconstrained optimization problems that is based on natural selection, the process that drives biological evolution. The genetic algorithm repeatedly modifies a population of individual solutions. At each step, the genetic algorithm selects individuals at random from the current population to be parents and uses them to produce the children for the next generation. Over successive generations, the population "evolves" toward an optimal solution. We can apply the genetic algorithm to solve a variety of optimization problems that are not well suited for standard optimization algorithms, including problems in which the objective function is discontinuous, non-differentiable, stochastic, or highly nonlinear. The genetic algorithm can address problems of mixed integer programming, where some components are restricted to be integer-valued.

The genetic algorithm uses three main types of rules at each step to create the next generation from the current population:

- *Selection rules* select the individuals, called *parents,* that contribute to the population at the next generation.
- *Crossover rules* combine two parents to form children for the next generation.
- *Mutation rules* apply random changes to individual parents to form children.

3.9. Genetic Algorithm Terminology

3.9.1. Fitness Functions

The fitness function is the function we want to optimize. For standard optimization algorithms, this is known as the objective function.

3.9.2. Individuals

An individual is any point to which we can apply the fitness function. The value of the fitness function for an individual is its score. An individual is sometimes referred to as a genome and the vector entries of an individual as genes.

3.9.3. Populations and Generations

A population is an array of individuals. At each iteration, the genetic algorithm performs a series of computations on the current population to produce a new population. Each successive population is called a new generation.

3.9.4. Fitness Values and Best Fitness Values

The fitness value of an individual is the value of the fitness function for that individual. If our objective is to find the minimum of the fitness function, the best fitness value for a population is the smallest fitness value for any individual in the population.

3.9.5. Parents and Children

To create the next generation, the genetic algorithm selects certain individuals in the current population, called parents, and uses them to create individuals in the next generation, called children. Typically, the algorithm is more likely to select parents that have better fitness values

CHAPTER 4: METHODOLOGY

This work is proposed based on the Genetic Algorithm where a noisy image I is used as input and the population is created by applying mutation operators on the noisy image. Furthermore, some of the crossover operators are the same introduced in [1]. However, this work proposed to combine the GA with image denoising technique using wavelets, which is different from the work in [1]. Each individual in this work will be represented by a two-dimensional array of pixels whose values are integers in the interval [0, 255].

The basic block diagram of the proposed method is mentioned in the following section.

4.1. Block Diagram

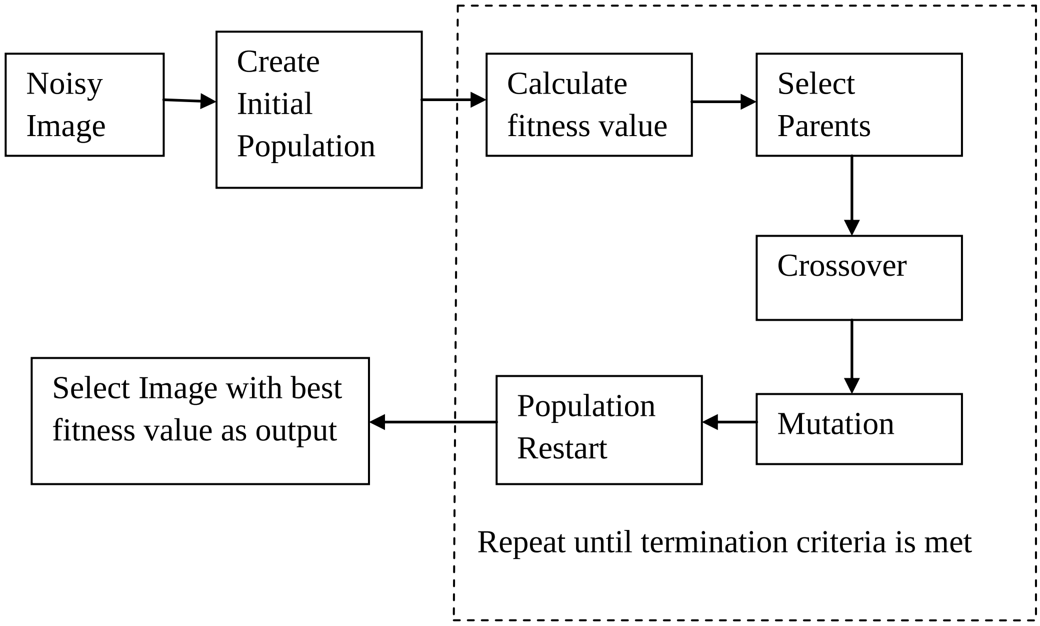


Figure 4.1: Block Diagram of Proposed Method

Each part of the block diagram is described briefly in the following section:

4.1.1. Population Initialization

The algorithm will start initializing its population of images following two steps. First, the input noisy image will be set as the initial individual that will be applied to three wavelet based denoising techniques:

- Visu Shrink
- Bayes Shrink

- NeighSure Shrink

These methods will be used as local search operators to improve the initial individual (original noisy image). Then a denoised image will be returned by each of the three wavelet based denoising techniques.

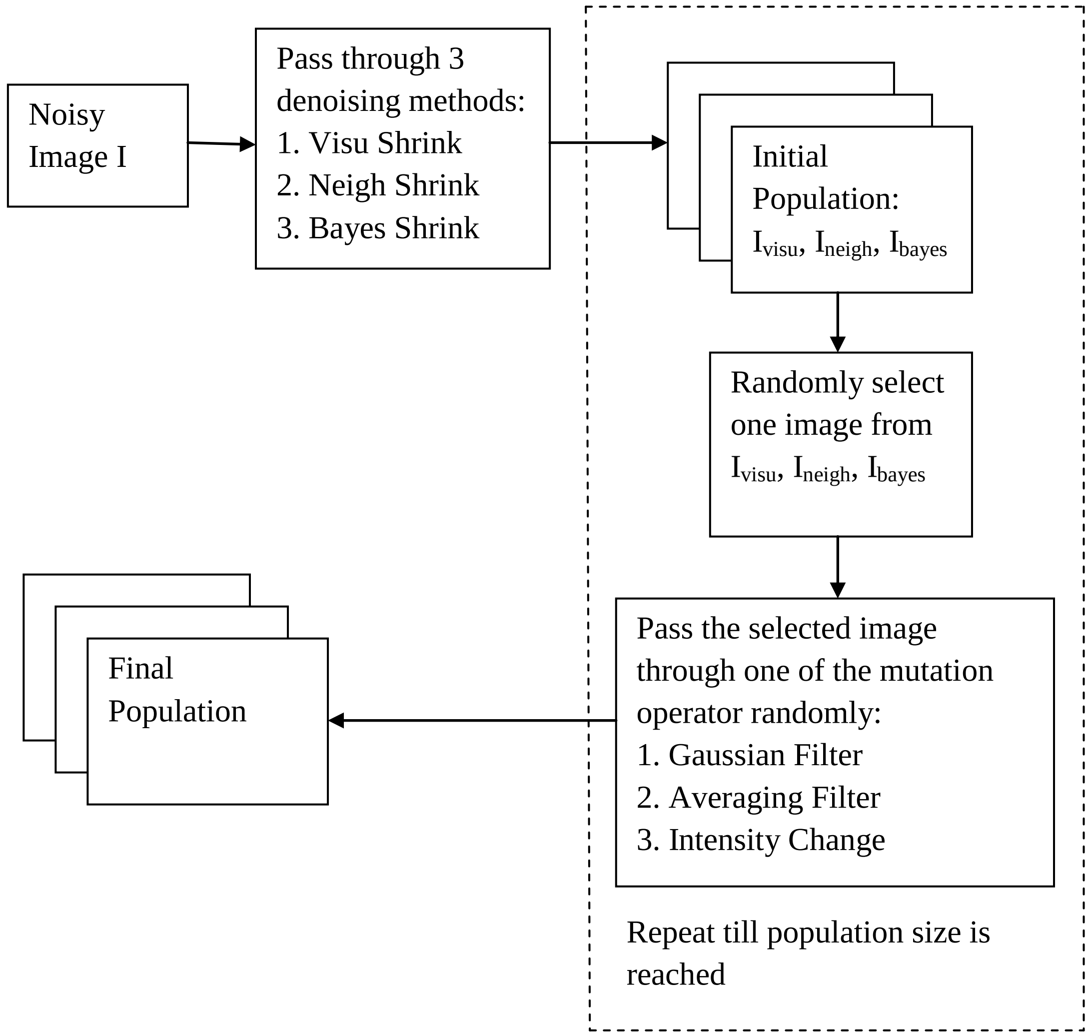


Figure 4.2: Procedure for initializing the population

The outcomes of these methods will be included in the initial population. Thus, we will have three individuals in the initial population at the end of the first step.

In the second step, one of these three outputs will be randomly chosen and submitted to a mutation operation, which will be also chosen randomly. The mutation operator is described as follows:

- **Gaussian blur**: filters the image with a Gaussian filter. The size of the filter will be chosen randomly between 3×3 pixels and 5×5 pixels.

- **Averaging filter**: filters the image with an averaging filter. The size of the filter will be chosen randomly between 3×3 pixels and 5×5pixels.
- **Intensity change**: all the pixels of the image will be multiplied by the same random factor, which lightens or darkens the image as a whole.

These filters are named as mutation operators since they will apply some changes in the image recovered by the previous denoising methods. The resulting image will be included in the population, so the second step continues until the population size has been reached. At the end of this process, the initial population will be formed by the outputs of the three image denoising methods and by the images submitted to the mutation rounds. Figure 4.2 illustrates the two steps of this initialization procedure.

4.1.2. Fitness Evaluation

This algorithm will be guided by the fitness function expressed in Equation (4.1).

$$\mathrm{fitness}(\mathrm{I}) = \left(\sum_{\Omega} \sqrt{1 + \beta^2 |\nabla \mathrm{I}|^2}\right) + \frac{\lambda}{2}(\mathrm{I} - \mathrm{I}_0)^2 \qquad (4.1)$$

This function is aware of the image edges and tries to preserve important features of the image as described in [18]. The term $(I-I_0)^2$ guarantees a certain degree of fidelity between the image being evaluated and the original image, where I is the image being evaluated and I_0 the noisy image. The parameter ∇I is a total variation regularizing term, β and λ are balancing parameters and Ω is the set of all points in the image. By minimizing Equation (4.1) we, are basically trying to reduce the total variation of the image while preserving fidelity in relation to the original image.

4.1.3. Parent Selection

Parents are chosen by a tournament selection process for crossover operation until the interim population size is reached. Out of the images selected as per tournament size, individual with better fitness value is selected as a parent.

4.1.4. Crossover

After the selection of the parents, the new individual will be created by randomly choosing one out of three crossover operators presented next:

- **one-point row**: a row of pixels is randomly chosen. All the pixels above this row will come from one parent and all the pixels below this row will come from the second parent.
- **one-point column**: similar to the previous method, but a column is chosen rather than the row.
- **point-to-point random**: randomly chooses each pixel from one of the parents until the new individual is created.

4.1.5. Mutation

After the crossover operator, the new individual may also pass through a local search operator, when a randomly chosen value from [0, 1] is lower than the local search rate of the algorithm. If an individual is selected for mutation, it will be done by a denoising method chosen at random out of the three: Visu Shrink, Neigh Shrink or Bayes Shrink.

