CHAPTER I

Introduction

1.1 Introduction

Image processors could be categorized into different levels by the human vision standard. Lower-level ones are to clean and enhance observations, interpolate missing image data, or identify regions occupied by objects without telling what they are. Higher-level processors are to recognize object features and identify the associated hidden real-world contexts, such as face recognition for video surveillance and terrain reading for automatic piloting.

In this sense, the human vision system is a highly advanced and complex image processing senor. It automatically tells what people really want and discards the useless details. But for digital cameras, denoising becomes a hard task. No matter how good cameras are, an image improvement is desirable to extend their range of action.

There are a number of sources of image noise contamination.

Heat generated by cameras or external sources might free electrons from the image sensor itself, thus contaminating the *true* photoelectrons. These *thermal electrons* give rise to a form of noise called thermal noise or dark current.

Another type of noise is more akin to the *grain* obtained by using a high ISO setting (or high ISO film in a film camera). When we use a higher ISO, we are amplifying the signal we receive from the light photons. Unfortunately, as we amplify the signal, we also amplify the background electrical noise that is present in any electrical system.

In low light, there is not enough light for a proper exposure and the longer we allow the image sensor to collect the weak signal, the more background electrical noise it

also collects. In this case the background electrical noise may be higher than the signal.

Figure 1.1: An original color image

Figure 1.2 A color image with salt and pepper noise

Practically, these noises roughly have a Gaussian distribution. This is the so-called Amplifier noise or Gaussian noise. Amplifier noise is a major part of the *read noise* of an image sensor, that is, of the constant noise level in dark areas of the image [26].

Figure 1.3: (a) Clean image (b) image with additive salt and pepper noise.

Figure 1.4: (a) Clean image (b) image with additive gaussian noise, $\sigma = 30$.

Another primary noise is Salt and Pepper noise (Figure 3). An image containing Salt-and-pepper noise will have dark pixels in bright regions and bright pixels in dark regions. This type of noise can be caused by dead pixels, analog-to-digital converter errors, bit errors in transmission, etc [27, 28].

A digital image is composed of picture elements called pixels. Each pixel is assigned an intensity, meant to characterize the color of a small rectangular segment of the scene. A small image typically has around $256^2 = 65536$ pixels while a high-resolution image often has 5 to 10 million pixels. Some blurring always arises in the recording of a digital image; because it is unavoidable that scene information "spills over" to neighbouring pixels. For example, the optical system in a camera lens may be out of focus, so that the incoming light is smeared out. The same problem arises, for example, in astronomical imaging where the incoming light in the telescope has been

slightly bent by turbulence in the atmosphere. In these and similar situations, the inevitable result is that we record a blurred image.

Figure 1.5 (a) Clean image (b)blurred image.

In image deblurring, we seek to recover the original, sharp image by using a mathematical model of the blurring process. The key issue is that some information on the lost details is indeed present in the blurred image—but this information is "hidden" and can only be recovered if we know the details of the blurring process.

Unfortunately there is no hope that we can recover the original image exactly. This is due to various unavoidable errors in the recorded image. The most important errors are fluctuations in the recording process and approximation errors when representing the image with a limited number of digits. The influence of this noise puts a limit on the size of the details that we can hope to recover in the reconstructed image, and the limit depends on both the noise and the blurring process.

1.2 Signal to Noise Ratio

A digital image is generally expressed as a matrix of grey level (1D) or color values (n-D). In a movie, this matrix becomes 3D since the third one is corresponding to time. We use the pair (i; x(i)), the position and the value at this position, to express a

digital image. For a grey value image, *x*(i) is a scalar; and for a color image, *x*(i) is a 3D or 4D vector.

Mathematically, we can write the observed image captured by devices as:

$$y(i) = x(i) + n(i),$$

Where y*(i)* is the observed value, *x(i)* is the true value, which needs to be recovered from *y(i)* and *n(i)* is the noise perturbation.

For a grey value image, the range of the pixel value is (0; 255), where 0 represents black and 255 represents white. To measure the amount of noise of an image, one may use the signal noise ratio (SNR)

$$SNR = \frac{\sigma(y)}{\sigma(n)} \tag{1.4}$$

Where σ (*x*) denotes the empirical standard deviation of *x(i),*

$$\sigma(x) = \sqrt{\frac{\sum_i (x(i) - \bar{x})^2}{|l|}}$$

$$\sigma(n) = \sqrt{\frac{\sum_i (x(i) - y(i))^2}{|l|}}$$

Where $\bar{x} = \frac{\sum_i x(i)}{|l|}$ is the average of grey level values, computed from a clean image and *l* is the number of pixel in the image. As many signals have a very wide dynamic range, PSNRs are usually expressed in terms of the logarithmic decibel scale. In decibels, the PSNR is, by definition, 10 times the logarithm of the power ratio:

$$PSNR(dB) = 10log_{10}\left(\frac{1}{MSE(x(i),y(i))}\right) \tag{1.5}$$

$$MSE = \left(\frac{\sum_i (x(i)-y(i))^2}{\sum_i (x(i)-\bar{x})^2}\right) \tag{1.6}$$

Figure 1.6 (a) clean image (b) image with Gaussian noise, with standard deviation σ = 25, SNR=5.64dB.

Although PSNR is widely used in digital image processing, it can only be taken as one of the criteria to determine the quality of an image, otherwise, it might be misleading.

1.3 Literature Survey

One of the most important tasks in image processing applications is noise filtering. Noise may be added in the image during acquisition by camera sensors and transmission in the channel. Linear filtering techniques available for image de-noising tend to blur the edges [1]. In images edge contains essential information. Aim of any filtering techniques should preserve the edge information. Noise having Gaussian-like distribution is very often encountered in acquired data. Generally, the Gaussian noise is added to every part of the image and it affects each pixel in the image from its original value by a small amount based on noise standard deviation [2]. Several techniques were developed to remove Gaussian noise. Fuzzy filters are easy to realize by means of simple fuzzy rules that characterize a particular noise.

Russo [4] introduced a multi-pass fuzzy filter consisting of three cascaded blocks. Each block is hooked to a fuzzy operator that attempts to cancel the noise while preserving the image structure. Khriji and Gabbouj [5] developed a multi channel

filter using adaptive approach. These adaptive techniques are formed by a two-layer filter based on rational functions using fuzzy transformations of either the Euclidean or angular distances among the different vectors to adapt to local data in the color image. This filter preserves the edges and chromaticity of the image. D. Androutsos, *et al.* [6] described the strong potential of fuzzy adaptive filters for multichannel signal applications, such as color image processing is illustrated with several examples. K. Rank and R. Unbehauen [13], proposed an adaptive Recursive 2-D Filter for Removal of Gaussian Noise in Images. The adaptation is performed with respect to three local image features; edges, spots, and flat regions, for which detectors are developed by extending some existing methods. Tuan-Anh Nguyen *et al.* [12] proposed spatially adaptive denoising algorithm for a single image corrupted by the Gaussian noise. The algorithm is consisting of two stages; first noise detection and then noise removal filtering. Local weighted mean, local weighted activity and local maximum were defined to corporate desirable properties into denoising process.

Major problem in removing Gaussian noise is to differentiate between noise and edges. In [7], the effective fuzzy derivatives are used for differentiating the noise and edge pixels in images corrupted with Gaussian noise. Schulte et al. [8] consider the fuzzy distance between color pairs as a weight to perform the weighted average filtering for the removal of the Gaussian noise in color images. Russo [9] proposes a method for Gaussian noise filtering that combines a nonlinear algorithm for detail preserving and smoothing of noisy data, and a technique for automatic parameter tuning base on noise estimation. In [14], an robust approach is presented for image enhancement based on fuzzy logic that addresses the seemingly conflicting goals of image enhancement; (i) removing impulse noise, (ii) smoothing out nonimpulse noise,

and (iii) enhancing (or preserving) edges and certain other salient structures. A novel method of adaptive parameter selection is proposed in [15] which is an improved algorithm for adaptive fuzzy image enhancement. Using the concepts of both [14] and [16], three sigma and Pi filters are developed in [17]. An efficient fuzzy filter for edge preservation is proposed in [18] using fuzzy technique for color images. In [19] a new fuzzy-logic-control based filter is introduced with the ability to remove impulsive noise and smooth Gaussian noise, while preserving edges and image details.

In this work, we have extended previous work on noise reduction in color images [3] where Euclidian distance is used to measure the distance between the noisy pixels. The distance calculates dissimilarity between the central pixel and the neighbouring pixels. The proposed approach uses the concept of similarity between the central pixel and the neighbouring pixels. Some of the latest development s in this area is discussed as follows:

Tuan-Anh Nguyen *et al.*[24] propose a spatially adaptive denoising algorithm using local statistics for a single image corrupted by Gaussian noise. The proposed algorithm consists of two stages: noise detection and noise removal filtering. To corporate desirable properties into denoising process, local weighted mean, local weighted activity, and local maximum are defined. Using the local statistics, constraint for noise detection is defined. In addition, a modified Gaussian noise removal filter based on the local statistics is used to control the degree of noise suppression.

Tzu-Chao Lin [21] uses Decision-based fuzzy averaging (DFA) filter consisting of a D–S (Dempster–Shafer) noise detector and a two-pass noise filtering mechanism. Bodies of evidence are extracted, and the basic belief assignment is developed using the simple support function, which avoids the counter-intuitive problem of

Dempster's combination rule. The combination belief value is the decision rule for the D–S noise detector. A fuzzy averaging method, where the weights are constructed using a predefined fuzzy set, is developed to achieve noise cancellation. A simple second-pass filter is employed to improve the final filtering performance.

Yu Xiao *et al.* [23] proposed an l_1–l_0 minimization approach where the l_1 term is used for impulse denoising and the l_0 term is used for a sparse representation over certain unknown dictionary of images patches. The main algorithm contains three phases. The first phase is to identify the outlier candidates which are likely to be corrupted by impulse noise. The second phase is to reconvert the image via dictionary learning on the free-outlier pixels. Finally, an alternating minimization algorithm is employed to solve the proposed minimization energy function, leading to an enhanced restoration based on the recovered image in the second phase.

The peer group of an image pixel is a pixel similarity based concept which has been successfully used to devise image denoising methods. The fuzzy peer group concept, which extends the peer group concept in the fuzzy setting, is described [22]. A fuzzy peer group will be defined as a fuzzy set that takes a peer group as support set and where the membership degree of each peer group member will be given by its fuzzy similarity with respect to the pixel under processing. The fuzzy peer group of each image pixel are determined by means of a fuzzy logic-based procedure. The fuzzy peer group concept is used to design a two-step color image filter cascading a fuzzy rule-based switching impulse noise filter by a fuzzy average filtering over the fuzzy peer group. Both steps use the same fuzzy peer group, which leads to computational savings according to [22].

CHAPTER II

Introduction to Fuzzy Image Processing

2.1 **Fuzzy Logic**

E.H. Mamdani is credited with building the world's first fuzzy logic controller. Dr. Mamdani, London University, U.K., stated firmly and unequivocally that utilizing a fuzzy logic controller for speed control of a steam engine was much superior to controlling the engine by conventional mathematically based control systems and logic control hardware. Dr. Mamdani found that, using the conventional approach, extensive trial and error work was necessary to arrive at successful control for a specific speed set-point.

Fuzzy image processing is the collection of different fuzzy approaches to image processing that understand, represent and process the images, their segments and features as fuzzy sets. The representation and processing depend on the selected fuzzy technique and on the problem to be solved.

Fuzzy image processing has three main stages: image fuzzification, modification of membership values, and, if necessary, image defuzzification. The fuzzification and defuzzification steps are due to the fact that we do not possess fuzzy hardware. Therefore, the coding of image data (fuzzification) and decoding of the results (defuzzification) are steps that make possible to process images with fuzzy techniques. The main power of fuzzy image processing is in the middle step modification of membership values.

After the image data are transformed from gray level plane to the membership plane (fuzzification), appropriate fuzzy techniques modify the membership values. This can

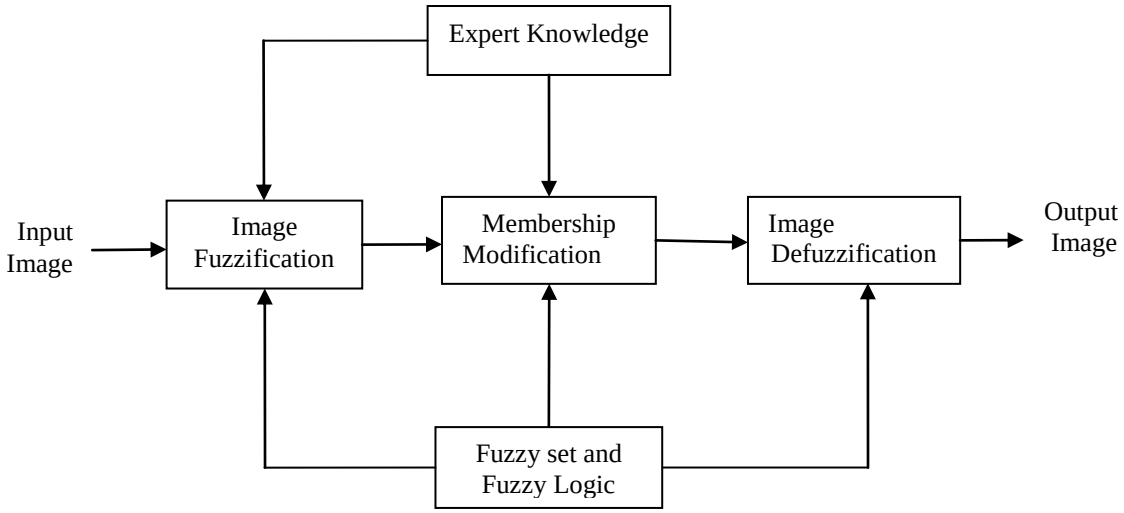

Figure 2.1: The structure of fuzzy image processing

be a fuzzy clustering, a fuzzy rule-based approach, and a fuzzy integration approach and so on.

2.2 **Reasons to use fuzzy logic in image processing**

Fuzzy techniques are powerful tools for knowledge representation and processing.

Fuzzy techniques can manage the vagueness and ambiguity efficiently. In many image processing applications, we have to use expert knowledge to overcome the difficulties (e.g. object recognition, scene analysis). Fuzzy set theory and fuzzy logic offer us powerful tools to represent and process human knowledge in form of fuzzy if then rules. On the other side, many difficulties in image processing arise because the data tasks results are uncertain. This uncertainty, however, is not always due to the randomness but to the ambiguity and vagueness. Beside randomness which can be managed by probability theory we can distinguish between three other kinds of imperfection in the image processing:

- Greyness ambiguity
- Geometrical fuzziness
- Vague (complex/ill-defined) knowledge

2.3 **Salient features of fuzzy logic**

(a) Fuzzy logic is conceptually easy to understand. The mathematical concepts behind fuzzy reasoning are very simple.

(b) Fuzzy logic is flexible. With any given system, it's easy to manage it or layer more functionality on top of it without starting again from scratch.

(c) Fuzzy logic is tolerant of imprecise data. Everything is imprecise if we look closely enough, but more than that, most things are imprecise even on careful inspection. Fuzzy reasoning builds this understanding into the process rather than tacking it onto the end.

(d) Fuzzy logic can model nonlinear functions of arbitrary complexity. We can create a fuzzy system to match any set of input-output data.

(e) Fuzzy logic is based on natural language. The basis for fuzzy logic is the basis for human communication. This observation underpins many of the other statements about fuzzy logic.