4.1.6. Population Replacement and Intermediate Population

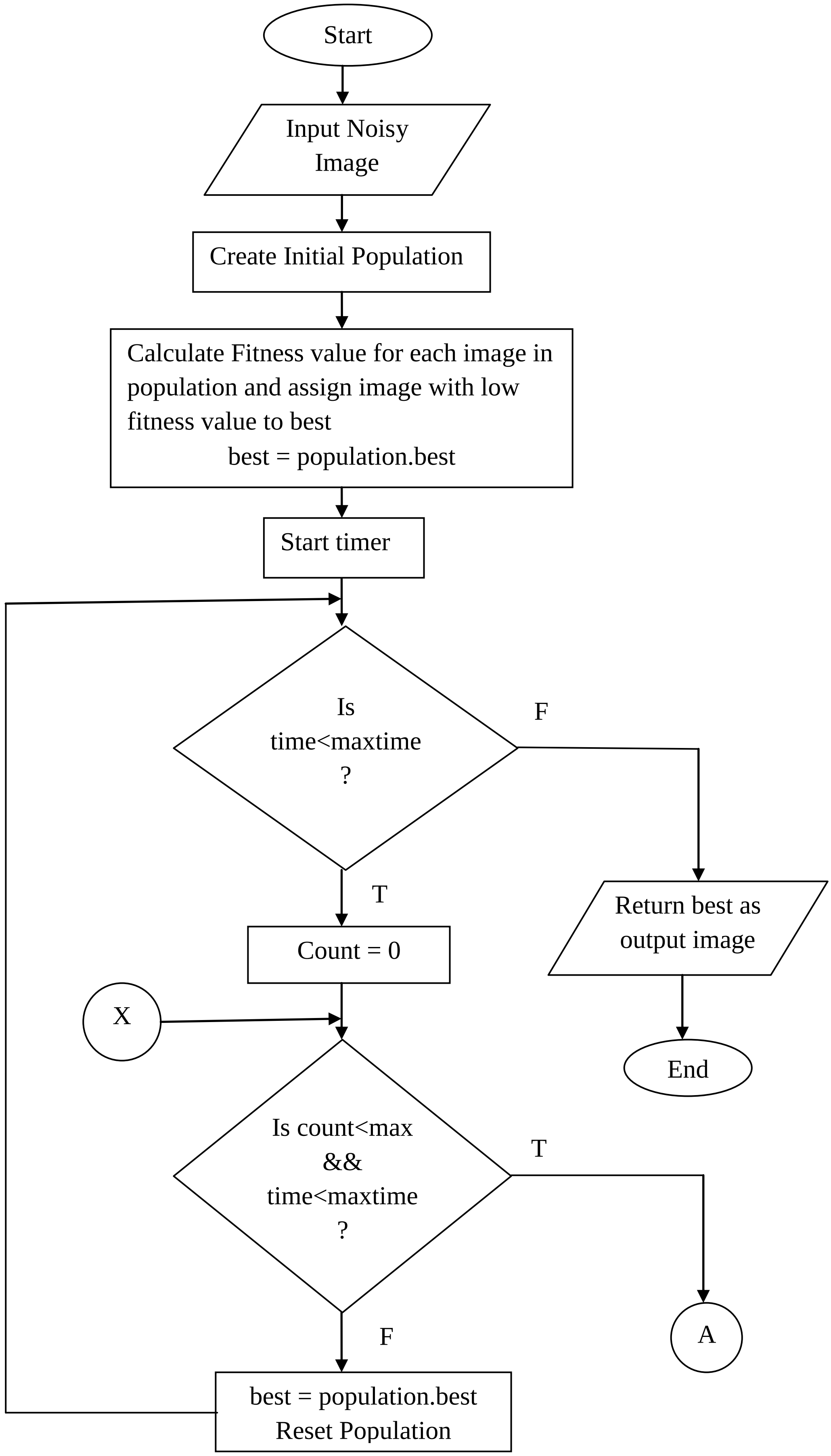
During the evolutionary process, the population will continue to evolve while there is no change in the best individual for a number of iterations. When this number is reached, the population will be restarted keeping only the best individual found so far. The other individuals will be generated again by the initialization procedure previously described. An intermediate population will be created applying crossover and local search operators through the evolutionary process. This population will be two times larger than the population size, formed by the individuals of the current population and those new generated individuals. Such individuals will be created by crossing over parents that are chosen by a tournament selection process.

Once the intermediate population is completed, all of its individuals will be sorted according to their fitness values and the best population size individuals become the population for the next iteration.

4.1.7. Termination Condition

The evolutionary process will run for a fixed amount of time. After that, the algorithm returns the best individual i.e. image with best fitness value (minimum fitness value in this case) from the population of the last generation.

4.2. Flow Chart



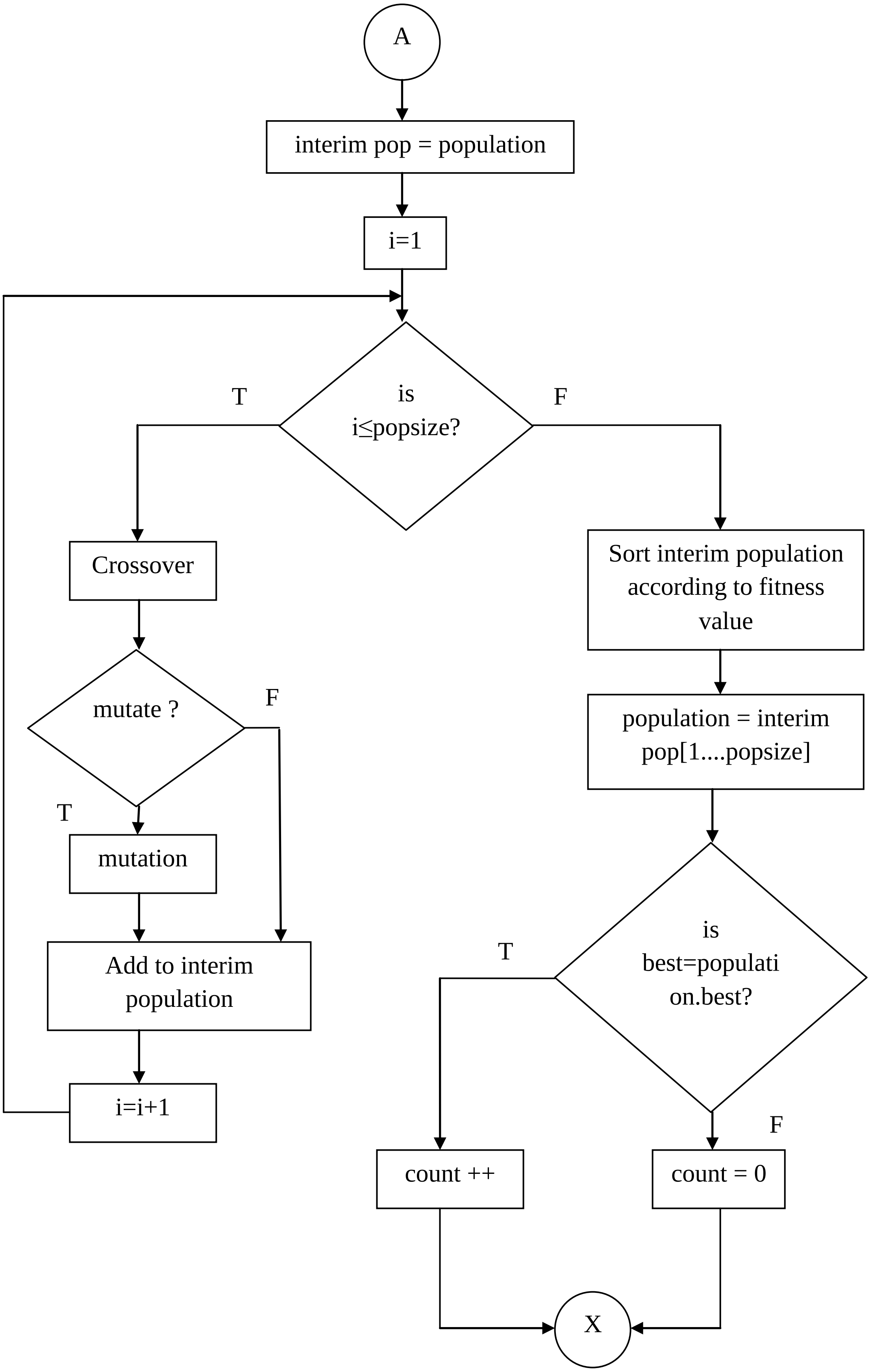


Figure 4.3: Flowchart for the proposed method

4.3. Comparison Metrics

4.3.1. PSNR

The peak signal to noise ratio is one of the most common metrics used in image processing. It is measured in decibels(dB) and defined as in Equation (4.2) for 8-bit gray scale images:

$$\mathrm{PSNR} = 10\log_{10}\left(\frac{255^2}{\mathrm{MSE}}\right)\,\mathrm{db} \tag{4.2}$$

where MSE is the mean squared error between the original and the recovered image. It is defined in Equation (4.3).

$$\mathrm{MSE} = \frac{1}{\mathrm{MN}}\sum_{i=1}^{M}\sum_{j=1}^{N}(X(i,j) - P(i,j))^2 \tag{4.3}$$

where,

M- Width of Image

N- Height of Image

X- Original Image

P- Recovered Image

4.3.2. SSIM

This metric maps two images to an interval [-1,1], where similar images have higher values. This metric is defined in Equation (4.4), where μ_A, μ_B, σ_A and σ_B are the values of mean and standard deviation for A and B, σ_{AB} is the covariance between A and B, $c_1=(k_1L)^2$ and $c_2=(k_2L)^2$, where L is the dynamic range of the pixel values ($2^{bitsperpixel}$-1) and k_1=0.001 and k_2=0.03 are constants.

$$\mathrm{SSIM}(A,B) = \frac{(2\mu_A\mu_B + c_1)(2\sigma_{AB} + c_2)}{({\mu_A}^2 + {\mu_B}^2 + c1)({\sigma_A}^2 + {\sigma_B}^2 + c_2)} \tag{4.4}$$

CHAPTER 5: RESULTS & ANALYSIS

5.1. Test Images

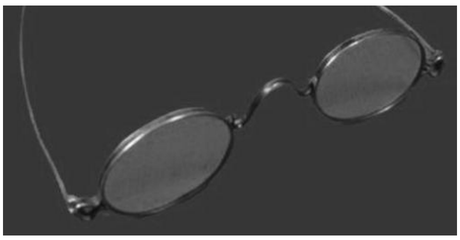
The images in Figure 5.1 were selected to evaluate the result of the algorithm. These images are commonly used by the scientific community as benchmark for image denoising problems.

(a) Boat

(b) Hill

(c) Man

(d) Lena



(e) Glasses

(f) Lightning

Figure 5.1: Test images used to evaluate the proposed denoising method

The algorithm as well as all other methods used for comparison was implemented in Matlab R2016b 64-bit. The algorithm was executed 10 times for each image and for each noise standard deviation level. These experiments were conducted on Intel Core i3 M380 2.53 GHz processor with 4 GB RAM in windows 7.

5.2. Setting Parameter

Tests were performed to find the configuration for the parameters of the algorithm. Each test evaluated different values for a specific parameter, then, PSNR and SSIM were calculated for each best image achieved and the values were compared against each other.

Initially, the configuration for the algorithm was set as tournament size of 3, local search rate of 0.4, population size of 15, β=1.5, and running time of 20mins. This configuration was used taking into account the computational time spent by executing the genetic algorithm combined with wavelet shrinkage denoising methods. For example, it is not possible to set a large sized population because it makes initialization, reinitialization and mutation processes very time consuming once they use the denoising methods at multiple times.

For setting the parameters value, the algorithm was executed over Boat image corrupted with additive Gaussian noise N(0,σ) with six different values for standard deviation σ = 10, 20, 30, 40, 50 and 60.

Tournament size for parent's selection was evaluated first. As the initial population size is not large, the values of tournament size tested were 3,6 & 9. The average (Avg), maximum (Max) and minimum (Min) results for these different values are shown in table 5.1 and 5.2. These values were obtained by executing the algorithm 10 times for each noise level.

Table 5.1: PSNR values for different tournament sizes

Noise	Tournament: 9			Tournament: 6			Tournament: 3		
	Avg	Max	Min	Avg	Max	Min	Avg	Max	Min
10	31.6389	32.4147	30.4146	31.8812	32.4147	30.5285	31.9387	32.4147	30.1996
20	27.1803	27.3791	26.5395	27.5872	28.6731	27.1174	27.4683	27.5652	27.4109
30	25.9128	26.3186	25.1755	25.5312	26.1133	25.2406	26.1126	26.5658	25.5348
40	24.24	24.4142	24.1568	24.4946	24.8153	24.3359	24.4592	24.56	24.3309

50	23.7751	23.8568	23.7147	23.8256	23.9244	23.7233	23.8439	23.9133	23.7957
60	23.1238	23.1872	23.0148	23.1045	23.2	23.0059	23.1496	23.1938	23.1163

Table 5.2: SSIM values for different tournament sizes

Noise	Tournament: 9			Tournament: 6			Tournament: 3		
	Avg	Max	Min	Avg	Max	Min	Avg	Max	Min
10	0.84654	0.854	0.83399	0.84986	0.854	0.83844	0.84924	0.854	0.83152
20	0.73163	0.7411	0.71188	0.74104	0.75788	0.72533	0.73881	0.74062	0.73637
30	0.66955	0.68257	0.65645	0.66619	0.67645	0.66167	0.6765	0.68419	0.66247
40	0.60714	0.61466	0.60222	0.61755	0.61883	0.61606	0.61854	0.62324	0.61263
50	0.57795	0.58104	0.57611	0.57942	0.58115	0.57655	0.57807	0.58051	0.57351
60	0.54461	0.54801	0.53966	0.54432	0.54801	0.53965	0.54568	0.54808	0.54287

The tournament size of 3 achieved better performance than other size values in table 5.1 especially for higher level of noise. The values found for SSIM in table 5.2 are similar. Thus, tournament size of 3 was set as default value.