(f) Fuzzy logic can be blended with conventional control techniques. Fuzzy systems don't necessarily replace conventional control methods. In many cases fuzzy systems augment them and simplify their implementation.

CHAPTER III

Introduction to Cosine Similarity

Cosine similarity is a measure of similarity between two vectors by measuring the cosine of the angle between them. The result of the Cosine function is equal to 1 when the angle is 0, and it is less than 1 when the angle is of any other value. Calculating the cosine of the angle between two vectors thus determines whether two vectors are pointing in roughly the same direction.

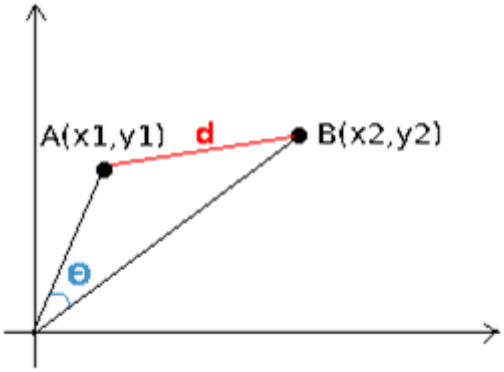

Figure 3.1: Graph depicting cosine similarity between two vectors

Let us consider two documents A and B represented by the vectors in the above figure 3.1. The cosine treats both vectors as unit vectors by normalizing them, giving us a measure of the angle Θ between the two vectors. It does provide an accurate measure of similarity but with no regard to magnitude. But magnitude is an important factor while considering similarity.

For example, the cosine between a document which has ‘machine’ occurring 3 times and ‘learning’ 4 times and another document which has ‘machine’ occurring 300 times and ‘learning’ 400 times will hint that the two documents are pointing in almost the same direction. If magnitude (euclidean distance) was taken into account, the results would be quite different.

Cosine of two vectors can be easily derived by using the Euclidean Dot Product formula:

$$A.B = ||A||\,||B||\cos\theta \quad (2)$$

Given two vectors of attributes, A and B, the cosine similarity θ is given as

$$similarity(A,B) = \cos(\theta) = \frac{A.B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=0}^{n}(A_i)^2} \times \sqrt{\sum_{i=1}^{n}(B_i)^2}} \quad (3)$$

where *i* is an integer value from 1 to *n.* The attribute vectors A and B are usually the term frequency vectors of the documents. The cosine similarity can be seen as a method of normalizing document length during comparison.

The resulting similarity ranges from −1 meaning exactly opposite, to 1 meaning exactly the same, with 0 usually indicating independence, and in-between values indicating intermediate similarity or dissimilarity.

There could be a number of ways we could combine the Euclidean distance and cosine similarity to determine the similarity measure. Another way to think of cosine similarity is as measuring the relative proportions of the various features or dimensions - when all the dimensions between two vectors are in proportion (correlated), than maximum similarity is obtained. Cosine similarity and Euclidean distance capture a lot of the same information. However whereas Euclidean Distance measures an actual distance between the two points of interest, Cosine can be thought of as measuring their apparent distance as viewed from the origin.

We have at our disposal two factors: one the cosine which gives us a measure of how similar two documents are, and the second the (Euclidean) distance which gives us the

magnitude of difference between the two documents. There could be a number of ways you could combine the two to determine the similarity measure.

The magnitude and cosine both provide us with a different aspect of similarity between two entities. It is up to us to either use them individually or in unison depending upon our application needs.

CHAPTER IV

Proposed Approach

4.1 **Fuzzy Filter for Image Restoration**

In proposed approach, a simple technique is introduced which deals with fuzzy filters that takes similarity between the color components as input. The output is the weighted average of the weights of all the neighbouring pixels that helps to compute the correction term for the Gaussian filters.

As Gaussian noise is additive, a color pixel in RGB color space with co-ordinates (x, y, z) degraded by random noise is expressed [3] as:

$$f(x, y, z) = I(x, y, z) + \eta(x, y, z) \quad (1)$$

Where f (x, y, z) is the noisy color image, I (x, y, z) is original color image both defined in RGB color space and η (x, y, z) represents the signal independent additive random noise in the same color space. The x and y represents the coordinates of the image pixel and z=1, 2, 3 represents the red, green and blue (RGB) color components (at x, y) respectively.

The methods for the reduction of Gaussian noise adopt the weighted average of neighborhood pixel values of the central pixel value [2]. The key point here is to select the weights to the neighborhood pixels in such a way as to obtain the corrected value. The use of color pairs to assign weights to the neighborhood pixels leads to a reduction in the ensuing artifacts. Adaptive fuzzy cosine similarity between the color pairs gives similarity between the central pixel and the neighborhood. This distance helps preserve edges by way of giving less weight to the noisy pixels and more weight to the similar pixels during the computation of the weighted average. Therefore, if

the adaptive fuzzy similarity value is more then more weight is assigned and vice-versa.

Adaptive fuzzy similarity is found between each color pair of the central pixel and that of the neighbourhood pixels. Color pairs are denoted in terms of the image function 'f' as follows [3]:

Red-Green (f (x, y, 1), f (x, y, 2))
Red-Blue (f (x, y, 1), f (x, y, 3))
Green-Blue (f (x, y, 2), f (x, y, 3)) (4)

Adaptive Similarity between a colour pair of central pixel and that of neighbourhood pixel, say between red-green pairs is found from:

$$S_{\mathrm{rg}}(x+i, y+j) = \frac{f(i,j,1) * f(x,y,1) + f(i,j,2) * f(x,y,2)}{(f(i,j,1)^2 + f(i,j,2)^2)^{1/2} * (f(x,y,1)^2 + f(x,y,2)^2)^{1/2}} \quad (5)$$

Where i, j are the neighbouring pixels and x, y is the central pixel for the window of size w x w. Similarly, we can find $S_{rb}(x+i,y+j)$ and $S_{gb}(x+i,y+j)$ for adaptive similarity between the red blue and green blue components. The adaptive fuzzy similarity between the color pairs is obtained by fuzzifying the adaptive similarity using the membership function Large to be introduced next.

4.2 **The Filter Structure**

In the proposed method the weighted average of the neighboring pixels in the window of interest is calculated. The weights to the neighboring pixels are determined according to the following fuzzy rules [3].

For the Red component

IF $S_{\mathrm{rg}}(x+i, y+j)$ is large AND $S_{\mathrm{rb}}(x+i, y+j)$ is large THEN weight w(x+i, y+j, 1) is a large. (6)

For the Green component

IF S_{rg}(x+i, y+j) is large AND S_{gb}(x+i, y+j) is large THEN weight w(x+i, y+j, 2) is a large (7)

For the Blue component

IF S_{rb}(x+i, y+j) is large AND S_{gb}(x+i, y+j) is large THEN weight w(x+i, y+j, 3) is a large. (8)

To express the degree to which an adaptive similarity is Large, the adaptive distances are fuzzified using the membership function Large, defined as:

$$\mu_l = \begin{cases} ae^{-\frac{(\lambda-b)^2}{2c^2}}, & \lambda \geq t \\ 0, & \lambda < t \end{cases} \quad (9)$$

This membership function for the set "Large" is shown in Figure 4.1.The parameter ***t*** is the minimum similarity between a color pair of a central pixel and that of the neighborhood in a window. The values for a, b and c are given in section 3.1. Parameter ***t*** for different color pairs is given as:

$$t_{rg}(x,y) = \min\left(S_{rg}(x+i, y+j)\right)$$

$$t_{rb}(x,y) = \min\left(S_{rb}(x+i, y+j)\right)$$

$$t_{gb}(x,y) = \min\left(S_{gb}(x+i, y+j)\right) \quad (10)$$

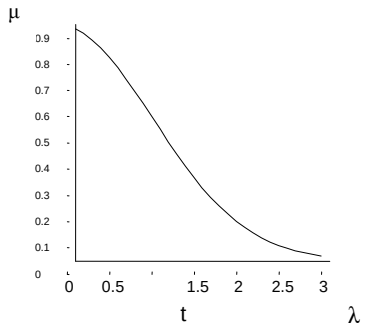

Figure 4.1: The membership Function for "Large"

The above fuzzy rules are implemented by calculating the adaptive fuzzy similarity using the membership function "Large". For example, fuzzy adaptive similarity between the red-green color pairs of a pixel at (x, y) and a neighboring pixel at

$(x+i, y+j)$ is represented as: $\mu_{rg(x,y)}(S_{rg} (x + i , y + j)$ where, μ_{rg} is the membership function of the red-green colour pair. The weights for a neighbouring pixel at the location (x + i, y + j) corresponding to red, green and blue components are derived from the three fuzzy rules [3] as:

$$w(x+i,y+j,1) = max\{\mu_{rg}(s_{rg}(x+i,y+j)), \mu_{rb}(s_{rb}(x+i,y+j))\}$$

$$w(x+i,y+j,2) = max\{\mu_{rg}(s_{rg}(x+i,y+j)), \mu_{gb}(s_{gb}(x+i,y+j))\}$$

$$w(x+i,y+j,3) = max\{\mu_{gb}(s_{gb}(x+i,y+j)), \mu_{rb}(s_{rb}(x+i,y+j))\} \quad (11)$$

The weights for red, green and blue components follow similarly and the final corrected value of a pixel at location (x, y) for the red component is given by [3]:

$$I(x,y,1) = \frac{\sum_{i=-k}^{k}\sum_{j=-k}^{k} w(x+i,y+j,1)\times f(x+i,y+j,1)}{\sum_{i=-k}^{k}\sum_{j=-k}^{k} w(x+i,y+j,1)} \quad (12)$$

Where k is the size of the window, similarly, we can find $I(x,y,2)$ and $I(x,y,3)$ for green and blue components respectively.

4.3 **Algorithm for the Gaussian Filter**

(a) Take a window of size *w*w* centred on the pixel of interest in the noisy image.

(b) Pick a neighbourhood pixel within the window and compute adaptive similarity for the three colour pairs (red-green, red-blue and green-blue) with the central pixel to this pixel via equation (5).

(c) Fuzzify each adaptive similarity using the membership function Large defined in equation (9), and parameter‘t’ using equation (10).

(d) Calculate weight for each colour component of the neighbourhood pixel using equation (11).

(e) Repeat Steps 2 to 4 for all neighbourhood pixels in the window.

(f) Obtain the final corrected value for the central pixel using equation (12).

The Figure 2 shows the flow chart for the proposed approach.

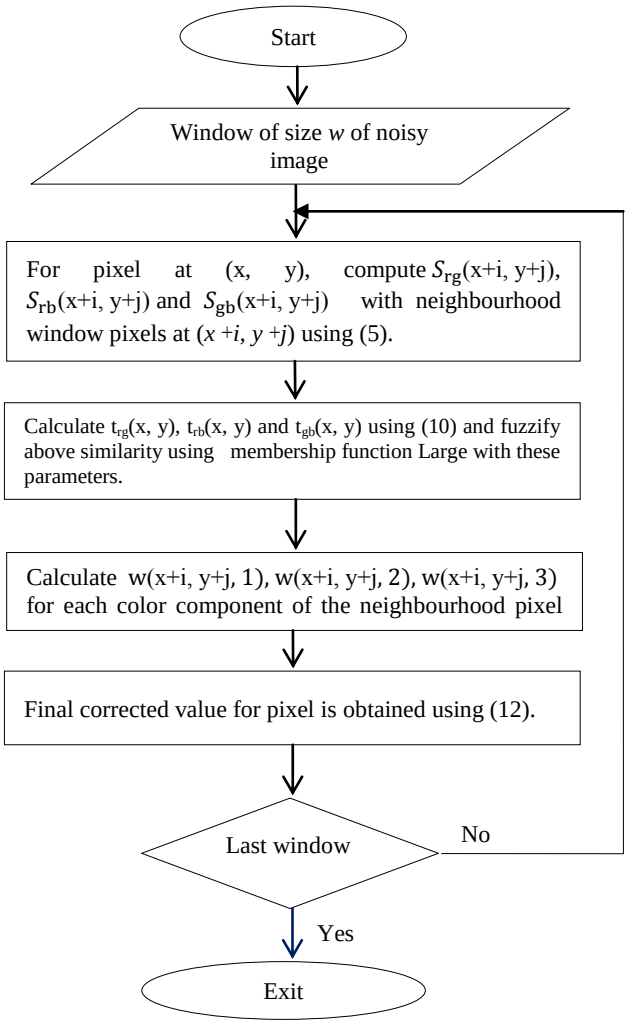

Figure 4.2: Flowchart for the proposed filter

Final Result *J* is obtained by merging the output results from the proposed filter *I* and the K-SVD algorithm K formulated as:

$$J = \alpha I + \beta K + \varepsilon \tag{5.13}$$

Where the values of α, β and ε are discussed in next chapter. The proposed algorithm in [35] is an iterative block-coordinate relaxation method .Software for the K-SVD algorithm can be found on internet [36]. The source code is freely available for academic and personal use. Description of K-SVD algorithm is given below, which is a generalization of the K-means clustering algorithm.

K-SVD algorithm [35]

Input: Noisy image f with additive Gaussian noise.

Parameters: λ (Lagrange multiplier); *C* (noise gain); *J* (number of iterations); *K* (number of the dictionary); *n* (size of image patch); σ (standard deviation of Gaussian noise).

Initialisation: set *u=f,* Let $D = (d_l \in \mathbb{R}^{n\times 1})_{l\epsilon\, 1\ldots K}$ be some initial dictionary.

Figure 5.1: Repeat J times

- Sparse Coding Stage: Use any pursuit algorithm to compute the representation vectors α_i for each example x_i to minimize the function:

$$\forall_{ij} min_{\alpha_{ij}} \left\|\alpha_{ij}\right\|_0 s.t \ \left\|R_{ij}u - D\alpha_{ij}\right\|_2^2 \leq (C\sigma)^2$$

- Dictionary Update Stage: For each column l= 1, 2,….,K in D.
  - Select the patches w_l that use this atom $d_{l,}$ $w_l = \{(i,j)|\alpha_{ij}(l) \neq 0\}$
  - For each patch $(i,j) \in w_l$, compute its residual without the contribution of the atom d_l

$$e_{ij}^l = R_{ij}u - D\alpha_{ij} + d_l\alpha_{ij}$$

  - Set $E_l = (e_{ij}^l)_{(i,j\in w_l)}$.Update d_l and the $\alpha_{ij}(l)$ using SVD decomposition of E_l.

- Reconstruction:

$$\hat{u} = (\lambda I + \sum\nolimits_{\text{ij}} R_{ij}^T R_{ij})^{-1}(\lambda f + \sum\nolimits_{\text{ij}} R_{ij}^T D\alpha_{ij})$$

Output: The reconstructed image *u*.

CHAPTER V

Experimental Results

A colour image consisting of an M × N × 3 array of pixel at locations (x, y) may be viewed as a "stack" of three scale images corresponding to RGB components. The colour images "Lena", "Parrot" and "Cafe" of size 256 × 256 impregnated with the Gaussian noise is considered as test images. The original images are shown in Figure 5.1.

Figure 5.1: Original colored Images used of size 256×256 Lena, Parrot, Café, Flower.

Experiments are performed using different sizes of windows and the results for these experiments are shown in the form of a plot between PSNR and Gaussian noise level (σ).The window sizes considered for the Lena image are: 3×3, 5×5 and 7×7. It can be

seen that the window size of 3×3 is the most suitable one for the noise level up to σ = 30 for which this Gaussian filter is best suited.