The second parameter tested was local search rate for values of 0.2, 0.4 and 0.6. The results with different mutation rates are shown in tables 5.3 and 5.4.

Table 5.3: PSNR values for different local search rate

Noise	lsr: 0.2			lsr: 0.4			lsr: 0.6		
	Avg	Max	Min	Avg	Max	Min	Avg	Max	Min
10	31.9267	32.4147	31.1852	31.9387	32.4147	30.1996	32.3307	32.4148	31.9953
20	27.5586	27.8273	27.2568	27.4683	27.5652	27.4109	27.8091	28.7925	27.3859
30	25.6207	26.0591	25.0685	26.1126	26.5658	25.5348	26.0378	26.5448	25.3924
40	24.4666	24.6315	24.3163	24.4592	24.56	24.3309	24.5368	24.7709	24.3177
50	23.7784	23.8185	23.7186	23.8439	23.9133	23.7957	23.8051	23.8948	23.7352
60	23.1423	23.1921	23.0902	23.1496	23.1938	23.1163	23.1562	23.1854	23.1258

Table 5.4: SSIM values for different local search rate

Noise	lsr: 0.2			lsr: 0.4			lsr: 0.6		
	Avg	Max	Min	Avg	Max	Min	Avg	Max	Min
10	0.84952	0.854	0.84206	0.84924	0.854	0.83152	0.85362	0.85401	0.85214
20	0.7402	0.74647	0.7347	0.73881	0.74062	0.73637	0.74503	0.76202	0.73893
30	0.66627	0.67583	0.65125	0.6765	0.68419	0.66247	0.67276	0.67943	0.66412
40	0.61856	0.62287	0.61384	0.61854	0.62324	0.61263	0.6186	0.62633	0.6123
50	0.57889	0.58046	0.57664	0.57807	0.58051	0.57351	0.57913	0.58048	0.57635
60	0.54341	0.54801	0.53918	0.54568	0.54808	0.54287	0.54714	0.54813	0.54441

Local search rate of 0.6 has better PSNR values as compared to others in table 5.3. Hence local search rate of 0.6 was set as default value.

Population size was the next parameter to be analyzed. For this test population size of 10, 15, 20 & 25 were used.

Table 5.5: PSNR values for different population size

Noise	Pop Size: 10			Pop Size: 15			Pop Size: 20			Pop Size: 25		
	Avg	Max	Min	Avg	Max	Min	Avg	Max	Min	Avg	Max	Min
10	31.9279	32.4147	31.0202	32.3307	32.4148	31.9953	32.3616	32.4148	32.2477	31.6269	32.4146	30.0736
20	27.4834	28.1651	27.1954	27.8091	28.7925	27.3859	27.7628	28.7757	27.2644	27.4818	27.8362	27.1064
30	25.4423	26.1134	24.8549	26.0378	26.5448	25.3924	25.9929	26.4761	25.5394	25.6264	26.137	25.1711
40	24.5722	25.0156	24.3003	24.5368	24.7709	24.3177	24.6633	25.1094	24.3455	24.6592	24.9244	24.452
50	23.766	23.815	23.7147	23.8051	23.8948	23.7352	23.7963	23.839	23.7639	23.7965	23.8974	23.752
60	23.13	23.2155	23.0431	23.1562	23.1854	23.1258	23.1713	23.2634	23.1279	23.1395	23.1765	23.0853

Table 5.6: SSIM values for different population size

Noise	Pop Size: 10			Pop Size: 15			Pop Size: 20			Pop Size: 25		
	Avg	Max	Min	Avg	Max	Min	Avg	Max	Min	Avg	Max	Min
10	0.85012	0.854	0.84201	0.85362	0.85401	0.85214	0.85295	0.85401	0.85021	0.84634	0.854	0.82972
20	0.73687	0.74531	0.72404	0.74503	0.76202	0.73893	0.7444	0.76197	0.73563	0.73848	0.74435	0.73253
30	0.6579	0.67645	0.62689	0.67276	0.67943	0.66412	0.67306	0.6805	0.66642	0.66492	0.67496	0.65882
40	0.61665	0.62776	0.60964	0.6186	0.62633	0.6123	0.62194	0.6312	0.61437	0.62074	0.62753	0.61693
50	0.57848	0.57983	0.57676	0.57913	0.58048	0.57635	0.57877	0.5799	0.5757	0.57872	0.58046	0.57621
60	0.54369	0.54801	0.53297	0.54714	0.54813	0.54441	0.54764	0.54803	0.5467	0.54536	0.548	0.54227

The values of PSNR and SSIM for different population sizes are similar for higher level of noise. These values for lower level of noise are somewhat distinct. Population size of 15 and 20 show distinct differences for lower noise levels as compared to others and they both don’t show significant difference between them. So population size of 15 was chosen so that initialization, reinitialization and mutation process are less time consuming as compared to size of 20.

For the parameter β in the fitness function the author in [18] mentioned that values between 1 & 3 would be more effective. So β was evaluated for 1, 1.5 & 2. The results are presented in table 5.7 & 5.8.

Table 5.7: PSNR values for different value of β

Noise	beta: 1			beta: 1.5			beta: 2		
	Avg	Max	Min	Avg	Max	Min	Avg	Max	Min
10	32.3082	32.4146	31.8829	32.3307	32.4148	31.9953	29.8747	31.1202	28.9559
20	28.92	28.9209	28.9165	27.8091	28.7925	27.3859	26.688	27.9813	25.4204
30	26.6888	26.7897	26.6434	26.0378	26.5448	25.3924	25.137	25.8411	24.7833
40	25.1294	25.268	25.0192	24.5368	24.7709	24.3177	24.3843	24.4908	24.3
50	24.0887	24.2065	23.9	23.8051	23.8948	23.7352	23.7357	23.8203	23.7053
60	23.2069	23.3216	23.1157	23.1562	23.1854	23.1258	23.1238	23.2284	23.0076

Table 5.8: SSIM values for different value of β

Noise	beta: 1			beta: 1.5			beta: 2		
	Avg	Max	Min	Avg	Max	Min	Avg	Max	Min
10	0.85336	0.854	0.85079	0.85362	0.85401	0.85214	0.82148	0.83836	0.80713
20	0.75538	0.76128	0.7539	0.74503	0.76202	0.73893	0.71745	0.74831	0.69021
30	0.66505	0.66989	0.66217	0.67276	0.67943	0.66412	0.65493	0.67406	0.64397
40	0.61121	0.62524	0.59321	0.6186	0.62633	0.6123	0.61731	0.62187	0.61468
50	0.57388	0.57893	0.56526	0.57913	0.58048	0.57635	0.57715	0.58027	0.57532
60	0.53458	0.54666	0.51349	0.54714	0.54813	0.54441	0.54363	0.54796	0.53946

From table 5.7 & 5.8 the value of β was chosen to be 1.

The last tested parameter was execution time of the algorithm. For this test the execution time of 10, 15, 20 & 25 minutes were chosen. The results are presented in table 5.9 & 5.10.

Table 5.9: PSNR values for different execution time

Noise	Time: 10mins			Time: 15mins			Time: 20mins			Time: 25mins		
	Avg	Max	Min	Avg	Max	Min	Avg	Max	Min	Avg	Max	Min
10	31.9692	32.4144	30.2842	32.106	32.4148	31.3956	32.3082	32.4146	31.8829	30.9926	32.1707	30.3906
20	27.5655	27.9077	27.3427	27.8246	28.3866	27.3141	28.92	28.9209	28.9165	27.56	27.885	27.3894
30	26.0669	26.3731	25.8664	25.7748	26.0985	25.2794	26.6888	26.7897	26.6434	25.7539	26.5574	25.2476
40	24.6382	24.8907	24.4114	24.4452	24.5842	24.2651	25.1294	25.268	25.0192	24.5635	24.6413	24.4491
50	23.7994	23.8835	23.6374	23.8141	23.8826	23.7153	24.0887	24.2065	23.9	23.831	23.8505	23.8192
60	23.1152	23.1627	23.0128	23.1722	23.2039	23.1462	23.2069	23.3216	23.1157	23.1621	23.1795	23.1515

Table 5.10: SSIM values for different execution time

Noise	Time: 10mins			Time: 15mins			Time: 20mins			Time: 25mins		
	Avg	Max	Min	Avg	Max	Min	Avg	Max	Min	Avg	Max	Min
10	0.84977	0.85398	0.83471	0.85143	0.85401	0.84461	0.85336	0.854	0.85079	0.83982	0.85271	0.8323
20	0.73701	0.74377	0.73352	0.74583	0.75582	0.73856	0.75538	0.76128	0.7539	0.73959	0.74891	0.73367
30	0.67423	0.6824	0.66889	0.66646	0.67632	0.66112	0.66505	0.66989	0.66217	0.66945	0.68217	0.66006
40	0.61992	0.62609	0.61505	0.61509	0.62365	0.60181	0.61121	0.62524	0.59321	0.61944	0.62387	0.61695
50	0.57375	0.58077	0.56731	0.57843	0.58099	0.57643	0.57388	0.57893	0.56526	0.5779	0.58052	0.57282
60	0.54515	0.54811	0.53959	0.5471	0.54806	0.54494	0.53458	0.54666	0.51349	0.54704	0.54754	0.54609

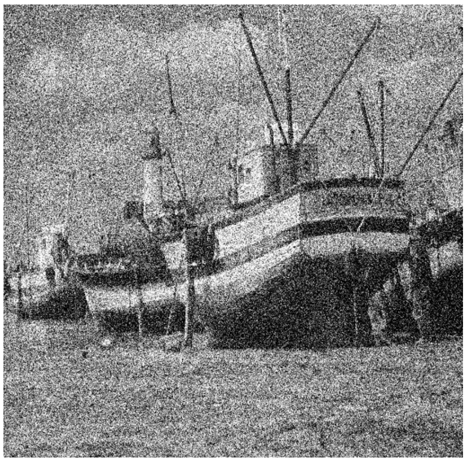
Execution time of 20minutes shows better value for PSNR as compared to others. Hence it was chosen as default value.

After performing these tests, the basic configurations of the parameters was set as tournament size of 3, local search rate of 0.6, population size of 15, value of β=1 and running time of 20 minutes.

5.3. Image with Gaussian Noise

The test images in figure 5.1 were used and these images were deteriorated with an additive Gaussian noise N(0,σ) with six different values for standard deviation σ = 10, 20, 30, 40, 50 and 60. The noisy image was used as input to the proposed algorithm and the algorithm was executed for 10 times for each image. After the completion of the algorithm the comparison metrics i.e. PSNR & SSIM were calculated for each obtained image. The maximum values of these metrics after 10 executions are compared with the metrics of the image obtained from wavelet based denoising.

(a) Original



(b) Noisy (σ=60)

(c) Visu Shrink

(d) NeighSure Shrink

(e) Bayes

(f) Proposed method

Figure 5.2: Denoising Result of Boat image

(a) Original

(b) Noisy (σ=50)

(c) Visu Shrink

(d) NeighSure Shrink

(e) Bayes Shrink

(f) Proposed method

Figure 5.3: Denoising Result of Man image

(a) Original

(b) Noisy (σ=60)

(c) Visu Shrink

(d) NeighSure Shrink

(e) Bayes Shrink

(f) Proposed method

Figure 5.4: Denoising Result of Hill image

(a) Original

(b) Noisy (σ=60)

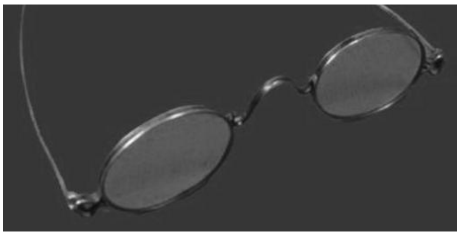
(c) Visu Shrink

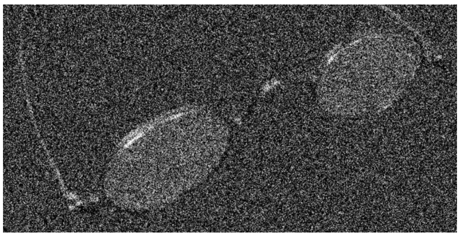
(d) NeighSure Shrink

(e) Bayes Shrink

(f) Proposed method

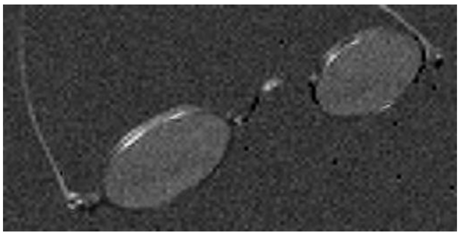
Figure 5.5: Denoising Result of Lena image





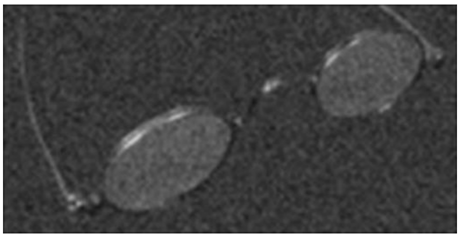
(a) Original

(b) Noisy (σ=60)



(c) Visu Shrink

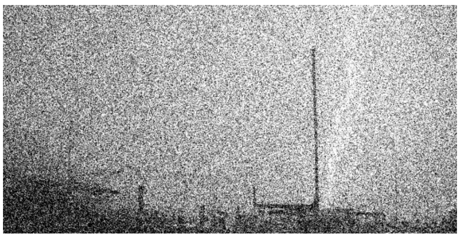
(d) NeighSure Shrink



(e) Bayes Shrink

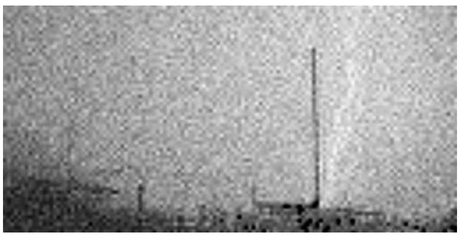
(f) Proposed method

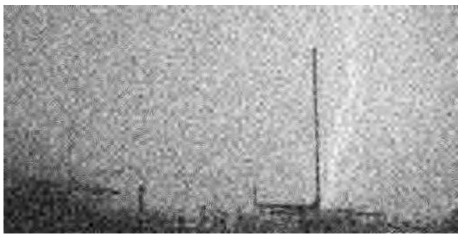
Figure 5.6: Denoising Result of Glasses image



(a) Original

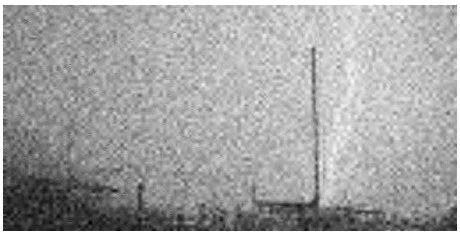
(b) Noisy (σ=60)





(c) Visu Shrink

(d) NeighSure Shrink



(e) Bayes Shrink

(f) Proposed method

Figure 5.7: Denoising Result of Lightning image

Table 5.11: Comparison of PSNR for denoised images corrupted by Gaussian Noise

Image	σ	Noisy Image	Visu Shrink	NeighSure Shrink	Bayes Shrink	Proposed Genetic Algorithm
boat	10	28.1477	28.9773	**32.4146**	31.7814	32.3347
	20	22.189	26.52	28.9208	28.31	**30.3156**
	30	18.7455	25.1555	26.67	26.1874	**28.5994**
	40	16.3544	24.1459	25.0926	24.7858	**27.7059**
	50	14.6079	23.2234	23.7633	23.5725	**24.7768**
	60	13.2382	22.3883	22.71	22.589	**23.4908**
glasses	10	28.1332	36.9027	38.1691	37.3638	**41.6787**
	20	22.115	32.34	32.8294	32.6083	**39.4528**
	30	18.8612	29.6567	30.0701	29.9977	**37.2065**
	40	16.6635	25.3908	27.8036	27.8917	**31.61**
	50	15.0928	21.7697	26.271	26.4765	**30.9628**
	60	13.8205	19.8891	24.7938	25.0507	**29.4783**
lightning	10	28.1238	34.8101	36.8311	34.7808	**37.7517**
	20	22.1436	31.0632	31.8819	31.3332	**36.4559**
	30	18.7537	28.7542	29.1397	28.9806	**34.2585**
	40	16.4704	26.9053	27.2644	27.1054	**31.1605**
	50	14.7755	25.3525	25.4163	25.4409	**30.009**
	60	13.5005	24.0056	24.005	24.1248	**27.9407**
man	10	35.8054	31.6216	36.5772	36.1712	**36.5772**
	20	29.9517	28.937	32.1374	31.4835	**32.1619**
	30	26.5438	27.3595	29.346	28.8883	**29.37**
	40	24.1766	26.2732	27.3991	27.057	**27.9497**
	50	22.3996	25.3267	25.9617	25.7318	**26.519**
	60	21.0517	24.6093	24.6524	24.5253	**25.3006**
hill	10	35.7995	32.2995	36.7512	36.2846	**36.7512**
	20	29.9779	29.6018	32.2903	31.7774	**32.2907**
	30	26.5894	28.0127	29.6441	29.2781	**29.9679**
	40	24.1366	26.8814	27.5982	27.3456	**28.3673**
	50	22.3602	25.9409	26.0126	25.8536	**27.0643**

	60	21.0194	25.1119	24.8655	24.7722	**25.9255**
lena	10	28.1171	31.4538	34.1918	33.2785	**34.1918**
	20	22.1234	28.752	30.4404	29.8219	**32.6679**
	30	18.6974	27.1272	28.0426	27.6784	**30.5616**
	40	16.3763	25.799	26.3306	26.1014	**29.0818**
	50	14.6111	24.7051	24.9242	24.8412	**26.6338**
	60	13.2838	23.6252	23.6124	23.6916	**26.1741**

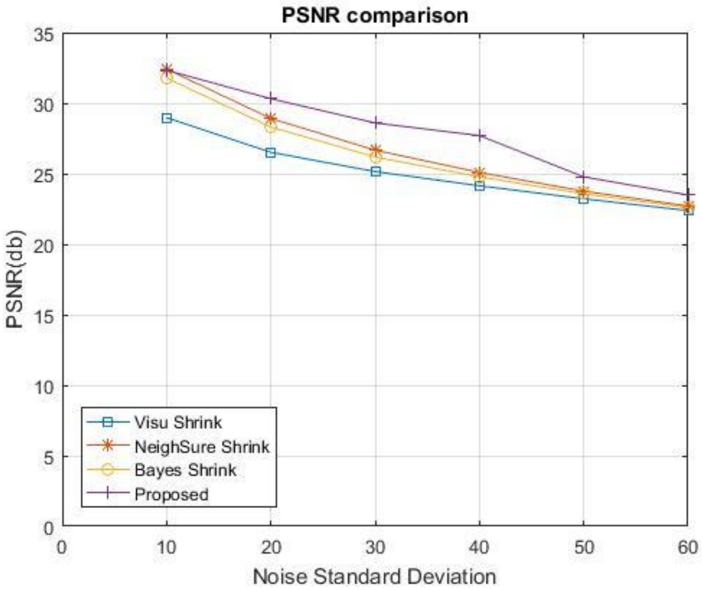


Figure 5.8: Comparison of PSNR values for boat image

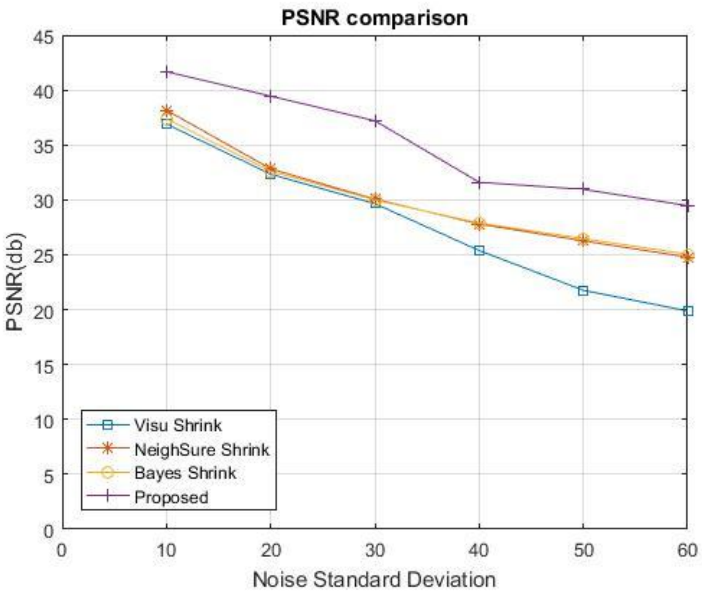


Figure 5.9: Comparison of PSNR values for glasses image

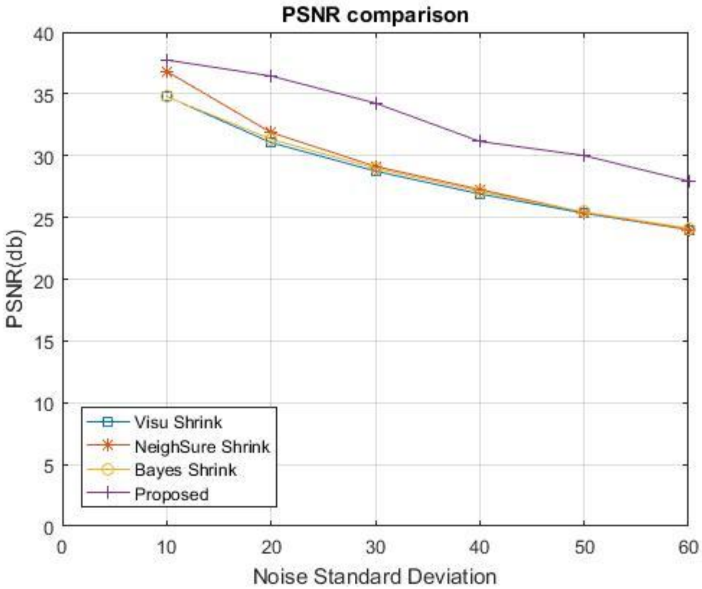


Figure 5.10: Comparison of PSNR values for lightning image

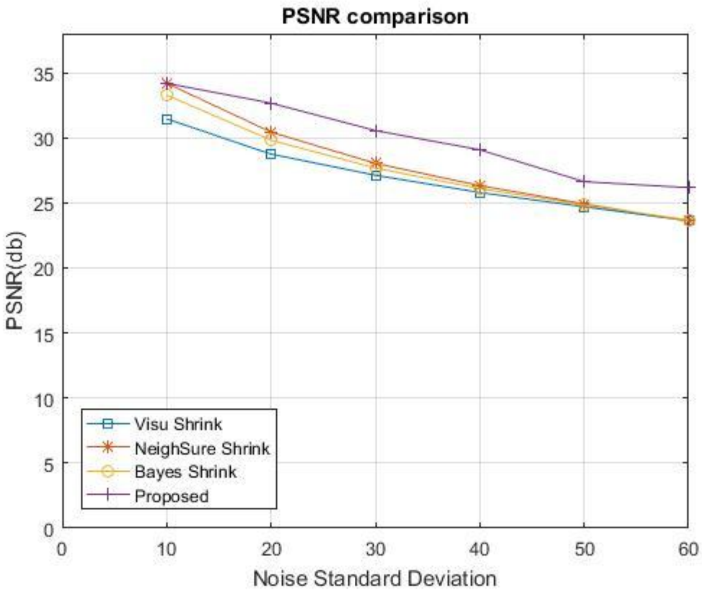


Figure 5.11: Comparison of PSNR values for lena image

Table 5.12: Comparison of SSIM for denoised images corrupted by Gaussian Noise

Image	σ	Noisy Image	Visu Shrink	NeighSure Shrink	Bayes Shrink	Proposed Genetic Algorithm
boat	10	0.69165	0.76473	**0.854**	0.83833	0.85313
	20	0.42721	0.67563	0.7539	0.72597	**0.80583**
	30	0.28954	0.60531	0.66217	0.63832	**0.75889**
	40	0.21171	0.54818	0.58834	0.5719	**0.73871**
	50	0.16234	0.49525	0.52007	0.5112	**0.62108**
	60	0.12828	0.45227	0.47017	0.46258	**0.59179**
glasses	10	0.44005	0.92379	0.92727	0.91034	**0.98071**
	20	0.18532	0.8043	0.7988	0.80072	**0.97332**
	30	0.10542	0.69442	0.689	0.69761	**0.96541**
	40	0.06961	0.53248	0.58221	0.59621	**0.92944**
	50	0.05112	0.38591	0.5166	0.52988	**0.92499**
	60	0.0388	0.31664	0.46319	0.4718	**0.91927**
lightning	10	0.46382	0.90677	0.91654	0.87344	**0.95922**
	20	0.20927	0.78889	0.78158	0.76993	**0.94989**
	30	0.12253	0.67483	0.65455	0.6654	**0.93354**
	40	0.08344	0.58502	0.57831	0.58276	**0.91747**
	50	0.06135	0.50129	0.47456	0.49277	**0.89685**
	60	0.04706	0.44117	0.40751	0.43279	**0.88374**
man	10	0.9349	0.8786	0.95472	0.94924	**0.95472**
	20	0.80688	0.80505	0.88739	0.86698	**0.88893**
	30	0.6843	0.75182	0.80789	0.78876	**0.8282**
	40	0.58269	0.70427	0.73235	0.71515	**0.77398**
	50	0.50309	0.65899	0.67296	0.6608	**0.70413**
	60	0.43901	0.62745	0.61315	0.6065	**0.67235**
hill	10	0.93661	0.87789	0.9532	0.9477	**0.9532**
	20	0.81022	0.8046	0.88427	0.86815	**0.88428**
	30	0.68412	0.74804	0.81203	0.79503	**0.82689**
	40	0.57108	0.69757	0.73446	0.71851	**0.77515**
	50	0.48632	0.65761	0.6688	0.65473	**0.72266**
	60	0.42251	0.62662	0.62055	0.61064	**0.68111**
lena	10	0.6134	0.84045	0.88219	0.85546	**0.88219**
	20	0.34361	0.75203	0.78225	0.75722	**0.86792**
	30	0.22183	0.66957	0.68653	0.67323	**0.80754**
	40	0.15842	0.59956	0.61265	0.60379	**0.80163**
	50	0.11943	0.53839	0.53931	0.54159	**0.75527**
	60	0.09475	0.48682	0.4754	0.48827	**0.74501**