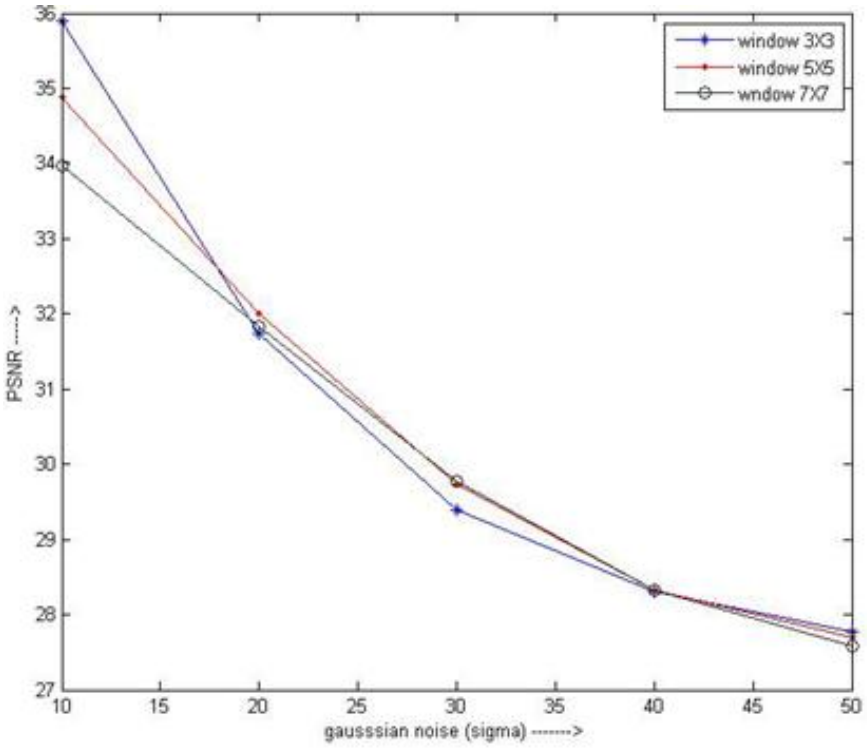

Figure 5.2: PSNR vs. Window sizes for Gaussian noise

The performance of Gaussian noise filter is evaluated over the three test colour images with σ = 10, 20 and 30. The parameters b = -0.2, c= 0.9 are found to be effective in the elimination of noise. The results of the proposed approach are compared in terms of MSE with methods developed by Om Prakash Verma *et al*. (FFNRCI) [3], Tzu-Chao Lin, Decision-based fuzzy image restoration for noise reduction based on evidence theory (DBFIR) [21], Restoration of images corrupted by mixed Gaussian-impulse noise via l1–l0 minimization (RICMG) [22], Fuzzy Peer Groups for Reducing Mixed Gaussian-Impulse Noise From Color Images [23] (FPGA), Spatially Adaptive Denoising Algorithm for a Single Image Corrupted by Gaussian Noise(SADA)[24].

TABLE 5.1 Comparison of PSNR of the proposed filter with FFNRIC[3], FPGA[23], DBFIR[21], RICMG[22] and SADA[24]

Variance	Noisy	FFNRIC	FPGA	DBFIR	RICMG	SADA	Proposed
Lena							
σ =10	31.94	36.22	35.78	35.20	35.73	35.72	36.15
σ =20	29.20	32.82	32.18	31.88	33.96	32.98	33.59
σ=30	28.38	30.31	31.33	30.19	31.60	31.12	31.84
Parrot							
σ =10	31.66	36.68	35.32	35.43	35.62	35.31	36.26
σ =20	29.05	32.50	32.11	31.02	31.01	32.54	33.60
σ=30	28.60	30.58	31.06	29.47	29.32	30.75	31.95
Café							
σ =10	32.19	36.76	35.12	34.21	34.99	34.97	36.21
σ =20	29.44	32.33	32.61	30.98	30.87	31.64	33.72
σ=30	28.54	30.74	29.36	29.89	29.01	29.77	31.77
Flower							
σ =10	32.15	36.39	35.01	33.21	35.93	34.11	36.10
σ =20	29.41	32.72	32.72	30.53	31.85	30.13	33.61
σ=30	28.59	30.37	29.47	29.47	28.65	28.43	31.88

Comparison of PSNR and MSE values resulting from the application of the proposed filter and the other methods in Table 5.1 and 5. 2 respectively show the superiority of the proposed filter over the other in the reduction of Gaussian noise. The results of denoising of the test images are illustrated in Figure 5.2.

TABLE 5.2 Comparison of MSE of the proposed filter with FFNRIC[3], FPGA[23], DBFIR[21], RICMG[22] and SADA[24]

variance	Noisy	FFNRIC	FPGA	DBFIR	RICMG	SADA	Proposed
Lena							
σ =10	38.69	16.58	15.34	16.21	15.44	15.67	14.23
σ =20	73.99	39.22	38.37	40.14	39.37	40.66	36.52
σ=30	89.54	68.52	67.45	66.66	59.92	67.54	54.31
Parrot							
σ =10	39.60	22.12	23.23	21.13	25.69	21.82	20.20
σ =20	72.52	40.12	39.45	42.49	41.54	42.71	41.02
σ=30	91.41	68.02	70.59	69.01	68.60	69.52	67.76
Café							
σ =10	40.18	17.27	18.01	18.74	16.61	17.78	15.42
σ =20	74.44	38.01	38.92	38.91	37.96	38.45	37.55
σ=30	92.81	54.21	57.21	57.29	56.31	57.78	56.63
Flower							
σ =10	37.28	18.12	18.43	17.89	16.88	18.17	16.42
σ =20	73.01	39.67	39.79	35.32	33.37	38.78	32.32
σ=30	95.18	63.29	64.26	55.18	63.78	58.63	48.63

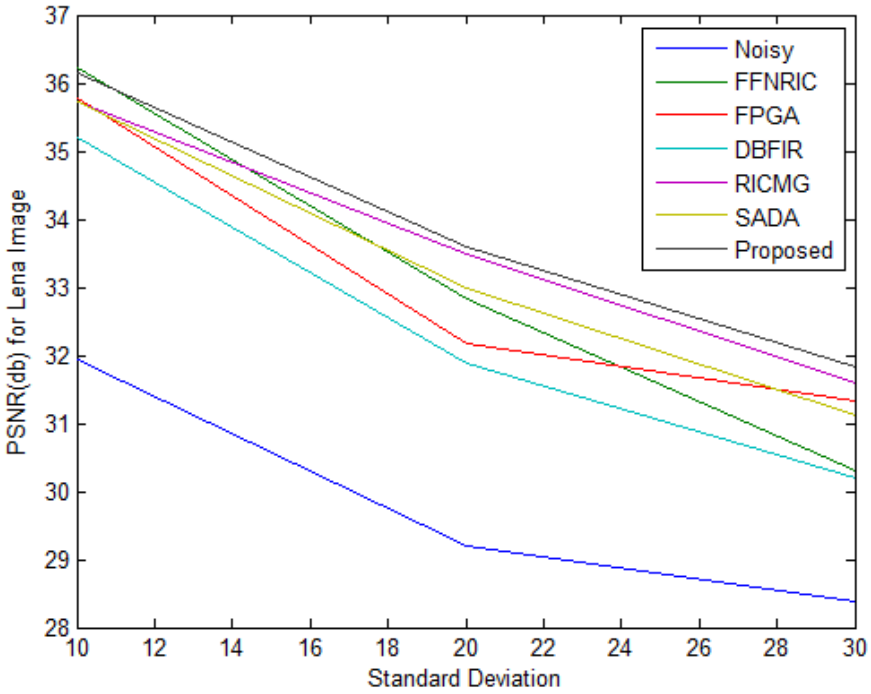

Figure 5.3: Comparison of PSNR (db) for Lena image at different noise levels σ with proposed and recent denoising techniques.