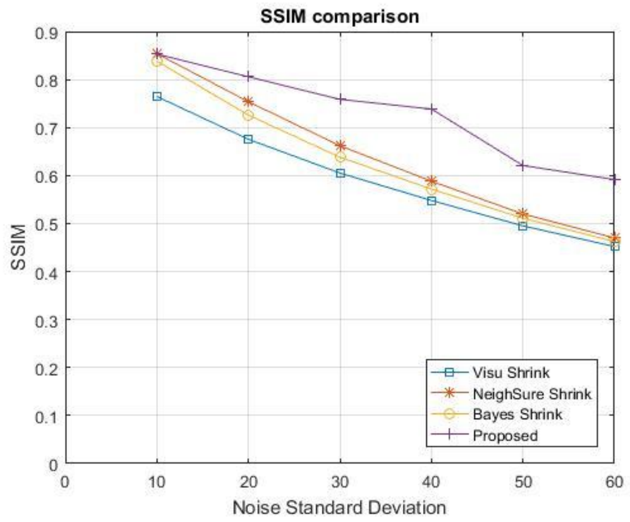


Figure 5.12: Comparison of SSIM values for boat image

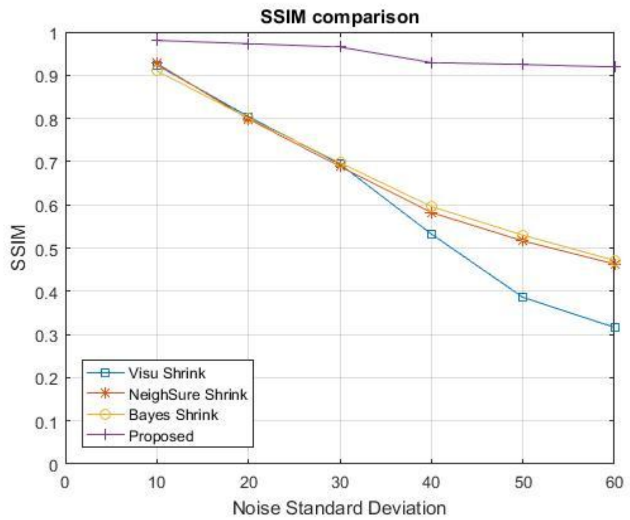


Figure 5.13: Comparison of SSIM values for glasses image

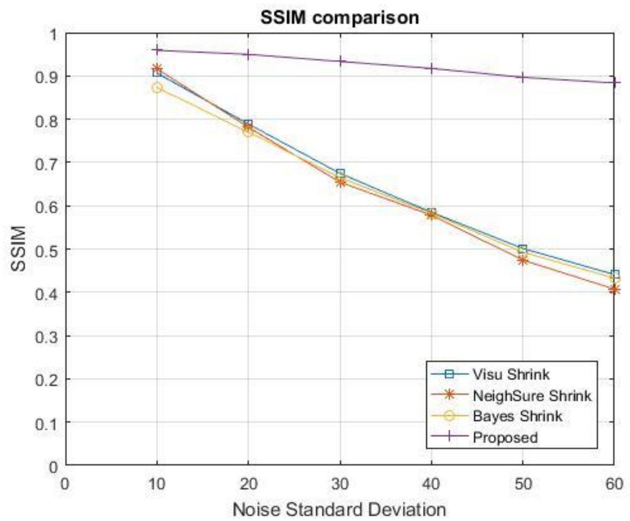


Figure 5.14: Comparison of SSIM values for lightning image

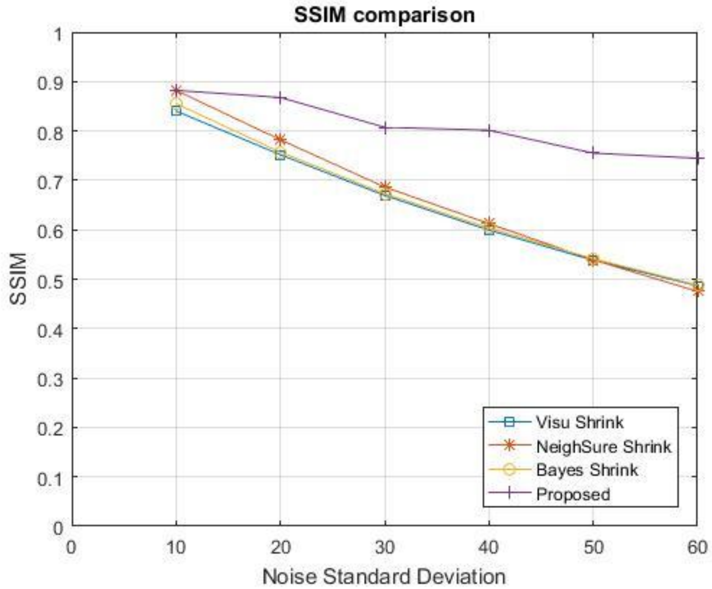


Figure 5.15: Comparison of SSIM values for lena image

5.4. Image with Speckle Noise

Boat and Lena image were taken and speckle noise with 6 different levels of noise was added. The noisy image was then used as input to the proposed algorithm and the algorithm was executed 10 times for each image. The comparison metrics were calculated and then it was compared with metrics obtained from wavelet based denoising methods.

(a) Noisy (σ=60)

(b) Proposed method

Figure 5.16: Denoising of Boat image corrupted by Speckle Noise

(a) Noisy (σ=60)

(b) Proposed method

Figure 5.17: Denoising of Lena image corrupted by Speckle Noise

Table 5.13: Comparison of PSNR for images corrupted by Speckle Noise

Image	V	Noisy Image	Visu Shrink	NeighSure Shrink	Bayes Shrink	Proposed Genetic Algorithm
boat	10	33.456	31.3082	35.6064	35.042	35.5917
	20	27.4475	28.9884	31.4285	31.1557	31.5466
	30	23.9576	27.4898	28.5641	28.7459	28.6143
	40	21.4781	26.4339	26.6008	27.0877	27.75
	50	19.5853	25.6272	25.0226	25.8143	26.8302
	60	18.0098	24.9516	23.5506	24.6082	25.7009
lena	10	33.7864	33.9609	36.9007	36.4534	36.9007
	20	27.7557	31.5216	32.068	32.3459	32.7983
	30	24.2561	29.8678	28.9011	29.6974	30.9998
	40	21.7968	28.6066	26.6029	27.7248	29.5763
	50	19.9182	27.6652	24.9726	26.3505	28.6377
	60	18.418	26.8529	23.6829	25.2651	27.6949

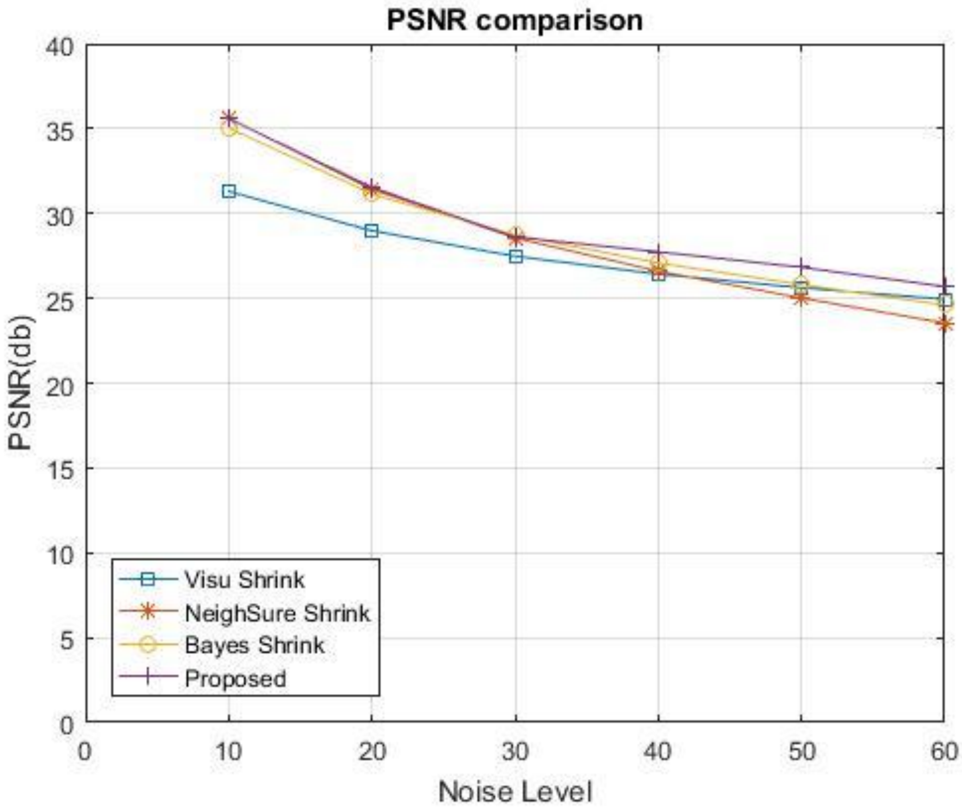


Figure 5.18: PSNR comparison of Boat Image Corrupted by Speckle Noise

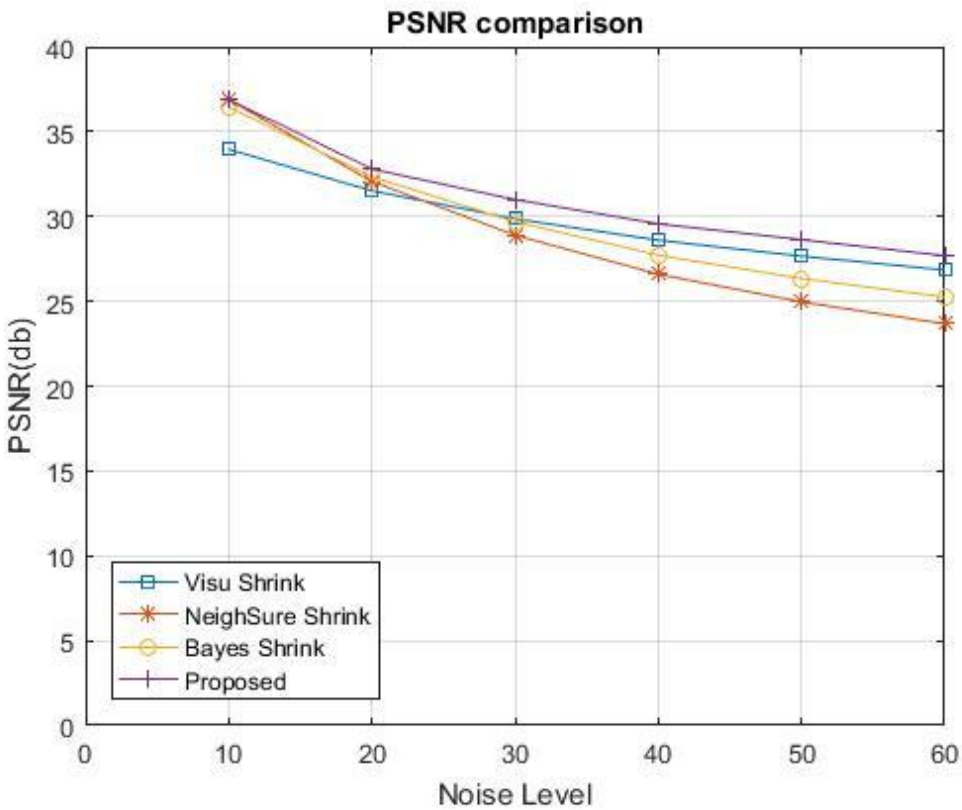


Figure 5.19: PSNR comparison of Lena Image Corrupted by Speckle Noise

Table 5.14: Comparison of SSIM for images corrupted by Speckle Noise

Image	V	Noisy Image	Visu Shrink	NeighSure Shrink	Bayes Shrink	Proposed Genetic Algorithm
boat	10	0.87345	0.82714	0.91667	0.90933	0.91596
	20	0.68509	0.76574	0.8282	0.81685	0.84944
	30	0.54629	0.71556	0.73136	0.73538	0.78531
	40	0.44908	0.67048	0.65307	0.67008	0.74223
	50	0.37824	0.63477	0.58651	0.61537	0.70976
	60	0.32202	0.60006	0.5236	0.56078	0.67082
lena	10	0.83271	0.88438	0.9126	0.90764	0.9191
	20	0.62472	0.83612	0.80158	0.80838	0.85367
	30	0.48713	0.78777	0.70182	0.72067	0.82229
	40	0.39448	0.73918	0.62163	0.64635	0.77615
	50	0.32604	0.69947	0.56065	0.59076	0.76805
	60	0.27611	0.65936	0.51293	0.54647	0.74189