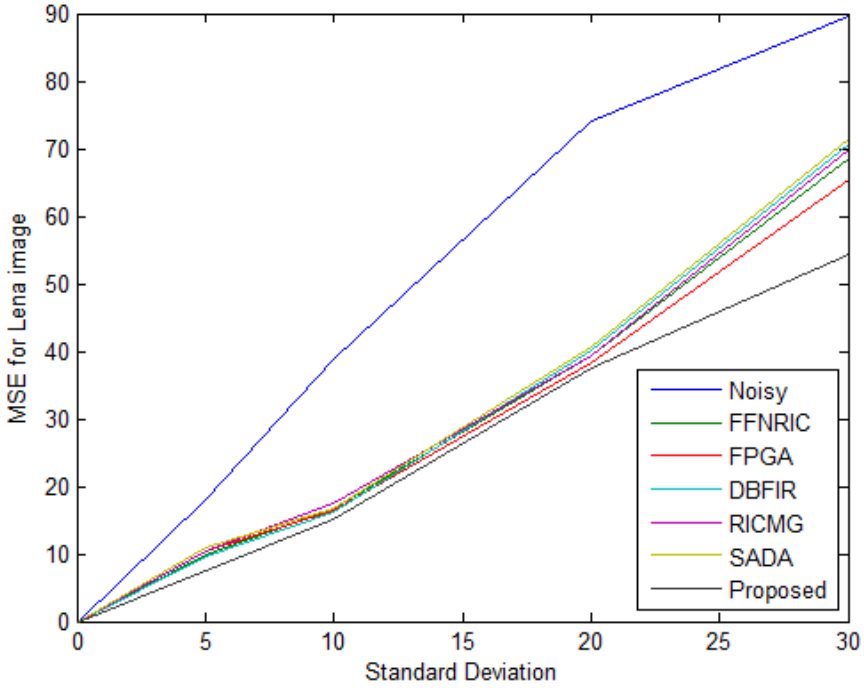

Figure 5.4: Comparison of MSE for Lena image at different noise levels σ with proposed and recent denoising techniques.

Blurring becomes pronounced as compared to the removal of noise. The undesirable effect of a Gaussian filter is that it blurs the edges. This process results in an image

with reduced “sharp” transitions in the gray levels. Hence the reduction of Gaussian noise is accompanied by the loss of finer details.

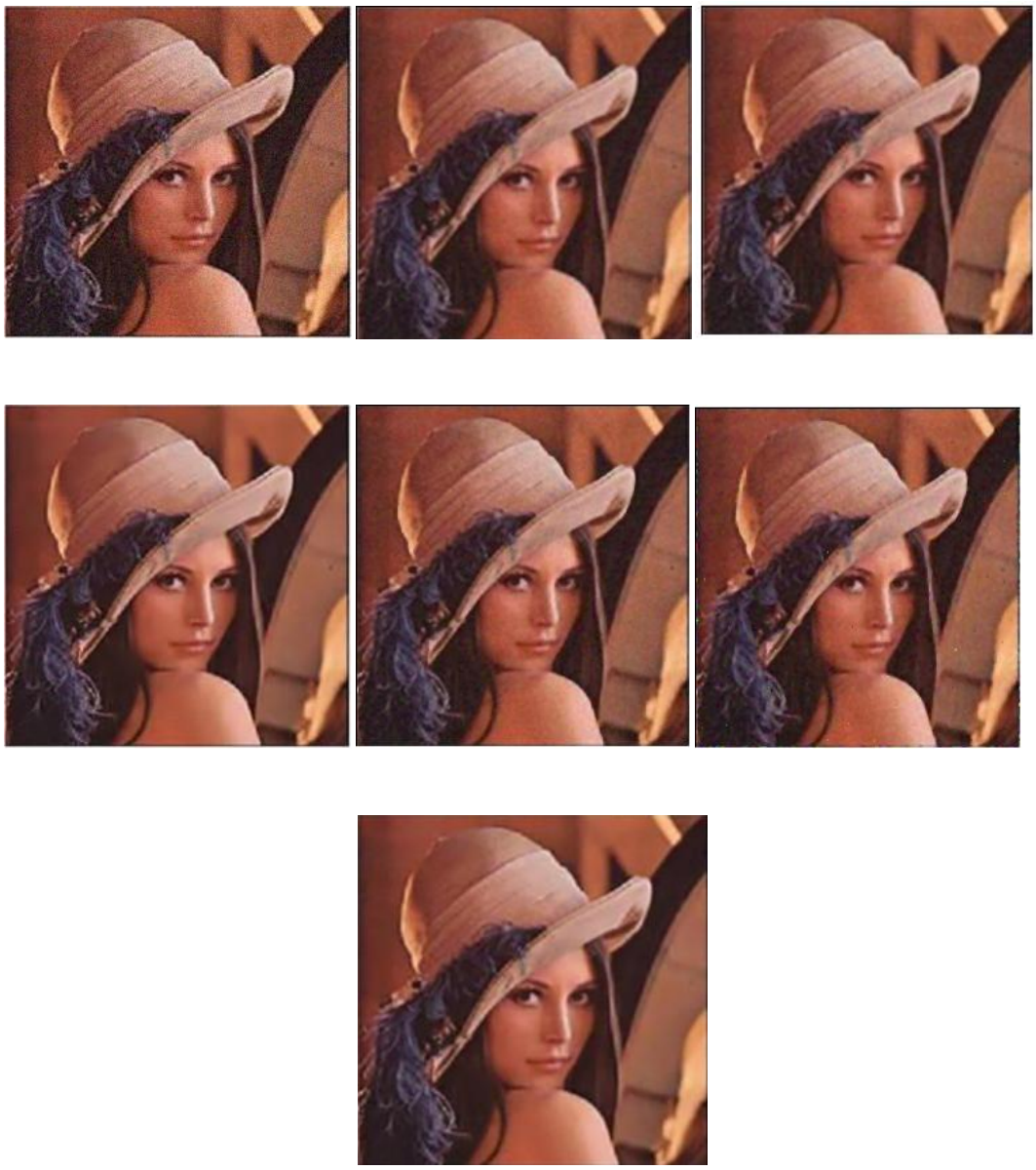

Figure 5.5: Denoised Lena images obtained with different filters while the input to each filter has the same Level of noise (A) with Gaussian noise σ =10, (B) FPGA, (C) SADA, (D) RICMG, (E) FFNRIC, (F) DBFIR, (G) the proposed method

Figure 5.6: Denoised Lena images obtained with different filters while the input to each filter has the same Level of noise (A) with Gaussian noise σ =20, (B) FPGA, (C) SADA, (D) RICMG, (E) FFNRIC, (F) DBFIR, (G) the proposed method