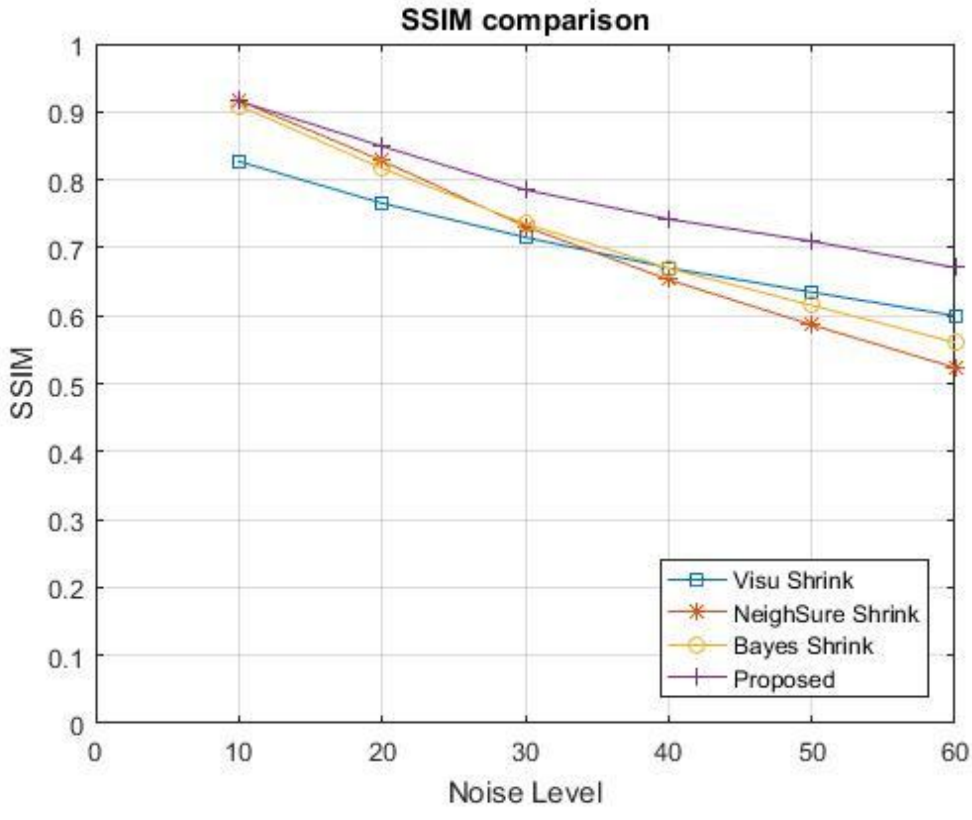


Figure 5.20: SSIM comparison of Boat Image Corrupted by Speckle Noise

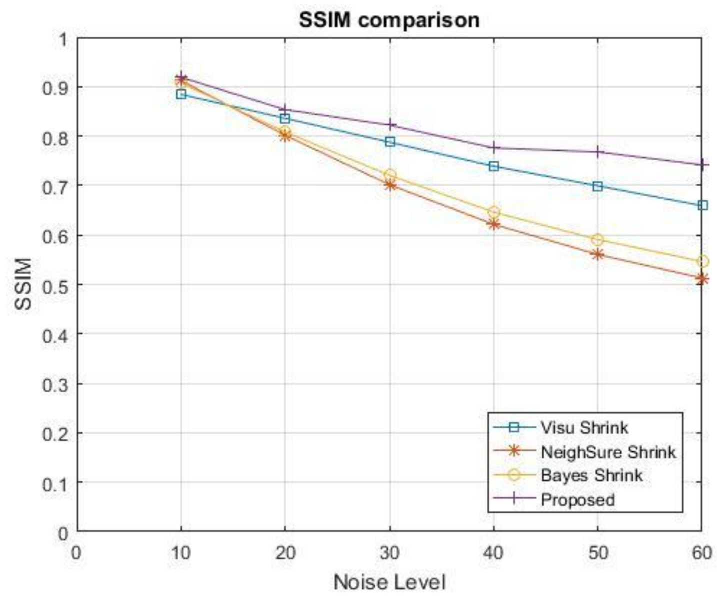


Figure 5.21: SSIM comparison of Lena Image Corrupted by Speckle Noise

5.5. Image with Salt & Pepper Noise

Boat image was taken and corrupted with SPN noise. It was then used as input to the proposed algorithm. The image obtained as output shows that this algorithm is not effective to remove the SPN noise. It is because the local search operators i.e Visu Shrink, Bayes Shrink & Neigh Shrink are not effective methods to remove the SPN noise.

Figure 5.22: Boat image corrupted with Salt & Pepper Noise of 0.1

Figure 5.23: Image obtained after applying the proposed algorithm to the Boat image corrupted by SPN of 0.1

5.6. Comparison with methods in literature

The results of the proposed algorithm for the test images were compared against other methods available in the literature. The methods used in the comparison were: Bivariate [17], Weiner-Chop [13], Median [3], Weiner [3], AD [8], BM3D [10], TV [4], HGA [21]. The values of PSNR & SSIM calculated for the above mentioned methods were obtained from [20]. Table 5.15 & 5.16 shows the results for PSNR & SSIM values.

Table 5.15: PSNR values of images obtained with the proposed algorithm compared against other state-of-the-art methods

Image	σ	Bivariate	Weiner-Chop	Median	Weiner	AD	BM3D	TV	HGA	Proposed GA
boat	10	26.7903	32.2421	29.4125	30.0429	**32.337**	31.0372	30.4606	32.2996	32.334698
	20	26.0863	29.3751	26.9582	28.1224	28.9834	**30.3156**	28.8781	29.2221	30.315589
	30	25.0151	27.6021	24.6743	26.3551	26.9763	28.5994	25.7193	27.5888	**28.599401**
	40	23.8048	26.3581	22.8231	24.8798	25.5933	22.3982	22.6886	25.8583	**27.705915**
	50	22.6273	**25.2238**	21.2298	23.6299	24.5193	17.6763	20.2274	24.7522	24.776816

	60	21.5751	**24.2287**	19.8736	22.5589	23.6157	15.2269	18.2512	23.9088	23.490799
glasses	10	36.5707	40.9602	35.1679	37.7402	40.0886	41.6732	40.1668	40.9327	**41.678712**
	20	32.3825	36.0197	29.4964	32.5533	35.8441	39.4528	33.6867	38.7093	**39.452837**
	30	29.4914	33.4845	26.1607	29.7962	33.359	35.5874	27.5652	35.3721	**37.206466**
	40	26.9682	31.0766	23.6213	27.6318	31.22	25.3897	23.6242	**32.9349**	31.610039
	50	25.1977	29.5027	21.8585	26.0461	29.6963	19.3816	21.1556	**31.5796**	30.962803
	60	23.5793	27.6438	20.3565	24.5026	27.9912	16.4151	19.2107	**29.7375**	29.478273
lightning	10	30.5364	37.5342	33.7314	36.007	38.3593	37.7517	37.4772	**38.9397**	37.751744
	20	29.0819	33.3033	28.989	31.9001	34.3706	**36.4559**	32.7605	35.8796	36.455873
	30	27.3286	31.1152	25.8219	29.1087	32.0056	33.4553	27.2108	33.7595	**34.258466**
	40	25.5988	29.4142	23.5385	26.8975	30.1831	24.2722	23.4488	31.1501	**31.160471**
	50	23.9835	27.9299	21.6647	25.1628	28.4836	18.4476	20.7659	29.8977	**30.009037**
	60	22.6705	26.6149	20.2309	23.8694	27.1088	15.8098	18.8113	**28.3782**	27.940654
man	10	28.1671	32.1111	30.2044	30.498	32.575	30.3763	30.7445	32.5008	**36.577183**
	20	27.188	29.4519	27.4023	28.5647	29.1575	29.9419	29.1906	29.4655	**32.161888**
	30	25.8903	27.8287	24.9632	26.8135	27.3148	28.4873	25.8917	27.753	**29.370019**
	40	24.5402	26.6115	22.9819	25.3515	26.1201	22.5147	22.8138	26.3014	**27.9497**
	50	23.2205	25.5153	21.2952	24.0202	25.0691	17.7983	20.3042	25.4661	**26.519036**
	60	22.0673	24.5344	19.9054	22.9644	24.1748	15.3813	18.3939	24.6525	**25.300648**
hill	10	28.1294	31.8511	29.9331	30.2655	32.2532	30.7287	30.5496	31.9964	**36.751176**
	20	27.2153	29.3742	27.3463	28.5634	29.0798	30.1539	29.1089	29.1272	**32.290679**
	30	25.9619	27.8733	24.9647	26.9788	27.367	28.7159	25.9542	28.0775	**29.967947**
	40	24.5148	26.6169	22.9656	25.4549	26.1664	22.4826	22.7956	26.5352	**28.367341**
	50	23.1516	25.5369	21.3353	24.1253	25.1843	17.8424	20.3461	25.4572	**27.064329**
	60	22.06	24.62	19.98	23.09	24.40	15.45	18.45	24.83	**25.92**

		55	77	58	72	13	92	38	5	**5453**
lena	10	30.1402	**34.3566**	32.1193	32.6629	34.1791	33.5725	33.0617	34.2869	34.191829
	20	28.737 7	31.5096	28.3629	29.9904	30.7859	**32.7177**	30.5887	31.8088	32.667882
	30	27.0225	29.6502	25.5344	27.8641	28.8337	30.4641	26.4766	30.0899	**30.561634**
	40	25.3627	28.0705	23.2989	26.0897	27.3802	23.0544	22.9908	28.1931	**29.081828**
	50	23.7615	26.7514	21.582	24.6371	26.2622	17.851	20.3899	**26.9147**	26.633832
	60	22.5478	25.6472	20.118	23.4853	25.2854	15.3379	18.4434	25.9642	**26.174106**
peppers	10	24.321	32.617	30.1027	30.6448	33.3773	31.5989	32.0986	**33.4444**	32.735626
	20	23.8865	29.2609	27.2841	28.4233	29.6071	**30.7666**	29.6484	29.9967	30.766571
	30	23.2153	27.113	24.8217	26.4935	27.2633	28.6545	25.9854	27.9267	**28.654638**
	40	22.3779	25.2522	22.7758	24.8508	25.6379	22.368	22.7096	25.7674	**26.865232**
	50	21.4984	23.8867	21.211	23.5504	24.2523	17.8708	20.3738	**24.3191**	24.064385
	60	20.7098	22.8394	19.8338	22.4435	23.155	15.4468	18.3824	**23.171**	23.077608
cameraman	10	24.0138	31.1959	26.5689	29.1468	**33.0887**	30.1921	29.1757	32.7442	32.081132
	20	23.505	27.8761	25.0051	27.1617	29.2375	29.6955	27.902	29.3795	**29.835802**
	30	22.7527	26.0024	23.4907	25.3307	26.7977	**28.2498**	25.4218	27.1361	28.116134
	40	21.8486	24.562	21.9719	23.6861	24.9921	22.795	22.6384	**25.2274**	24.176179
	50	20.8717	23.2894	20.6057	22.2428	23.3892	18.0795	20.2392	23.6481	**23.662178**
	60	20.035	22.4537	19.4807	21.2165	22.3219	15.736	18.4819	**22.6256**	22.411623
barbara	10	24.9574	30.8904	24.8824	28.0309	31.5467	32.2624	26.4833	30.4048	**32.296117**
	20	24.3989	27.1393	23.8288	26.2605	27.2607	**31.415**	25.7698	25.9776	27.692084
	30	23.6018	25.271	22.5829	24.7533	25.0973	**28.9539**	24.02	24.6277	28.433321
	40	22.6924	**24.1788**	21.34	23.5217	23.8676	22.461	21.8636	23.6568	23.840548
	50	21.7473	**23.3445**	20.1377	22.4813	23.0339	17.872	19.816	22.9547	23.017615
	60	20.8107	22.5836	19.0536	21.5639	22.3261	15.4864	18.1229	22.418	**22.675865**

couple	10	26.4161	31.622	29.0872	29.081	**32.3118**	30.9306	30.1002	32.1606	32.306473
	20	25.7284	28.8668	26.7042	27.3639	28.5495	**30.1695**	28.4947	28.6127	30.152793
	30	24.7387	27.227	24.5348	25.8136	26.4714	28.2985	25.4987	26.833	**28.312189**
	40	23.5538	25.9441	22.6839	24.44	25.1026	22.2221	22.5766	24.9835	**26.789514**
	50	22.4623	24.8957	21.1492	23.338	24.0949	17.528	20.1076	24.2056	**24.904231**
	60	21.4676	**24.0161**	19.7665	22.4	23.3459	15.1453	18.1722	23.5216	23.745353
fingerprint	10	25.845	**31.507**	29.1241	24.8826	30.5083	29.047	28.7845	30.2072	31.410669
	20	25.1507	**28.3756**	26.4634	24.2366	26.4974	28.3731	26.8703	26.3925	27.639023
	30	24.2251	26.2983	24.3536	23.4772	24.1052	**26.3179**	24.3241	24.3802	25.54342
	40	23.2473	**24.8435**	22.6007	22.7244	22.4698	21.2356	21.8539	22.7052	24.09448
	50	22.1369	**23.5772**	21.0414	21.9023	21.1856	17.4083	19.765	21.367	23.131654
	60	21.1997	**22.4922**	19.6725	21.1476	20.176	15.2148	18.0132	20.1606	21.038928

Table 5.16: SSIM values of images obtained with the proposed algorithm compared against other state-of-the-art methods

Image	σ	Bivariate	Weiner-Chop	Median	Weiner	AD	BM3D	TV	HGA	Proposed GA
boat	10	0.7381	**0.8571**	0.7801	0.7795	0.8516	0.8204	0.8025	0.8408	0.853129
	20	0.6774	0.7875	0.6475	0.7325	0.7626	0.8058	0.7615	0.7727	**0.805833**
	30	0.5969	0.7234	0.5247	0.6659	0.688	**0.7589**	0.6049	0.7309	0.758889
	40	0.519	0.6721	0.4334	0.5909	0.6277	0.4364	0.4497	0.6601	**0.73871**
	50	0.452	0.6163	0.3602	0.5209	0.5725	0.249	0.3385	**0.6267**	0.62108
	60	0.3988	0.5714	0.3042	0.4624	0.5275	0.1707	0.2613	**0.5991**	0.591792
glasses	10	0.9273	0.9642	0.8328	0.9465	0.9637	0.9807	0.9761	0.9761	**0.980708**
	20	0.7812	0.8974	0.5804	0.8093	0.9082	0.9733	0.8458	0.9685	**0.973318**

	30	0.6439	0.836	0.404	0.6754	0.8486	0.8873	0.5181	0.9463	**0.965412**
	40	0.5168	0.7597	0.2842	0.5649	0.7783	0.3311	0.2919	**0.9357**	0.929442
	50	0.4335	0.7141	0.217	0.4868	0.7326	0.1187	0.1915	**0.9257**	0.924994
	60	0.3704	0.669	0.1661	0.4205	0.6925	0.0663	0.1308	0.9178	**0.919269**
lightning	10	0.8928	0.9474	0.8238	0.9296	0.948	0.9592	0.9564	**0.9615**	0.959223
	20	0.7591	0.8808	0.5829	0.8346	0.89	**0.9499**	0.8357	0.943	0.949891
	30	0.6155	0.8089	0.4077	0.7004	0.8253	0.8544	0.5152	0.9137	**0.933543**
	40	0.5037	0.7476	0.2983	0.5779	0.766	0.3224	0.3066	0.9079	**0.917473**
	50	0.4115	0.6841	0.2263	0.4838	0.6967	0.1271	0.1983	**0.9012**	0.896849
	60	0.3423	0.6382	0.177	0.4164	0.6482	0.0742	0.1383	**0.8894**	0.883743
man	10	0.784	0.8777	0.8134	0.8067	0.8728	0.8176	0.8229	0.8668	**0.954717**
	20	0.7144	0.801	0.6679	0.7556	0.7734	0.8089	0.7787	0.7888	**0.88893**
	30	0.6301	0.7351	0.5375	0.6835	0.6992	0.7578	0.6079	0.7371	**0.828195**
	40	0.548	0.6769	0.4361	0.6032	0.6402	0.4209	0.4441	0.6797	**0.773982**
	50	0.478	0.6282	0.3568	0.5322	0.5893	0.2272	0.3283	0.6551	**0.704131**
	60	0.4194	0.5833	0.298	0.4726	0.5436	0.156	0.2518	0.6286	**0.672349**
hill	10	0.7441	0.8438	0.7797	0.7624	0.8421	0.7915	0.7798	0.8264	**0.953199**
	20	0.6833	0.7669	0.6463	0.7155	0.7365	0.7783	0.7454	0.7373	**0.88428**
	30	0.6077	0.7075	0.5194	0.6521	0.6633	0.7382	0.5963	0.7035	**0.826889**
	40	0.5264	0.6488	0.4173	0.577	0.6046	0.4205	0.4334	0.6467	**0.775146**
	50	0.4596	0.6025	0.3406	0.5097	0.5588	0.2221	0.3172	0.6067	**0.722659**
	60	0.4064	0.5636	0.2838	0.458	0.522	0.1472	0.2424	0.5893	**0.681108**
lena	10	0.8435	**0.8972**	0.822	0.8604	0.8882	0.8802	0.8697	0.8898	0.882187
	20	0.7552	0.8388	0.6525	0.798	0.8149	**0.8699**	0.8002	0.852	0.867923
	30	0.649	0.777	0.510	0.707	0.751	0.800	0.585	**0.813**	0.807

		2	4	2	3	3	2	5	**4**	544
	40	0.553	0.7137	0.4006	0.6094	0.688	0.3831	0.3987	0.7764	**0.801626**
	50	0.4694	0.6616	0.3262	0.5292	0.6371	0.1903	0.2849	0.7519	**0.755268**
	60	0.4155	0.6175	0.2704	0.469	0.5934	0.1287	0.2167	0.7423	**0.745009**
peppers	10	0.8342	0.9082	0.8485	0.8719	0.9073	0.8935	0.8949	**0.9096**	0.88978
	20	0.7657	0.847	0.7072	0.8235	0.8338	0.8817	0.8331	0.8614	**0.881723**
	30	0.6775	0.7901	0.575	0.7589	0.7642	0.8061	0.6506	**0.8215**	0.806128
	40	0.588	0.7235	0.4742	0.6758	0.7029	0.4455	0.4763	0.7789	**0.805952**
	50	0.5206	0.684	0.4016	0.605	0.6505	0.267	0.3692	**0.7564**	0.742671
	60	0.4652	0.6312	0.3446	0.5405	0.5972	0.1943	0.2923	0.7224	**0.729277**
cameraman	10	0.7612	0.8864	0.7758	0.817	**0.9111**	0.8617	0.8567	0.9053	0.883333
	20	0.677	0.8096	0.6146	0.7612	0.8325	0.8521	0.8007	0.8472	**0.863592**
	30	0.5693	0.7357	0.495	0.6923	0.7459	0.8066	0.621	0.7932	**0.810091**
	40	0.4752	0.6556	0.4025	0.6044	0.6667	0.4878	0.4584	**0.7422**	0.726493
	50	0.4115	0.6033	0.3435	0.5227	0.6004	0.3173	0.3584	0.7054	**0.71222**
	60	0.3586	0.5575	0.2927	0.456	0.5404	0.2209	0.2873	**0.6761**	0.661157
barbara	10	0.7573	0.8899	0.7343	0.7948	0.8917	**0.9089**	0.79	0.8778	0.905638
	20	0.6944	0.7955	0.6053	0.7382	0.7885	**0.8969**	0.7448	0.7577	0.803513
	30	0.6154	0.7228	0.4926	0.6718	0.7021	**0.8375**	0.6022	0.7012	0.829479
	40	0.5369	0.6581	0.4055	0.5948	0.6321	0.5144	0.4625	0.6521	**0.660336**
	50	0.4725	0.606	0.3381	0.5246	0.5789	0.3034	0.3608	**0.6198**	0.616647
	60	0.4154	0.5559	0.2866	0.4636	0.5319	0.2155	0.2882	0.5974	**0.59969**
couple	10	0.7156	0.867	0.7947	0.7637	0.8672	0.8439	0.8119	0.8558	**0.873354**
	20	0.6564	0.7892	0.6629	0.7123	0.7626	**0.8257**	0.7675	0.767	0.825156
	30	0.5786	0.7214	0.5456	0.646	0.6743	0.7668	0.6171	0.7071	**0.768117**

	40	0.5024	0.6603	0.4488	0.5698	0.6049	0.4443	0.4659	0.624	**0.721022**
	50	0.4383	0.6034	0.3751	0.5004	0.5458	0.2545	0.3508	0.5908	**0.629529**
	60	0.3894	0.5602	0.3183	0.4486	0.5041	0.1797	0.2751	0.5639	**0.578486**
fingerprint	10	0.8654	0.9608	0.9349	0.8275	0.9512	0.9246	0.9173	0.9436	**0.961261**
	20	0.8457	**0.9255**	0.8895	0.8126	0.8806	0.9191	0.8943	0.8677	0.914444
	30	0.8173	0.8861	0.8355	0.792	0.8061	**0.8883**	0.8369	0.8037	0.870503
	40	0.7865	**0.8504**	0.7785	0.7698	0.7365	0.7279	0.7559	0.7371	0.830023
	50	0.7448	**0.8098**	0.7175	0.7381	0.669	0.5426	0.6648	0.6689	0.80004
	60	0.7092	**0.7701**	0.6587	0.7066	0.6056	0.4255	0.5792	0.5949	0.703025

5.7. Validation

The dataset of images available in [20] corrupted by additive Gaussian noise N(0,σ) with six different values for standard deviation σ=10, 20, 30, 40, 50, 60 was taken. There were total of 66 noisy images. Different denoising algorithms were executed over these images and PSNR & SSIM values were calculated for each denoised images. The average PSNR & SSIM values for each noise deviation are calculated and tabulated corresponding to the respective methods in table 5.17 & 5.18.