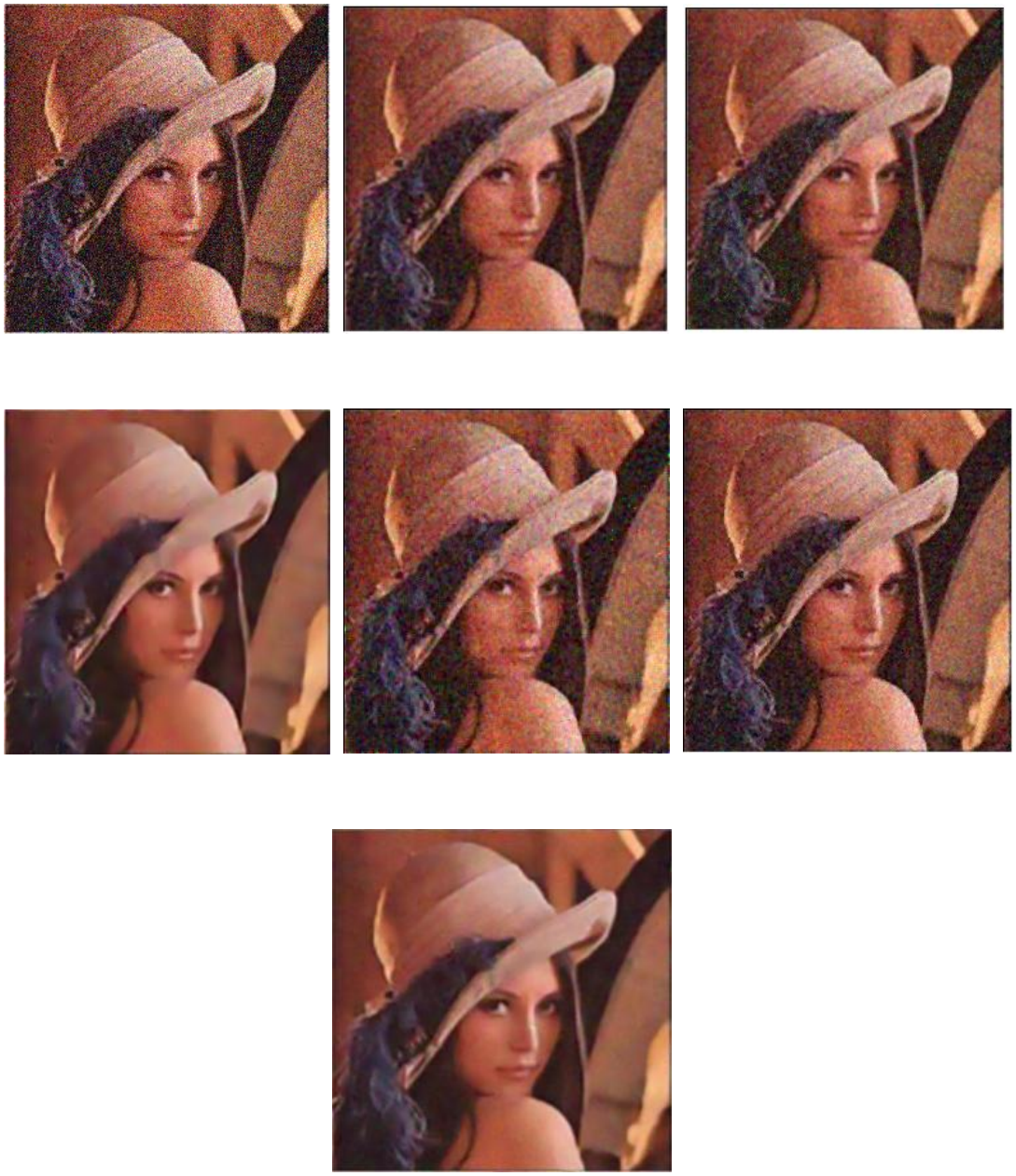

Figure 5.7: Denoised Lena images obtained with different filters while the input to each filter has the Same Level of noise (A) with Gaussian noise σ=30, (B) FPGA, (C) SADA, (D) RICMG, (E) FFNRIC, (F) DBFIR, (G) the proposed method

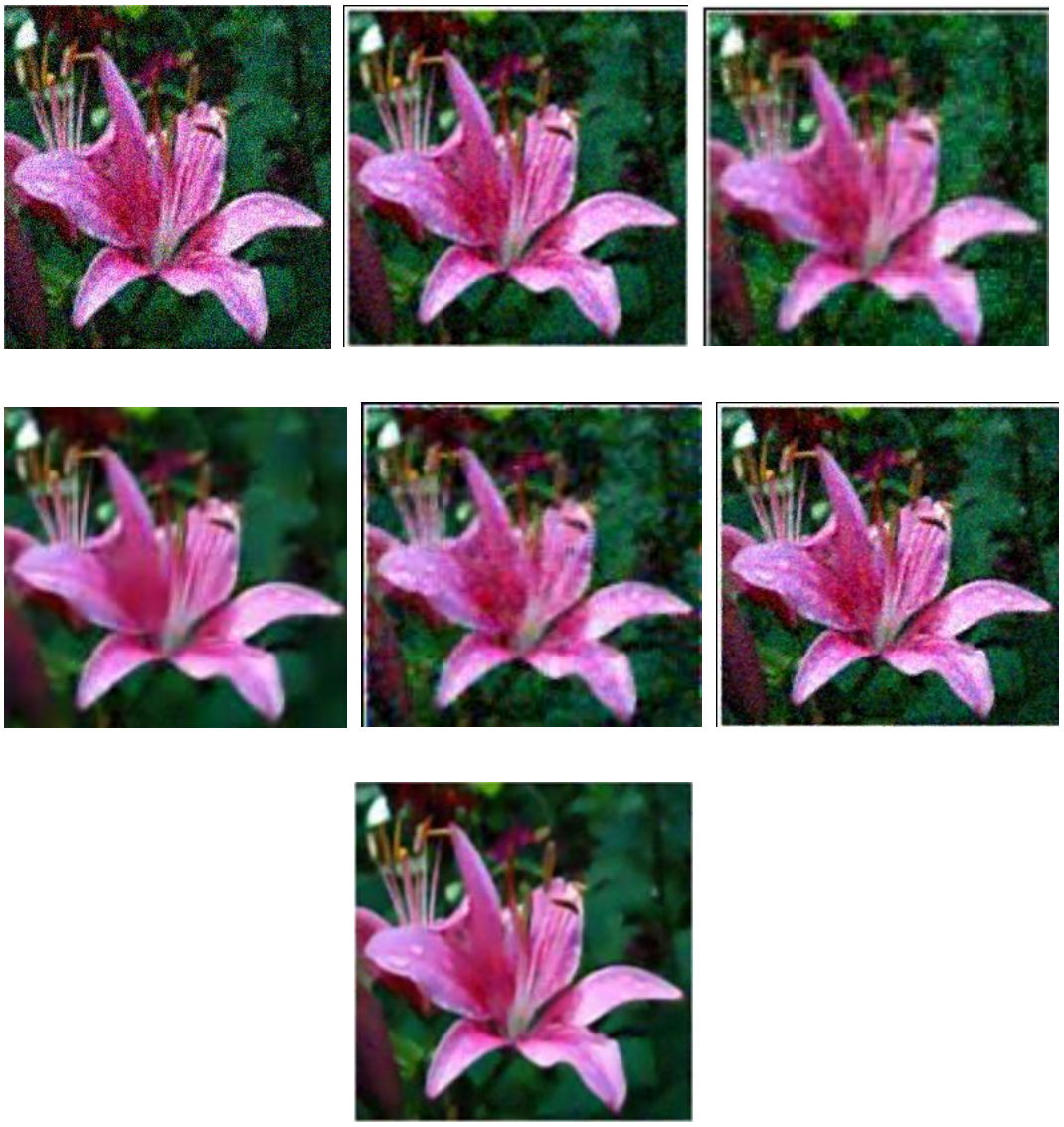

Figure 5.8: Denoised Flower images obtained with different filters while the input to each filter has the same Level of noise (A) with Gaussian noise σ =20, (B) FPGA, (C) SADA, (D)RICMG, (E) FFNRIC, (F) DBFIR, (G) the proposed method

Figure 5.9: Denoised Parrot images obtained with different filters while the input to each filter has the same Level of noise (A)with Gaussian noise σ =30, (B) FPGA, (C) SADA, (D) RICMG, (E) FFNRIC, (F) DBFIR, (G) the proposed method

Figure 5.10: Denoised Cafe images obtained with different filters while the input to each filter has the same Level of noise (A) with Gaussian noise σ =30, (B) FPGA, (C) SADA, (D)RICMG, (E) FFNRIC, (F) DBFIR, (G) the proposed method

Conclusion and Future Work

The removal of Gaussian noise is accomplished via fuzzy adaptive similarity between the colour components of a pixel of interest and the neighbourhood pixel. The Adaptive fuzzy similarity between colour pairs approach produces a de-noised image with all the significant details preserved. It was shown that this filter is capable of reducing Gaussian noise up to $\sigma = 40$. This is observed that the proposed filter produces the better results than the recent previous works. Future work will seek to reduce the blurring exceeding this level by way of designing a deblurring stage to the cascaded filter. The existing deblurring methods are not applicable as they are only designed for globally degraded images whereas we are concerned with locally degraded images caused by smoothening of the Gaussian noise.

The significance and relevance of the work is discussed for the noise removal. Firstly the design of the Gaussian filter differs from the existing methodology in terms of the usage of the fuzzy rule and membership function. Secondly, the use of the adaptive fuzzy similarity between colour components in the RGB space is new.

REFERENCES

[1] A.K.Jain, Fundamentals of digital image processing, Prentice Hall, Englewood Cliffs, 1989.

[2] Gonzalez R.C., Woods R.E: Digital Image Processing, Addison- Wesley, 2003.

[3] Om Prakash Verma, Madasu Hanmandlu, Anil Singh Parihar and Vamsi Krishna Madasu," Fuzzy Filters for Noise Reduction in Colour Images" ICGST-GVIP Journal, Volume 9, Issue 5, September 2009, ISSN: 1687-398X

[4] F. Russo and G. Ramponi, "A noise smoother using cascaded FIRE filters", Proc. of 4th Intl. Conf. on Fuzzy Systems, Vol.1, pp. 351-358, 1995.

[5] L. Khriji, M. Gabbouj, "A Class of Multichannel Image Processing Filters", IEICE Trans. on Information and System, Vol.E82-D, No.12, pp.1589-1596, 1999.

[6] D. Androutsos, K.N. Plataniotis, and A.N. Venetsanopoulos, "Colour image processing using vector rank filters", International Conf. on Digital Signal Processing, Vol. 2, pp.614-619, 1995.

[7] D. Van De Ville, M. Nachtegael, D. Van der Weken, E.E. Kerre, W. Philips, and I. Lemahie, "Noise Reduction by Fuzzy Image Filtering", IEEE Trans. on Fuzzy Systems, Vol. 11, 2003.

[8] S. Schulte, V. De Witte, and E.E. Kerre, "A Fuzzy Noise Reduction Method for Color Images", IEEE Transactions on Image Processing, Vol.16, No. 5. May 2007.

[9] F. Russo, "A Method for Estimation and Filtering of Gaussian Noise in Images", IEEE Trans. on Instrumentation and Measurement, Vol. 52, No. 4, 2003.

[10] Irfan T. Butt, Nasir M. Rajpoot," multilateral filtering: a novel framework for generic similarity-based image denoising", Image Processing (ICIP), 2009 16th IEEE International Conference, E-ISBN: 978-1-4244-5655-0

[11] Daichi Hanagaki and Yuko Osana, "Similarity-based image retrieval considering artifacts by self-organizing map with refract toriness", Neural Networks, 2009. IJCNN 2009. International Joint Conference, On page(s): 1095

[12] Tuan-Anh Nguyen, Won-Seon Song, Min-Cheol Hong," Spatially adaptive denoising algorithm for a single image corrupted by gaussian noise", Consumer Electronics, IEEE Transactions on Volume: 56 , Issue: 3, 2010 , Page(s): 1610 – 1615

[13] Rank, K., Unbehauen, R., "An adaptive recursive 2-D filter for removal of Gaussian noise in images", Image Processing, IEEE Transactions on Volume: 1 , Issue: 3, 1992 , Page(s): 431 - 436

[14] Y. S. Choi and R. Krishnapuram, "A Robust Approach to Image Enhancement Based on Fuzzy Logic", IEEE Trans. Image Proc.Vol. 6, pp.808-825, June 1997.

[15] Dawei Chen, Ying Qian," An Improved Algorithm of Adaptive Fuzzy Image Enhancement, Computer Science and Computational Technology, 2008. ISCSCT '08. International Symposium Volume: 1 Publication Year: 2008 , Page(s): 594 – 598

[16] M. Hanmandlu, D. Jha, B. Vittal and Vivek Gera, " Image smoothing using fuzzy clustering", Proc. of MM International Symposium on Information and Communication Technologies (M2USIC' 2001), Kaula Lumpur, Malaysia, pp.4.1(1)-4.1(5), 16-17 October 2001.

[17] H.S. Kam, M. Hanmandlu, and W.H. Tan, "An adaptive fuzzy filter system for smoothing noisy images", Proc. of Intl. Conf.on Convergent Technologies for Asia-Pacific Region, Vol.4 pp. 1614 – 1617, 2003.

[18] O.P. Verma, A. S. Parihar M. Hanmandlu, D. ,"An Edge Preserving Fuzzy Filter for Color Images", Computational Intelligence and Communication Networks (CICN), 2010 International Conference, On page(s): 122, Print ISBN: 978-1-4244-8653-3, INSPEC Accession Number: 11776685

[19] F. Farbiz, M.B. Menhaj, S.A. Motamedi, M.T. Hagan, "A new fuzzy logic filter for image enhancement", Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on Volume: 30 , Issue: 1 Publication Year: 2000 , Page(s): 110 – 119

[20] O.P. Verma ,"Fuzzy edge detection based on similarity measure in colour image", India Conference (INDICON), 2010 Annual, Digital Object Identifier: 10.1109/INDCON.2010.5712692, IEEE , Publication Year: 2010 , Page(s): 1 – 6

[21] Tzu-Chao Lin, "Decision-based fuzzy image restoration for noise reduction based on evidence theory", Expert Systems with Applications 38 (2011) 8303–8310.

[22] Samuel Morillas, Valentín Gregori, and Antonio Hervás, "Fuzzy Peer Groups for Reducing Mixed Gaussian-Impulse Noise From Color Images", IEEE Transactions On Image Processing, Vol. 18, No. 7, July 2009

[23] Yu Xiao, TieyongZeng, JianYu, MichaelK.Ng "Restoration of images corrupted by mixed Gaussian-impulse noise via l1–l0 minimization", Pattern Recognition 44 (2011) 1708–1720.

[24] Tuan-Anh Nguyen, Won-Seon Song, and Min-Cheol Hong, *Member*, IEEE, "Spatially Adaptive Denoising Algorithm for a Single Image Corrupted by

Gaussian Noise". IEEE Transactions on Consumer Electronics, Vol. 56, No. 3, August 2010.

[25] Michael Elad, On the Origin of the Bilateral Filter and Ways to Improve It

[26] Junichi Nakamura (2005). Image Sensors and Signal Processing for Digital Still Cameras.

[27] Linda G. Shapiro and George C. Stockman (2001). Computer Vision.

[28] Charles Boncelet (2005). Image Noise Models. in Alan C. Bovik. Handbook of Image and Video Processing.

[29] P. Perona and J. Malik, Scale space and edge detection using anisotropic diffusion, IEEE Trans. Patt. Mach. Intell., 12, pp. 629-639, 1990.

[30] F. Guichard, J.M Morel and R. Ryan, Image Analysis and P.D.E.'s.

[31] L. Alvarez and P-L. Lions and J-M. Morel, "Image selective smoothing and edge detection by nonlinear diffusion (II)", SIAM Journal of numerical analysis 29, pp. 845-866, 1992

[32] Y. Wang and H. M. Zhou, "A Total Variation Wavelet Algorithm for Medical Image Denoising , the International Journal on Biomedical Imaging, Volume 2006, article ID 89095, 6 pages, 2006.

[33] M. Aharon, M. Elad, A. Bruckstein, The K-SVD: an algorithm for designing of over complete dictionaries for sparse representation, IEEE Transactions on Image Processing 54 (2006) pp. 4311–4322.

[34] M. Elad, M. Aharon, Image denoising via sparse and redundant representations over learned dictionaries, IEEE Transactions on Image Processing 15 (2006) pp. 3736–3745.

[35] M. Aharon, M. Elad, A. Bruckstein, The K-SVD: an algorithm for designing of overcomplete dictionaries for sparse representation, IEEE Transactions on Image Processing 54 (2006) pp. 4311–4322.

[36] http://www.cs.technion.ac.il/~ronrubin/software.html