Table 5.17: Average PSNR values for images in dataset [20] denoised by different methods

Average PSNR values						
Method	σ=10	σ=20	σ=30	σ=40	σ=50	σ=60
Bivariate	27.8079	26.6691	25.3858	24.0463	22.7872	21.7026
Weiner-Chop	33.3534	30.0502	28.1332	26.6299	25.4049	24.3347
Median	30.0304	27.0764	24.7184	22.782	21.1919	19.8434
Weiner	30.8184	28.4673	26.6168	25.048	23.7397	22.659
AD	33.6932	29.943	27.781	26.2485	25.0155	23.9911
BM3D	32.6519	**31.7689**	29.6167	22.8358	17.9778	15.5145
TV	31.7366	29.3544	25.8243	22.7276	20.3173	18.4306
HGA	33.6288	30.4156	28.504	26.6648	25.5056	24.4885
Proposed GA	**34.5559**	31.7665	**29.9112**	**27.4219**	**25.886**	**24.6599**

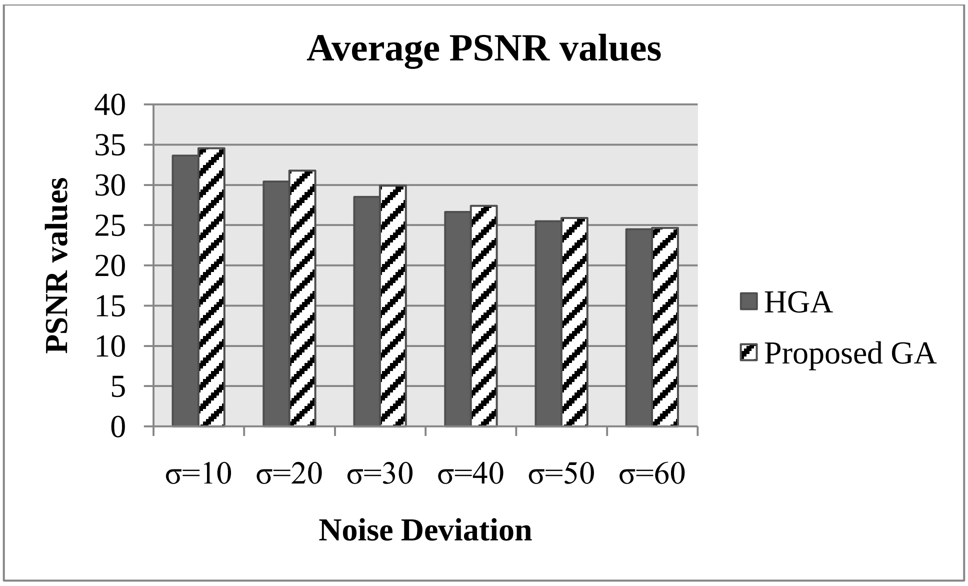


Figure 5.24: Average PSNR values for images in dataset [20] denoised by proposed GA & HGA

Table 5.18: Average SSIM values for images in dataset [20] denoised by different methods

Average SSIM values						
Method	σ=10	σ=20	σ=30	σ=40	σ=50	σ=60
Bivariate	0.80577	0.72816	0.63649	0.55054	0.48106	0.42642
Weiner-Chop	0.89997	0.83084	0.76767	0.70606	0.65575	0.61072
Median	0.81273	0.65973	0.53156	0.43449	0.3639	0.30913
Weiner	0.83273	0.77212	0.69505	0.61254	0.54122	0.48311
AD	0.89954	0.81669	0.74257	0.67708	0.62105	0.57331
BM3D	0.8802	0.86924	0.80928	0.44857	0.25629	0.17991
TV	0.86165	0.8007	0.61415	0.4494	0.34206	0.2694
HGA	0.89577	0.83303	0.78833	0.74007	0.70987	0.68375
Proposed GA	**0.91787**	**0.87805**	**0.8368**	**0.78911**	**0.73874**	**0.7059**

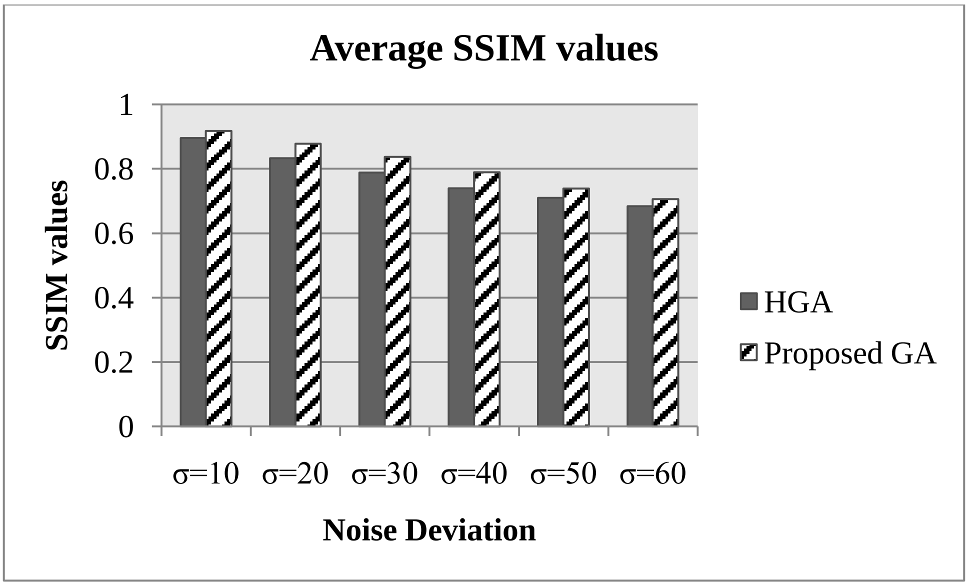


Figure 5.25: Average SSIM values for images in dataset [20] denoised by proposed GA & HGA

5.8. Analysis

Table 5.11 shows comparison of PSNR values for different images denoised by different methods. The entries highlighted in the bold are the best PSNR values among the compared methods. It clearly shows that the proposed genetic algorithm produces better PSNR values as compared to Visu Shrink, NeighSure Shrink & Bayes Shrink. Figure 5.8, 5.9, 5.10 & 5.11 show corresponding PSNR plot for Boat, Glasses, Lightning & Lena images respectively. From these plots, it is evident that the proposed genetic algorithm clearly produces denoised images with higher PSNR values than the compared wavelet based denoising methods.

Similarly table 5.12 shows comparison of SSIM values for different images denoised by different methods. The values highlighted in bold are the best values for that noise level. Figure 5.12, 5.13, 5.14 & 5.15 shows SSIM plot for Boat, Glasses, Lightning & Lena images respectively. From the table and these plots we can easily conclude that the proposed genetic algorithm produces denoised images having SSIM values far better that those produced by the mentioned wavelet denoising techniques.

Table 5.13 shows PSNR comparison for the images corrupted with speckle noise. Figure 5.18 & 5.19 shows PSNR plot for Boat and Lena image respectively. It can be seen that as noise level increase the proposed algorithm produces better PSNR values. Table 5.14 shows SSIM comparison for the images corrupted with speckle noise.

Figure 5.20 & 5.21 shows SSIM plot for Boat & Lena image. These plots show that the image obtained from the proposed algorithm is more structurally similar to the original image.

Table 5.15 & 5.16 shows PSNR and SSIM values of images obtained with the proposed algorithm compared against other state-of-the-art methods respectively. The values highlighted in bold shows the best value for the particular noise level. The proposed method was able to outperform other methods listed in the table in terms of PSNR & SSIM values.

Table 5.17 shows the average PSNR values for each noise level for images in dataset [20] denoised by different method. It shows the proposed genetic algorithm has higher PSNR values on average for each noise level as compared to other denoising method. Figure 5.24 shows the comparison between proposed genetic algorithm and HGA [21] in terms of PSNR. The proposed method produces higher PSNR values on average than that of HGA.

Similarly, table 5.18 shows the average SSIM values for each noise level for images in dataset [20] for different denoising methods. The values highlighted in bold shows the best average value for each noise level. It is seen from this table that the proposed genetic algorithm produces best SSIM values on average as compared to other methods. Figure 5.25 shows the comparison in terms of SSIM values between proposed genetic algorithm and HGA [21]. From this chart we can conclude that the proposed genetic algorithm produces more structurally similar image than HGA.

CHAPTER 6: CONCLUSIONS

The main objective of image denoising technique is to remove the noise content from noisy images while preserving relevant information, for instance, textures & edges. In this work Genetic Algorithm was proposed to be applied with image denoising using wavelets, where three different techniques i.e. Visu Shrink, Bayes Shrink & Neigh Shrink were used as mutation operators and helped to initialize and reinitialize the populations. A total of 6 levels of noise were applied to the test images and the algorithm was executed 10 times for each image. The proposed method was evaluated against Visu Shrink, Bayes Shrink & Neigh Shrink. The numerical values for PSNR & SSIM showed that the proposed method outperformed the mentioned denoising schemes, which indicates that the combination of GA with specific methods for image denoising can bring significant gain.

The proposed method was also compared with techniques available in the literature. The numerical values for PSNR & SSIM showed this method outperformed the denoising techniques mentioned in the literature.

CHAPTER 7: LIMITATIONS & FUTURE WORKS

The results obtained by the proposed method shows significant gain can be obtained against wavelet denoising methods by application of genetic algorithm. But this gain can be obtained on the cost of the execution time. The current stop criterion defines a specific amount of time to execute the method, so the proposed method is slow when compared to other methods. This is worse when several executions are necessary. Thus, stop criteria able to conclude the execution when a better image is already found can be under evaluation.

The fitness function was able to guide the algorithm through the solution space, but there may be cases where good images are not recognized by it. For instance, an image recovered by Bayes Shrink can have high values of PSNR or SSIM, but the fitness function does not assign to it a relevant value. This can occur once the fitness function makes a calculation without the original image, so it works without a clue about how far we are from the original image.

Also this method was unable to remove the SPN noise because the local search operators i.e. Visu Shrink, Bayes Shrink & Neigh Shrink are not effective methods to remove SPN noise. In future, methods which can remove SPN can be integrated as local search operators. Also, it can be improved to remove mixed type of noise in real world scenario.

This work did not focus on the computational cost of the method, but on the quality of the denoised images. Investigating the computational cost and reducing the current execution time may be a topic for future work.

CHAPTER 8: REFERENCES

[1] C. Toledo, L. de Oliveira, R. Dutra da Silva, H. Pedrini, “Image denoising based on genetic algorithm”, IEEE Congress on Evolutionary Computation (CEC), 2013,pp. 1294–1301.

[2] Yali Liu, “Image Denoising Method based on Threshold, Wavelet Transform and Genetic Algorithm”, International Journal of Signal Processing, Image Processing and Pattern Recognition(IJSIP), vol. 8, No. 2 (2015), pp 29-40.

[3] R.C. Gonzalez, R.E. Woods, “Digital Image Processing”, 3rd edition, Pearson, 2016.

[4] L. Rudin, S. Osher, E. Fatemi, “Nonlinear total variation based noise removal algorithms”, Phys. D 60 (1992) 259–268.

[5] A. Chambolle, “An algorithm for total variation minimization and applications”, J. Math. Imaging Vis. 20 (1–2) (2004) 89–97.

[6] C. Drapaca, “A nonlinear total variation-based denoising method with two regularization parameters”, IEEE Trans. Biomed. Eng. 56 (3) (2009) 582–586.

[7] P. Perona, J. Malik, “Scale-space and edge detection using anisotropic diffusion”, IEEE Trans. Pattern Anal. Mach. Intell. 12 (7) (1990) 629–639.

[8] V. Katkovnik, K. Egiazarian, J. Astola, “Local Approximation Techniques in Signal and Image Processing”, vol. PM157, SPIE Press, 2006.

[9] M.J. Black, G. Sapiro, D.H. Marimont, D. Heeger, “Robust anisotropic diffusion”, IEEE Trans. Image Process. 7 (3) (1998) 421–432.

[10] K. Dabov, A. Foi, V. Katkovnik, K. Egiazarian, “Image denoising with block-matching and 3D filtering” SPIE Electronic Imaging: Algorithms and Systems,vol. 6064, 2006, 606414-1-606414-12.

[11] R.D. da Silva, R. Minetto, W.R. Schwartz, H. Pedrini, “Adaptive edge-preserving image denoising using wavelet transforms”, Pattern Anal. Appl. 16 (4) (2013)567–580.

[12] V. Thavavel, J.J. Basha, M. Krishna, R. Murugesan, “Heuristic wavelet approach for low-dose EPR tomographic reconstruction: an applicability analysis with phantom and in vivo imaging”, Expert Syst. Appl. 39 (5) (2012) 5717–5726.

[13] S. Ghael, E.P. Ghael, A.M. Sayeed, R.G. Baraniuk, "Improved wavelet denoising via empirical wiener filtering", Proceedings of SPIE, San Diego, CA, USA, vol. 3169,1997, pp. 389–399.

[14] C. Deledalle, L. Denis, F. Tupin, "Iterative weighted maximum likelihood denoising with probabilistic patch-based weights", IEEE Trans. Image Process. 18 (12)(2009) 2661–2672.

[15] H. Ishikawa, "Global Optimization Using Embedded Graphs", Department of Computer Science, New York University, 2000 (Ph.D. thesis).

[16] V. Gupta, C.C. Chan, P.T. Sian, "A differential evolution approach to PET image denoising", in: 29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, 2007, pp. 4173–4176

[17] L. Sendur, I.W. Selesnick, "Bivariate shrinkage with local variance estimation", IEEE Signal Process. Lett. 9 (12) (2002) 439–441.

[18] D. Zosso and A. Bustina, "A Primal-Dual Projected Gradient Algorithm for Efficient Beltrami Regularization", 2014, preprint.

[19] Chen G. Y., Bui T. D. and Krzyzak A. (2004), "Image Denoising Using Neighbouring wavelet Coefficients" ICASSP Vol. 9, pp 917-920.

[20] http://hgaicmc.s3-website-sa-east-1.amazonaws.com/.

[21] J. Paiva, C. Toledo, H. Pedrini, "A Hybrid Genetic Algorithm for Image Denoising", IEEE Congress on Evolutionary Computation (CEC), Sendai, Japan, 2015.